ISOTENSOR DECAYS IN LIGHT NUCLEI

by

Roy Edgar Fauber

A dissertation submitted to the faculty of the University of North Carolina at Chapel Hill in partial fulfillment of the requirements for the degree of Doctor of Philosophy in the Department of Physics and Astronomy.

Chapel Hill
1990

Approved by:

Edward G. Ludwig Advisor

W. J. Thompson Reader

Thomas B. Clegg Reader

ROY EDGAR FAUBER. Isotensor Decays in Light Nuclei (Under the direction of Edward J. Ludwig.)

ABSTRACT

As part of an ongoing investigation of isospin mixing in light nuclei, we have studied alpha particle decays from the lowest $T = 2$ states in ^{24}Mg, ^{28}Si and ^{32}S. These decays furnish unique insight into T-mixing mechanisms since they are sensitive to only the isotensor ($\Delta T = 2$) component of the mixing. Forbidden when one assumes strict isospin conservation, decays from the $T = 2$ levels in question manifest themselves as narrow (< 1 keV) resonances in cross section excitation functions. While the $T = 2$ state in ^{28}Si has been observed through ^{24}Mg(α,α)^{24}Mg, small alpha branching ratios necessitate the study of the $T = 2$ levels in ^{24}Mg and ^{32}S via the (p,p) and (p,α) reactions. Specialized hardware and software were developed to obtain the resonance data and to measure the beam energy spread. Additional programs were written to analyze the cross section data by utilizing a Breit-Wigner isolated resonance model, and the appropriate resonance parameters were extracted from the best fits determined by a minimum χ^2 criterion. These methods of data acquisition and analysis have provided determinations of resonance widths, resonance energies, alpha partial widths, and reduced widths. These results are compared with previously determined values for the lowest $T = 2$ states in nuclei in this mass region. The only definite trend observed for the reduced widths in this mass region is a general increase with increasing mass number.

ACKNOWLEDGEMENTS

First, I would like to thank my parents who made everything possible. Their love, understanding, and guidance provided a solid foundation for life in and out of graduate school. Thanks to my mother for showing me, in the last few years, what true courage is, for giving me the strength to continue, and for helping me keep things in perspective.

Secondly, I would like to thank all those people who helped me to finish graduate school. Over the considerable time of my graduate school "career," in an experimental program such as ours, there are many people who contribute to the research effort. I hesitate to attempt to name the people involved with this work, for fear of inadvertently omitting some. However, as is customary, I will try to acknowledge the main perpetrators. Of course, I would like to thank my principal advisor, Dr. Ed Ludwig, and the other members of the UNC faculty from whom I often solicited advice: Drs. T. B. Clegg, W. J. Thompson, and H. J. Karwowski. Thanks also to these people, along with Dr. Y. J. Ng, for serving on the dissertation committee and making many helpful suggestions.

As a result of my graduate school longevity, several "generations" of graduate students and visiting scholars have assisted in the project. Among those deserving thanks are Drs. B. Burks, J. Wilkerson, R. Varner, R. Anderson, T. McAbee, M. Schneider, and J. Bowsher. Also, thanks to T. Spencer, D. Abbott, T. Gibbons, E. Crosson, and S. Lemeiux. One person left out of the above list who deserves special thanks is Dr. T. M. Mooney. As a co-worker in high resolution experiments, he is responsible for much of the necessary hardware and played an integral part in developing the data taking and

analysis software. The importance of his role is evidenced by the many references to him in this work.

I gratefully acknowledge the essential aid of the TUNL personnel and faculty under the direction of Dr. E. G. Bilpuch with particular thanks to Messrs. R. Rummel, P. Carter, S. Edwards, P. Mulkey, and C. Westerfeldt. Also, thanks to members of the TUNL and UNC instrument shops. Without these people, there would be no data.

Next, I would like to thank all the people who helped me **not** to finish graduate school. These people, creating the diversions which took time away from my work, are much too numerous to mention here, but their contributions continue to evoke many fond memories. Among this "real world" cast are volleyball, softball, basketball, and tennis players as well as the boys at the Oasis. I would like to think that the substantial time spent in finishing is somehow proportional to the many friendships which I developed.

Finally, I would like to thank my wife Cathy, whose contributions span most of the above categories and beyond. She has endured my neglect, frustration, and sadness, and shared in my happiness, all with patience and understanding. Her contributions are immeasurable, her friendship invaluable, and to her I devote my utmost admiration and respect.

This work was supported in part by a grant from the U. S. Department of Energy.

for my mother,

in loving memory of my father

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

I. INTRODUCTION AND OVERVIEW

In the last fifty years, there has been growing experimental evidence which indicates the approximate charge independence of the nuclear force (also referred to as the strong or hadronic force). This evidence is, for example, provided by scattering lengths[1] for the two-nucleon system and the similarity of the level structures of isobaric multiplets. The latter is evidenced by measurements of Coulomb energy displacements of nuclear levels and the general success of the isobaric multiplet mass equation [Jän69]. The results of these experiments suggest that, except for electromagnetic effects, the forces between protons and/or neutrons are nearly equal (sometimes given schematically as: pp = nn = pn). The equality of these forces is one definition of *charge independence*. A weaker condition, known as *charge symmetry*, requires only that the proton-proton and neutron-neutron forces are equal (pp = nn) and is thus a necessary but not sufficient condition for charge independence.

Experiments indicate charge independence of the strong force holds to ~1 percent. However, at this level there are still some discrepancies which cannot be accounted for by electromagnetic and other known charge dependent effects and thus may point to a slight charge dependence of the nuclear interaction. Some of the most precise experiments used to explore this possibility involve the study of resonance states whose formation and decay are forbidden for completely charge independent interactions. The present work provides measurements for three such resonances in ^{24}Mg, ^{28}Si, and ^{32}S and should furnish insight into the charge dependent processes in nuclear systems.

---

[1] The scattering length is a parameter used, along with the effective range, to describe low energy scattering cross sections in a manner which is approximately independent of the details of the nuclear potential.

1.1 Organization of Material

Proper presentation of the data and results requires that background material and preparatory work be described first. The remaining sections of this chapter consist of some of the definitions and information necessary for the understanding of the requirements and problems encountered in this type of experiment. Also presented are a brief review of similar previous experiments and a summary of the data as it existed prior to this work.

Subsequent chapters will deal with the various experimental and theoretical aspects of obtaining and analyzing the data to produce results suitable for interpretation. The general organization of the remaining material is as follows:

Chapter 2: This chapter describes the experiment itself -- the hardware and the methods and procedures used to overcome many of the difficulties involved in obtaining the data.

Chapter 3: Here we examine the very important problem of experimental energy resolution. We examine the origins of a non-zero spread of beam energies and our attempts to measure and/or estimate the resulting beam profile.

Chapter 4: Although in some sense separated from the nuclear physics, target making is a very important and non-trivial part of the experiment. As such, the trials and tribulations of the production of thin solid targets deserve and are accorded a separate chapter.

Chapter 5: Here, we explain the methods of the data analysis. The theory of resonance reactions is reviewed and the details and approximations which comprise the analysis programs are discussed.

Chapter 6: Included here are the specifics for each experiment. The experimental setups are described, the unique features of the data are outlined, and some data and intermediate results pertaining to the $T = 2$ states are presented.

Chapter 7: Finally, we offer the final outcome of the fitting procedures with best fit values and errors for the resonance parameters. Also, we give conclusions, a compilation of results involving alpha decays from $T = 2$ levels, and suggestions for improvements and further study.

1.2 Isospin

Although it would probably be neither possible nor desirable to define all concepts and ideas from the outset, it may be helpful to provide some discussion of terms and concepts which are essential to the understanding of the work as a whole. We begin with a discussion of *isospin*, a concept which is intimately related to the idea of charge independence. When the notion of isospin was first introduced, it was, at best, a mathematical device useful only for labelling protons and neutrons. However, since its beginnings, the concept of isospin has been expanded upon and applied to many different areas of physics. It has been invoked in a variety of contexts, often referred to, apparently understood, and only occasionally defined. That some confusion has resulted is evidenced by the name itself; first "isotopic spin," then "isobaric spin," and finally a compromising "isospin." Although the concept is important in many areas of physics (applied to pion system it predicted the existence of the neutral pion), here we shall only be concerned with isospin as it relates to low energy nuclear physics and more specifically to resonance reactions and particle decays. For this purpose the most

convenient and probably the most satisfying "definition" of isospin is an outline of its historical development. (For a complete discussion see [Wil69a].)

In 1932, after the discovery of the neutron, Heisenberg suggested that perhaps the proton and neutron may be regarded as alternate states of the same particle -- the nucleon. This would certainly work mathematically, since the formalism for such a two level system had already been worked out by Pauli for spin-½ particles. The variable needed to describe this nucleon system is the isospin, usually denoted by the letter "t."

Discussions of isospin formalism can be approached on many different levels, some requiring an extensive knowledge of quantum mechanics and others being couched in permutation symmetries, group theory, and representations of SU(2) [Ell69]. Trying to provide some understanding of a somewhat enigmatic concept, we will present a brief synopsis of the simplest aspects of the isospin formalism, in all cases emphasizing its analogy with ordinary spin. The more ambitious reader is referred to the collection of various articles in [Wil69].

As noted, the mathematical formalism for isospin is identical to that for spin, except that the isospin operators do not act in real space but in isospin or charge space (we will see the relationship with charge in the next section). The isospin operator $\mathbf{t}$ can be thought of as the set of operators $\{t_1, t_2, t_3\}$ analogous to the Pauli spin operators usually represented as 2×2 matrices. Operating on states of good isospin (e.g. eigenstates of $t^2 = t_1^2 + t_2^2 + t_3^2$ and t_3), the operators t^2 and t_3 produce eigenvalues related to the "quantum numbers" t' and t_3'. That is,

$$t^2 | t', t_3' \rangle = t'(t'+1) | t', t_3' \rangle$$

and

$$t_3 | t', t_3' \rangle = t_3' | t', t_3' \rangle \qquad [1.1]$$

Also, we can define isospin raising and lowering operators which do not in fact change t' but the third component t_3'. This raising and lowering of t_3' is like changing protons to

neutrons and vice versa. In practice, it is often easier to think of the isospin **t** as a vector in charge space (it transforms as such) with magnitude t and projection t_3 (note we have dropped the primed notation). Furthermore, it is common to think in terms of rectangular coordinates with the z-axis as the quantization axis and thus t_3 becomes t_z. Also, we note that there is often a somewhat cavalier attitude about referring to "isospin" as an operator, variable, eigenvalue, vector, or quantum number. Indeed, if one is aware of this propensity, the meaning is most often made clear by the context. For convenience, we will also adopt this practice; however, we will use boldface type when we wish to emphasize the vector or operator nature of the isospin.

Again, like spin, there are $2t+1$ possible projections for a particle with isospin t, and the nucleon has isospin $\frac{1}{2}$ since it has only two possible states -- the proton and the neutron. In low energy nuclear physics the proton is generally assigned to the negative projection or $t_z = -\frac{1}{2}$ and the neutron is given $t_z = +\frac{1}{2}$. Note that this choice is somewhat arbitrary[2]; indeed the convention is reversed in particle physics.

We can begin to see the relationship between isospin and charge by noting that the charge operator, Q, can be written in terms of the third component of isospin:

$$Q = e\left(\frac{1}{2} - t_z\right) \qquad [1.2]$$

where e is the elementary (electron) charge. So, it is clear that charge conservation implies conservation of the third component of isospin, and thus t_z is always a "good" quantum number. Quantum mechanically, this symmetry results from an invariance property of the Hamiltonian of the system and is expressed by saying that the Hamiltonian commutes with the charge or t_z operator:

$$[\,Q\,,H\,] = [\,t_z\,,H\,] = 0 \qquad [1.3]$$

---

[2] Most stable nuclei have $N > Z$. Thus, sign errors may be minimized because the third component of nuclear isospin, $(N-Z)/2$, is usually positive.

Similarly, complete charge independence implies a rotational invariance[3] of the Hamiltonian in isospin space, entails isospin conservation,[4] and requires the Hamiltonian to commute with the isospin operator:

$$[\,H\,,\,\mathbf{t}\,]=0 \quad \text{or} \quad [\,H\,,\,t^2\,]=0\,. \tag{1.4}$$

Note that since the total Hamiltonian is almost never completely charge independent, isospin is, in general, not a good quantum number. However, the usual expression of charge independence refers to only that part of the Hamiltonian which governs the nuclear or hadronic interaction. In this case, the commutation relations are:

$$[\,H_{nuc}\,,\,\mathbf{t}\,]=0 \quad \text{or} \quad [\,H_{nuc}\,,\,t^2\,]=0\,, \tag{1.5}$$

and in this context isospin is very nearly conserved.

Often, there is a distinction made between *direct* and *indirect* charge dependent effects. Direct effects are those which act irrespective of the nuclear force. That is, even if the nuclear force were not present, charge independence would still be broken by these direct processes. Examples of direct effects include the Coulomb force, magnetic interactions, and the proton-neutron mass difference. On the other hand, indirect effects are those which act through the nuclear force, altering it in such a way as to violate charge independence. Thus, indirect effects are absent without the hadronic interaction, and examples include pion mass differences, radiative corrections to meson-nucleon coupling constants, and quark degrees of freedom. It follows that another way to view the usual expression of charge independence is to consider only forces in the nucleus after corrections have been made for direct charge dependent effects.

---

[3] While charge independence requires the Hamiltonian to be completely rotationally invariant, the less demanding charge symmetry represents invariance under a 180° rotation (reflection) about the x or y axis.

[4] A rigorous proof that isospin conservation implies charge independence (and vice versa) may be found in [Wic58].

1.3 Isospin in Nuclei

The concept of isospin can be extended to nuclei by defining

$$\mathbf{T} = \sum_{i=1}^{A} \mathbf{t}^{(i)}$$

and

$$T_z = \sum_{i=1}^{A} t_z^{(i)} \qquad [1.6]$$

where we have used **t** for nucleon isospin and **T** for nuclear isospin and *i* labels the individual nucleons. Like spin, allowed values of T_z are from $-T$ to $+T$, but for a particular nucleus we see that T_z is simply the algebraic sum of the nucleon t_z's and as such is always given by $\frac{1}{2}(N - Z)$. However, **T**, being the vector sum of the constituent nucleon isospins, does not have such a simple algebraic formula. For "real-life" nuclei, considerations such as the Pauli exclusion principle and the related requirement of an antisymmetric wave function play an important part in the determination of the isospin of the nuclear levels. However, in general, nuclear forces favor the lowest possible isospin consistent with $T_z = \frac{1}{2}(N - Z)$ [Sop69]. Thus, for all but the most troublesome nuclei (a few odd-odd nuclei such as ^{34}Cl and ^{42}Sc), the ground state isospin is equal in magnitude to $|T_z|$, although excited states can and do have $T > |T_z|$. Levels in nuclei with isospin equal to the ground state isospin are sometimes referred to as T-lower ($T_<$) states, while those with $T > |T_z|$ are labelled T-upper or T-higher ($T_>$). For example, we will be dealing with $T = 2$ states ($T_>$) in nuclei with ground states of $T = 0$ ($T_<$).

Analogous to spin, there should be five ($2T + 1$) different projections ($T_z = -2, -1, 0, 1, 2$), corresponding to five different levels, associated with each $T = 2$ state. This is indeed the case, and figure [1.1] illustrates the system for the lowest lying $T = 2$ states in

^{20}Ne. Note that the T_z value, being $\frac{1}{2}(N-Z)$, distinguishes the particular nucleus in the group. These states of the same isospin but different projections form what is known as an *isobaric multiplet* or *T-multiplet*, and the states themselves are *isobaric analog states*.[5] Within the multiplet, the nucleus with the highest value of T_z ($T_z = +|T|$) is sometimes referred to as the *parent* and its ground state as the *parent state* of the multiplet. The multiplet may be generated from the parent state by repeated action of the isospin lowering operator, and one can think of the states of the multiplet as being linked through the proper application of the isospin raising and lowering operators. In addition, nuclei with the same mass number whose T_z values are equal in magnitude (but opposite signs) are sometimes referred to as *mirror nuclei* (e.g. ^{20}F and ^{20}Na or ^{20}O and ^{20}Mg in figure [1.1]) since, for strict isospin conservation, their energy level spacing would be identical. Also, in discussions of isospin mixing one often encounters references to *anti-analog states*. These are simply states of the same T_z (same nucleus) which have identical space-spin wave functions but have lower total isospin.

Beyond the formal definitions given above, charge independence (implying strict isospin conservation) in nuclei has some interesting physical consequences.

1) First, there would be exact energy degeneracy for members of a multiplet. Figure [1.1] shows only an approximate matching since not all charge dependencies are accounted for.

2) In addition, the states of the multiplet would have an exact correspondence in space-spin wave functions. Thus, mirror nuclei, as defined above, would be identical except for the signs of their projections of isospin.

3) Finally, there would be complete purity of isospin for all nuclear stationary

---

[5] The adjective "isobaric" simply indicates that the nuclei have approximately the same atomic mass and is often omitted when the meaning is clear.

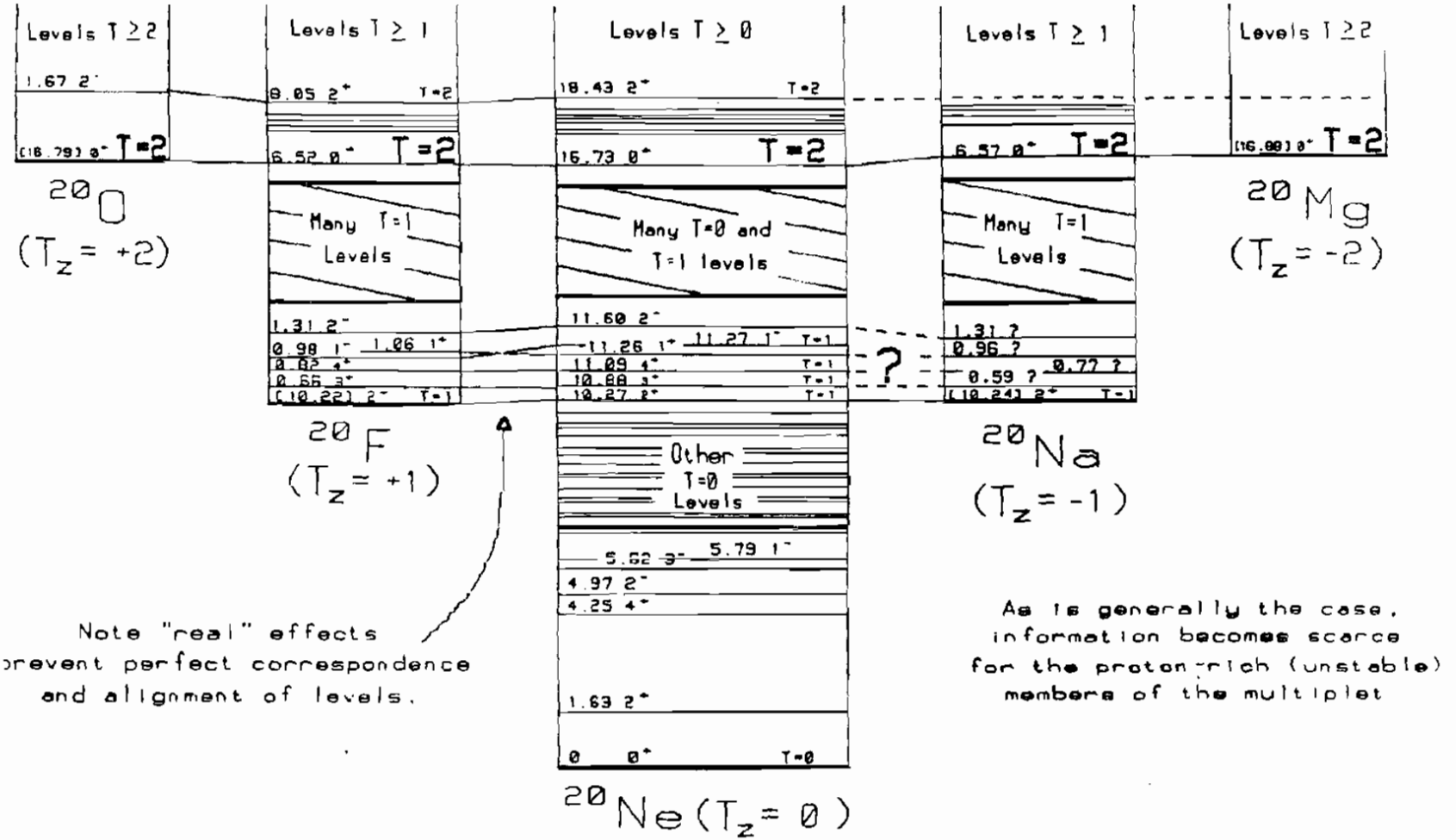


Figure [1.1]: Simplified isobar diagram for A=20. The vertical scale is shifted to account for proton-neutron mass difference and Coulomb energy shift ($E_c = 0.6Z(Z-1)/A^{\frac{1}{3}}$). Energies in brackets are approximate - calculated from $E_N = M(Z,A) - AM(H) - NM(n) - E_c$ minus E_N for ^{20}Ne, where M is atomic mass excess in MeV. Solid lines connect states which are known analogs. Dotted lines represent probable analog states & levels whose assignments are not verified. From [Ajz87].

states. This implies no mixing of states, and thus no particle decays that do not conserve isospin.

In this context, "mixing of states" results in a nuclear level which is actually a linear combination of wave functions of different isospins. Isospin mixing is the result of charge dependent forces in the nucleus, and the amount of mixing is a measure of the extent of the charge dependence. We note that for the isospin-forbidden resonance reactions of interest, the mixing may occur in the target, compound system, projectile, or some combination of the three. See [Wil82] for a complete discussion of possible mixing mechanisms. In any case, the presence of these decays and the lifetimes of the $T_>$ states can be a measure of the mixing and thus of charge dependence in the nuclear system.

Of course we know that some of the forces acting in the nucleus are charge dependent and the main problem in interpreting any of these measurements is the presence of electromagnetic interactions. The major electromagnetic effect is the static Coulomb force which is highly charge dependent and as such may violate isospin conservation. Furthermore, it is always present in the nucleus; that is, we cannot experimentally "turn off" the charge on the proton and must therefore learn to work around it in the analysis and interpretation of the data. Fortunately, the Coulomb interaction is weak compared to the strong force, is relatively well understood, and can perhaps be subtracted out theoretically. The results of this type of analysis, with proper accounting of electromagnetic effects, may then provide information about the possible charge dependence of the nuclear or strong force.

1.4 Isospin-Forbidden Resonance Reactions

In these types of resonance measurements, one observes, in an energy excitation function, a resonant effect corresponding to a state whose formation and decay are

forbidden by isospin conservation. A resonance energy *total width*, Γ, is extracted by fitting the data to a Lorentzian (Breit-Wigner) line shape. The width, related to the lifetime, τ, by the uncertainty principle

$$\Delta\Gamma\,\Delta\tau \sim \hbar, \qquad [1.7]$$

is a measure of the isospin mixing in the system. That is, for strict isospin conservation, the state would not be mixed, there would be no decay (an infinite lifetime), and the level width would be zero. Any amount of mixing produces a non-zero width, and, since isospin is approximately conserved for the strong interaction, we expect the level widths to be much smaller than those normally associated with particle decays. Indeed, the isospin forbidden resonances which have been observed are found to have total widths of ≤ 1 keV [Wil82].

Just as the total width can be thought of as an indicator of the probability of decay, we can define a *partial width*, Γ_i, which is a measure of the probability of decay into a particular channel. Alternately, we can view the partial width as proportional to the square of the transition amplitude for a particular decay (or formation) reaction. Generally, the sum of the partial widths is equal to the total width, and thus $\Gamma_i < \Gamma$. Since we are interested in particular particle decays of the $T_>$ states, determination of the partial width is really the main goal of the analysis. However, before we can interpret the results, certain kinematic factors must be removed from the extracted value of the partial width.

Particles entering and exiting a nucleus have to overcome angular momentum[6] (for $\ell > 0$) and Coulomb (for charged particles) barriers. Although our particle energies

---

[6] When solving the Schrödinger equation in three dimensions, one obtains, from separation of variables, a term in the radial equation of the form $\ell(\ell+1)\hbar^2/2mr^2$. This term has the dimensions of energy and suggests the form of a classical centrifugal potential, $L^2/2mr^2$ where L is the orbital momentum of a particle with mass m moving in a radius r. We think of this new term as added to the nuclear potential and forming a centrifugal barrier for particles with $\ell > 0$ (classically, for collisions of noncentral nature).

are high enough to surmount these barriers, the probabilities for formation and decay and hence the widths of the nuclear states in question are nonetheless affected. Luckily, these effects are well understood and can be relatively easily calculated for any particular decay and divided out of the partial widths. The TUNL library program PENET calculates these *Coulomb penetrabilities*, P_ℓ, including angular momentum effects based on the decay products and the energies and ℓ-values of the nuclear reactions. The result of this correction is the *reduced partial width*, γ_i^2, (sometimes simply called "reduced width") given by:

$$\gamma_i^2 = \frac{\Gamma_i}{2P_\ell}. \qquad [1.8]$$

The reduced width is more characteristic of the nuclear physics involved in the decay and thus more suitable for comparison with the results from other nuclei and/or reactions. The square in the symbol emphasizes its role as proportional to a probability or square of a transition amplitude.[7]

Finally, to produce a dimensionless quantity, the reduced width is divided by the *single particle reduced width*. This is simply a calculation of the width based on a model in which the resonance can be described in terms of the motion of a single particle in the nuclear potential well. A common single-particle estimate is given by:

$$\gamma_{sp}^2 = \frac{\hbar^2}{ma^2} \qquad [1.9]$$

where m is particle mass and a is the radius of the well. Sometimes one encounters ratios where the single particle widths are calculated by a simple model suggested by Wigner. This *Wigner limit width*, $\gamma_{wl}{}^2$, differs by constant factor from more realistic single particle estimates and is usually an overestimate. The final ratio, $\gamma_i^2/\gamma_{sp}{}^2$, is sometimes called the

[7] One can also view $\gamma_i{}^2$ as a measure of the overlap of the $T_>$ state and the final target state wave functions.

single particle fraction and may be viewed as the probability of the resonant state being in the state described by the model, somewhat like a spectroscopic factor.

Having presented an overview of the general features of isospin forbidden resonance reactions, we proceed with a more detailed look at specific studies of this type. However, before we describe the data from the present study, it will be useful to review some previous related experiments which made this work possible.

1.5 Previous Work

Previous measurements of the type described above have yielded some very precise data for isospin-forbidden particle decays. In particular, Wilkerson [Wil82], building on the work of Ikossi [Iko75], and more recently Mooney [Moo89][8], have studied $T = 3/2$ states in light $T_z = -1/2$ nuclei via unpolarized and polarized proton elastic scattering. Wilkerson has extracted proton partial widths for the lowest $T = 3/2$ states in ^{13}N, ^{17}F, ^{25}Al, ^{29}P, ^{33}Cl, and ^{37}K and has also examined several higher lying $T = 3/2$ states in some of these nuclei. We should point out that there also exist less precise measurements for neutron decays in some of the mirror nuclei [Hin81] [Jun82]. By virtue of its being one of the most recent and comprehensive studies, we shall mainly refer to the work of Wilkerson. However, we certainly are not intending to disregard or lessen the importance of the large amount of work which provided the foundations for this type of investigation, and we refer the reader to [Wil82] for a complete list of references.

By studying the proton decays from $T = 3/2$ to $T = 0$ states, one is dealing with

---

[8] Studying the lowest-lying $T = 3/2$ states in ^{21}Na and performed at TUNL, this experiment used procedures and an experimental setup almost identical to those of the present work. It was intended that the two theses be somewhat complementary (and not too redundant) concerning descriptions of experimental details. Also, the Mooney thesis provides some interesting alternate views of isospin mixing and places more emphasis on shell model considerations.

interactions in which the magnitude of the total isospin may change by one or two units. That is, the proton itself has isospin ½, and we have, for example:

$$^{21}\mathrm{Na}\left(T=\tfrac{3}{2}\right) \longrightarrow^{\Delta T}\rightarrow \ ^{20}\mathrm{Ne}\,(T=0) + \mathrm{p}\left(T=\tfrac{1}{2}\right) \ . \qquad [1.10]$$

Since the isospins add vectorially, we can have $\Delta T = 1$ or 2. The components of the interaction which can violate isospin conservation are referred to as the *isovector* and *isotensor* components, so named because of their transformation properties in isospin space. An isovector interaction can change the isospin by zero or one unit ($\Delta T = 0, \pm 1$), while an isotensor component can alter the isospin by zero, one, or two units ($\Delta T = 0, \pm 1, \pm 2$). Similarly, a purely charge-independent interaction would have only *isoscalar* ($\Delta T = 0$) terms. Furthermore, any charge-dependent two-body force can, in general, be written as the sum of isoscalar, isovector, and isotensor terms [Gar69]. In the language of operators, an isovector (isotensor) operator can "connect" states which differ in isospin by $\Delta T \leq 1$ ($\Delta T \leq 2$), and the matrix elements thus formed are designated isovector (isotensor) matrix elements.

We see that the proton decay of the $T = ½$ states as outlined above can be the result of isovector or isotensor interactions, and figure [1.2] summarizes results from proton scattering data, most of which are from [Wil82]. Some of the effects are rather surprising and have lead to some interesting conclusions and predictions. The general increase of the reduced widths with increasing mass is predicted by most of the isospin mixing theories and can naively be viewed as an increase in reaction probability because there are more protons (in higher energy states) available to the decay channel. However, most of these same theories predict the magnitudes of these reduced widths to be greater than the data by about a factor of four.

Perhaps the most striking and unexpected feature of the data is the $A = 8$ periodic oscillation of the points, wherein the level widths in the A=8n+1 compound nuclei seem

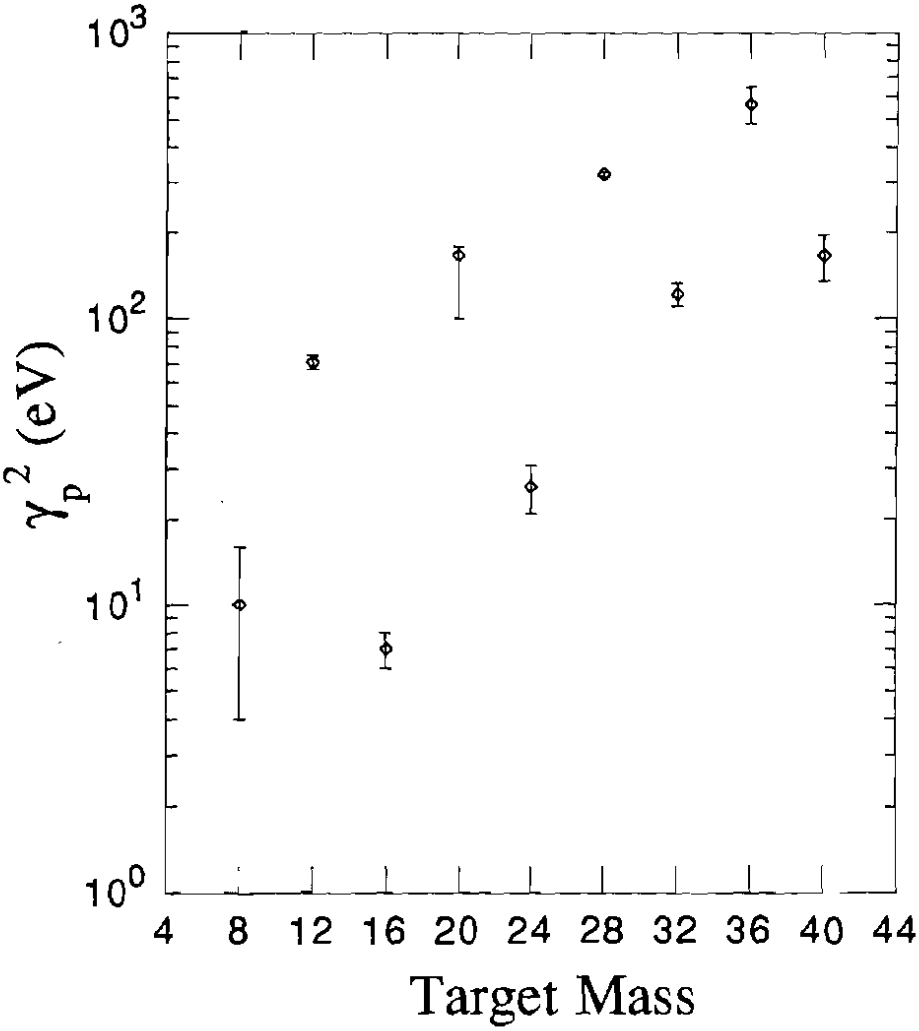


Figure [1.2]: Reduced proton widths for decays of the lowest-lying T = 3/2 states in some light nuclei (from [Wil82]).

to have a different linear relationship with mass than do the states in $A=8n+5$ nuclei (n = 1,2,3,...). Even recent state-of-the-art shell model calculations do not seem to reproduce this periodicity [Orm86]. Wilkerson *et al.* [Wil83] presented a schematic model based on delta resonances ($T = 3/2$ resonance states of the nucleons themselves) which produces the $A = 8$ periodicity as well as the approximate magnitude of the reduced widths. In addition, their analysis of existing isospin-forbidden p, n, and α decay data indicates comparable magnitudes for the isovector and isotensor terms of the interaction. It also suggests that the pure isotensor alpha decay widths will not exhibit the aforementioned $A = 8$ periodicity.

Whether or not the proton width oscillation stems from nuclear physics effects, it is likely that the difference in magnitudes from the theory to experiment is due to the presence of the isotensor term of the isospin mixing interaction. These isotensor mixing terms which, for protons, have the opposite sign to the isovector terms, have usually been neglected in the mixing theories ([Wil83] and the references therein). So, it can be concluded from the experimental data and analyses at hand that isotensor terms are not at all negligible; indeed it can be inferred that the isotensor matrix elements are of the same order of magnitude as the isovector terms.

As previously noted, data from proton (or neutron) decay of $T = 3/2$ states to $T = 0$ states are sensitive to both the isovector ($\Delta T \leq 1$) and isotensor ($\Delta T \leq 2$) components of the interaction, and as such one can only infer their relative strengths. In order to learn more about both terms, it is desirable to study decays which are sensitive to only one of these two terms. Since both terms can produce $\Delta T = 1$ transitions, we must look toward $\Delta T = 2$ reactions which involve only isotensor components. Perhaps the most obvious way to accomplish this is by examining alpha decays of $T = 2$ to $T = 0$ states. Since the alpha particle nominally has isospin $T = 0$, this decay would exhibit pure isotensor character. However, another way to exclude isovector contributions would be to examine

particle decays from $T = 5/2$ states. This includes alpha decays to $T = 1/2$ states or proton decays to $T = 0$ ground states using the same nuclei as for the $T = 3/2$ study. As a means of comparing the merits of the two systems, we will examine eaeh, beginning with the decays from $T = 5/2$ states.

1.6 $T = 5/2$ Feasibility Study

It is possible to isolate the isotensor mixing elements by using elastic scattering on $T = 0$ targets to examine isospin forbidden particle decays of states with $T > 2$. For example, as mentioned previously, proton decays from $T = 5/2$ to $T = 0$ ground states or alpha decays from $T = 5/2$ to $T = 1/2$ levels should be sensitive to only the isotensor part of the interaction. The two situations are shown schematically in figures [1.3a] & [1.3b]. The proton decays involve the same $A = 4n+1$ compound systems that were studied in $T = 3/2$ measurements [Wil82], while the alpha scattering experiment would make use of stable, $T_z = +1/2$ targets and thus populate states in $A = 4n+3$ nuclei. For either the alpha or proton scattering data, the energies, spins, and parities of the $T = 5/2$ states in question are mostly unknown. Also, for the alpha decay data, the non-zero target spin greatly complicates the analysis. Since the study of these alpha decays from $T = 5/2$ states yields no more information about isospin mixing than does the study of the alpha decays from $T = 2$ states, this type of experiment was not considered further.

On the other hand, the examination of proton decays from $T = 5/2$ states offers at least one advantage over alpha scattering data -- the availability of analyzing powers. However, acquisition of the data is still complicated by the ignorance of the energies, spins, and parities for the $T = 5/2$ states. Also, perhaps the biggest problem arises from the availability of energetically allowed, isospin allowed, proton decays to $T = 2$ states in the target nucleus. With an allowed particle decay channel open, it might seem that the

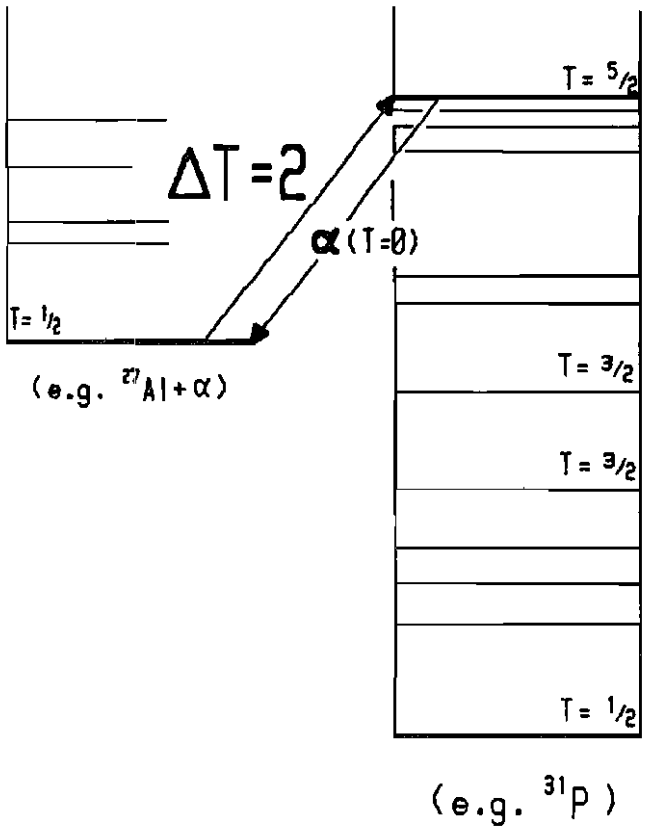


Figure [1.3]a : Pure ΔT=2 decays can be achieved via elastic alpha scattering through $T=\frac{5}{2}$ states.

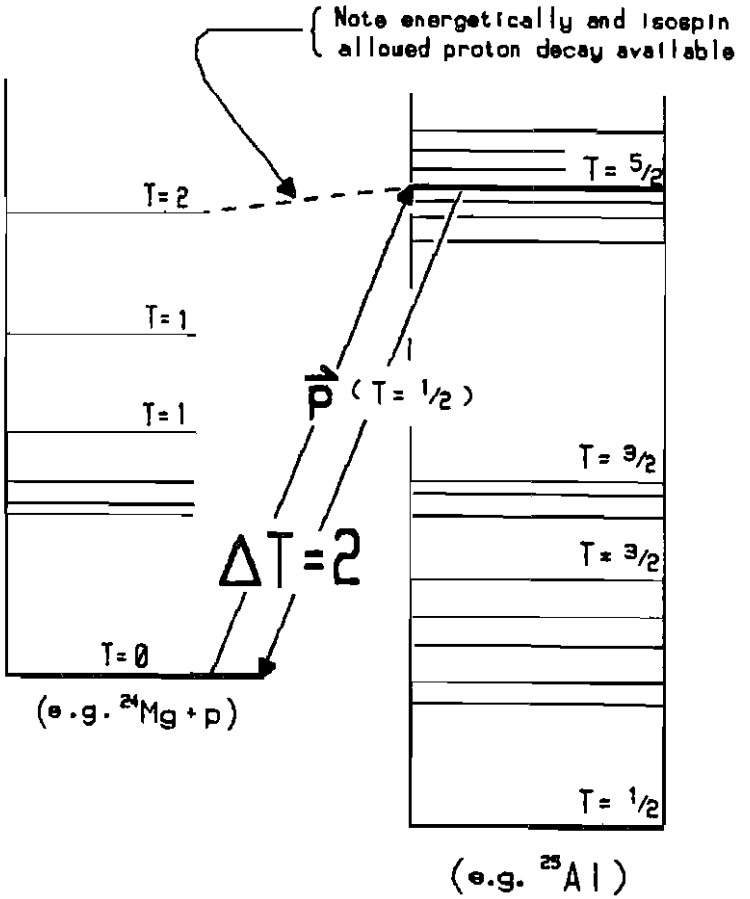


Figure [1.3]b : Proton scattering from $T=\frac{5}{2}$ states allows studies of ΔT=2 with polarized beams

probability for the isospin forbidden decay would be negligibly small. However, if the energy difference between the $T = 5/2$ state in the compound system and the $T = 2$ level in the target is small, then the proton decay may be suppressed because of angular momentum and Coulomb barrier penetrability considerations. Since the energy locations of the $T = 2$ states are fairly well established, we can calculate the penetrabilities, P_ℓ, for decays to these levels as well as decays to the ground state if we can estimate the energy positions of the $T = 5/2$ states. Following is a rough estimation of the requirements for the ratio of the penetrabilities of the two decays in order for the ground state decay cross section to be greater than or equal to that of the $T = 2$ decay.

$$\frac{\sigma(p,p')}{\sigma(p,p_0)} \sim \frac{P_\ell(E_{p'})}{P_\ell(E_p)} \frac{\gamma_{p'}^2}{\gamma_p^2}$$

$$\sigma(p,p_0) \gtrsim \sigma(p,p') \Rightarrow \frac{P_\ell(E_{p'})}{P_\ell(E_p)} \lesssim \frac{\gamma_p^2}{\gamma_{p'}^2} \sim 10^{-3} \qquad [1.11]$$

where primed and unprimed quantities represent isospin allowed and isospin forbidden transitions, respectively.

Since we know from previous work [Wil82] that the ratio of the reduced widths for isospin allowed to isospin forbidden decays is on the order of 10^3, we see that a ratio of penetrabilities of 10^{-3} or less might make the measurement possible. We note again that some knowledge of the energies, spins, and parities of the $T = 5/2$ states is essential for any kind of estimation of this type. The large uncertainty in the energy location of these states also indicates some severe experimental difficulties. Since the resonance widths are expected to be very small (< 1 keV), excitation functions must be acquired using small energy steps (usually hundreds of eV). Combining this fact with the need for thin targets and the low cross sections (tens of millibarns) which are common for these

reactions,[9] we see that searching a range of hundreds of keV (a thousand or so steps) would be very time consuming and probably not practical for this type of experiment.

One means of estimating level spacings is by using the isobaric multiplet mass equation (IMME), but it requires the knowledge of the masses (energies) of three members of the isobaric multiplet.[10] For the $T = 5/2$ states in question, most often information is only available for the neutron-rich, $T_z = +5/2$ member of the multiplet. Alternately, we can use estimates of the Coulomb energy difference (ΔE_c) from the parent ($T_z = +5/2$ member) to the $T = 5/2$ state in question and the ground state mass differences to predict the location (E_x) of the $T = 5/2$ levels. One established way of obtaining Coulomb energy differences is based on shell model considerations [Com83]. Basically, the method consists of using available data and finding a best fit, subject to shell-model restraints, to coefficients related to those in the IMME. Here, ΔE_c represents only Coulomb, or perhaps Coulomb-like, forces and does not include proton-neutron mass differences or the ground state mass excesses of the parent and final nucleus. So, in order to predict the energy location of a state in the final nucleus (E_x), one needs:

$$E_x = \Delta E_c - k m_{n-H} + (\Delta_{parent} - \Delta_{final}) \qquad [1.12]$$

where k is the number of neutrons that were "changed" to protons in going from the parent to the final state nucleus, Δ is the atomic mass excess, and m_{n-H} is the mass difference between the neutron and the hydrogen atom.[11] Using a simple computer program and mass excess values from the 1983 Mass Evaluation [Wap85], we obtain the

---

[9] The resonance signatures are most obvious at very backward scattering angles.

[10] In its usual form, derived from first order perturbation theory and assuming only two-body interactions:

$$M(A,T,T_z) = a(A,T) + b(A,T)T_z + c(A,T)T_z^2$$

where M is the mass and a, b, and c are three unknown coefficients.

[11] We use Hydrogen mass since the tables of mass excesses are atomic masses.

following table [1.1]. Note that we have included the previously discussed penetrability calculations from the TUNL library program PENET and the locations of the neighboring $T = 2$ states which provide energy and isospin allowed decay channels. Also, note that only the effects mentioned above are included in the estimates, and there are certainly other effects such as the Thomas-Ehrman shift [Ehr51] which are not considered.

Compound system	Predicted E_x (MeV)	Probable J	T=2 state E_x (MeV)	P_ℓ'/P_ℓ ($\times 10^{-3}$)
^{17}F	26.15	?	23.32	*
^{21}Na	19.71	?	19.16	*
^{25}Al	18.32	$(\frac{1}{2})^+$	17.70	0.1
^{29}P	19.83	$(\frac{3}{2})^+$	17.97	0.8
^{33}Cl	15.99	$(\frac{3}{2})^+$	14.33	0.3
^{37}K	15.33	$(\frac{1}{2})^-$	12.72	0.4
^{41}Sc	14.80	$\frac{1}{2}^-$	13.07	0.01

* need to know J values

Table [1.1]: Predicted energies and probable J^π values for some $T = \frac{5}{2}$ states. Also tabulated are locations of the nearby $T = 2$ states and the ratio of penetrabilities discussed in section 1.6. Values of E_x are given relative to the ground state of the nucleus in column one.

Errors in the predicted values of E_x for the $T = \frac{5}{2}$ states can be estimated by using the same procedure to estimate the energy positions of the $T = \frac{3}{2}$ states and then comparing these estimates to the known experimental values. Some of these results are shown in table [1.2] and indicate errors from 20 to >400 keV with the average being about 60 keV (excluding the value for ^{41}Sc which is probably way off because of shell closure effects).

Compound System	Predicted E_x (MeV)	Actual E_x (MeV)	Error (keV)
^{21}Na	8.95	8.97	20
^{25}Al	7.78	7.90	120
^{29}P	8.30	8.38	80
^{33}Cl	5.51	5.54	30
^{37}K	5.12	5.05	70
^{41}Sc	6.35	5.94	410

Table [1.2]: Estimates of errors in theory from predicted and known values of some $T = 3/2$ states.

From the data at hand and the predictions shown, we conclude that measurements of proton elastic scattering from $T = 5/2$ to $T = 0$ states, while perhaps not impossible, would almost certainly be extremely difficult, and the information obtained would be much the same (except for the additional information afforded by the use of polarized beams) as that from alpha decays of $T = 2$ states. If the locations of the $T = 5/2$ states in question or other members of the $T = 5/2$ multiplet could be more accurately determined, the measurements of these states through isospin-forbidden resonance reactions would become an interesting experimental challenge.

1.7 $T = 2$ States

Having emerged from a slight digression into the feasibility of studying decays of states with $T > 2$, we are now better equipped to understand the attractiveness of studying alpha decays from $T = 2$ states. We see that many of the disadvantages involving $T = 5/2$

states do not exist in the study of alpha decays from $T = 2$ levels in self-conjugate ($Z = N$) light nuclei, and indeed there are several distinct advantages to this latter scheme.

1) First, there are no open isospin-allowed particle decays. The only open allowed channels are electromagnetic, and, since these decays are governed by the weak and electromagnetic interactions, they are accordingly suppressed compared to the strongly interacting particle decays. There are, however, competing proton decays and alpha particle decays (also isospin forbidden) to excited states in the final nucleus. As we shall see, the branching ratios for some of these decays are substantial, in some instances being much greater than the alpha decay to the ground states. Note also that the neutron channel is energetically not available to most of the $T = 2$ states in the nuclei in question.

2) Secondly, the spins and parities of the $T = 2$ states are known to be $J^{\pi} = 0^{+}$ since they are analog states of even-even ($|T_z| = 2$) nuclear ground states and as such must have predominately the same space-spin wave function. This greatly simplifies the mathematics involved in the resonance analysis, even though some of the targets may have non-zero spin.

3) Thirdly, the branching ratios for the decays of the $T = 2$ states have been measured by populating the states with the isospin allowed (p,t) reaction and observing coincidences between the tritons and the $T = 2$ state decay products [McG70][Fre79]. This is a very important factor in the analysis of the data, since it represents one of the three unknown parameters of the Breit-Wigner resonance form.

4) Lastly, from previous works the energies of the states are known to within a few keV. This aspect is probably the most important factor experimentally, since only a relatively small energy range needs to be

searched in order to locate the state via resonance reactions.

A schematic of the system for alpha elastic scattering is give in figure [1.4a].

A summary of information about the $T = 2$ states in several light nuclei is given in table [1.3]. For the lowest $T = 2$ states of nuclei with $16 < A \leq 40$ (the 2s-1d or simply s–d shell), the required input projectile energies are well within the range of the TUNL accelerator, and the energies are low enough to provide a small beam energy spread in the TUNL high resolution system. Also, we see from the table that certain of the nuclei would be almost impossible to study by virtue of their open neutron channels and/or very small branching ratios. Other nuclei, ^{40}Ca in particular, although possessing many desirable features, would require expensive and hard-to-make targets (in this case ^{36}Ar gas-jet or ion-implanted). Furthermore, we note for some target nuclei, e.g. ^{24}Mg, the small alpha branching ratio renders impractical the observation of the $T = 2$ states via alpha elastic scattering. However, since we are really only interested in the alpha decays from these $T = 2$ states, the states may be populated by incoming protons thus using the (p,α) reaction to extract the alpha partial widths. We shall see later that using the (p,α) reaction introduces some complications into the formulae for the scattering amplitudes and the proton partial widths also need to be determined from elastic proton scattering. An analysis along these lines has been ventured by [McD78] with some success, and a schematic of the level scheme for ^{32}S is shown in figure [1.4b].

In the present work, we have undertaken the study of the lowest $T = 2$ states in ^{24}Mg, ^{28}Si, and ^{32}S. By virtue of their small alpha branches, the data for the ^{24}Mg and ^{32}S states are obtained via the (p,α) reaction as described above. Also, since the $T = 2$ state in ^{28}Si has been previously studied via high resolution alpha elastic scattering [Iko79], it should provide us with a valuable check for accuracy and consistency. As they existed prior to our study, reduced alpha widths of the lowest $T = 2$ states in some light nuclei are given in figure [1.5]. Note that values which consist of a single upper limit are

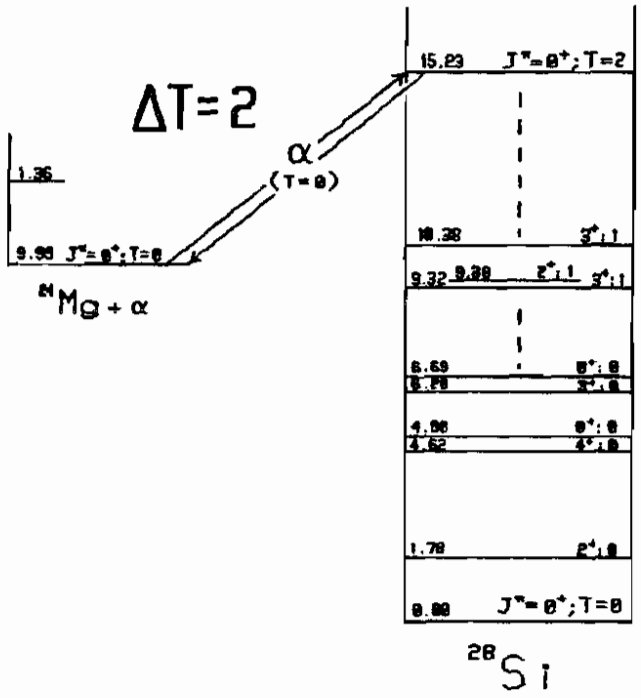


Figure [1.4]a : Example of decay scheme for alpha elastic scattering from the lowest T=2 state in ^{28}Si.

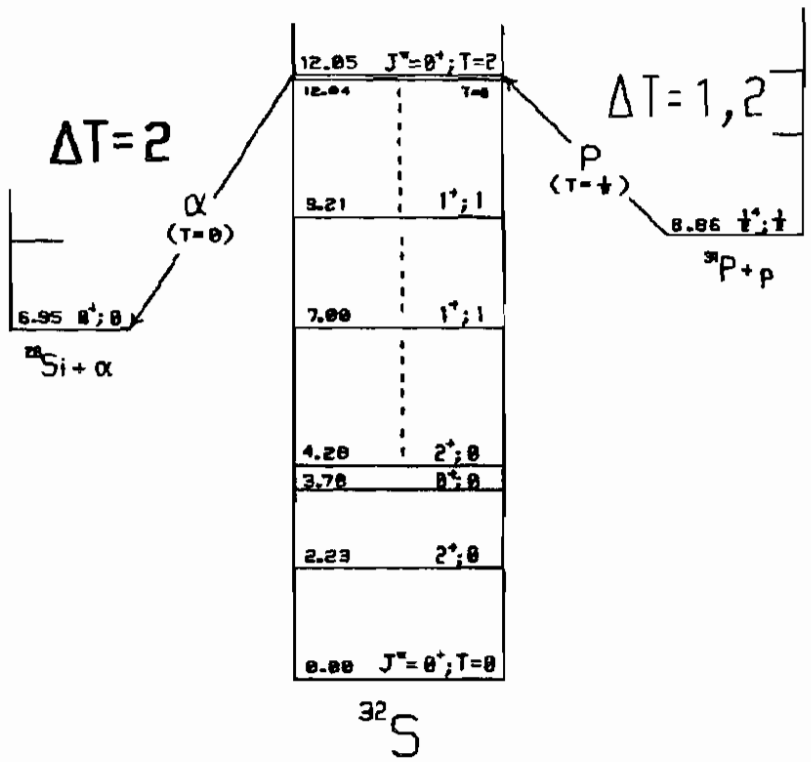


Figure [1.4]b : Sample system for the study of ΔT=2 decays via the (p,α) reaction.

Lowest T=2 States in some Light Nuclei

Compound System (shell)	E_x (MeV)	E_α (MeV)	E_p (MeV)	Γ (eV)	Γ_α/Γ (%)*	Γ_p/Γ (%)*	comments & refs.
^{8}Be (1p)	27.48	55.2 α+^{4}He	11.6 p+^{7}Li	5500 ±2000	0.2 ±0.4	0.6 ±0.6	n channel open [Fre79]
^{12}C (1p)	27.6	30.4 α+^{8}Be	12.7 p+^{11}B	‹30000	10.5 ±3.0	3.0 ±2.2	n channel open [Fre79][Rob78]
^{16}O (1p)	22.72	20.75 α+^{12}C	11.3 p+^{15}N	12000 ±3500	1.6 ±0.7	7. ±2.	n channel open [Fre79]
^{20}Ne (2s1d)	16.73	15.00 α+^{16}O	4.1 p+^{19}F	2100 ±500	–6 ±5	‹14 ±9	[McG70] large α branches to excited states
^{24}Mg (2s1d)	15.43	7.3 α+^{20}Ne	3.9 p+^{23}Na	435 ±135	6 ±1.3	49 ±6	[Fre79] [Osb78][McD78]
^{28}Si (2s1d)	15.22	6.1 α+^{24}Mg	3.8 p+^{27}Al	317 ±40	69 ±6	3.5 ±2	[Fre79] [Iko79]
^{32}S (2s1d)	12.05	5.83 α+^{28}Si	3.3 p+^{31}P	‹170	4 ±4	100 ±13	[Fre79] [Ver73]
^{36}Ar (2s1d)	10.86	4.75 α+^{32}S	2.4 p+^{35}Cl	‹35000	‹6	109 ±15	[Fre79]
^{40}Ca (2s1d)	11.98	5.49 α+^{36}Ar	3.7 p+^{39}K	‹1000	93 ±9	‹5	[Fre79] [Pri82]
^{44}Ti (1f2p)	9.34	4.64 α+^{40}Ca	0.71 p+^{43}Sc	1.10 ±.28	32 ±5	‹4	[Fre79] [Fre78]

* branching ratios are for decays to ground states

Table [1.3]: T = 2 states which are candidates for isospin forbidden resonance reactions study.

those which can only be estimated based on a relatively poor experimental energy resolution.

The small magnitude of the widths for these decays signals one of the main obstacles for acquisition of these data. When observing features of such narrow energy width, resolution effects become very important, since any experimental energy spread could easily "wash out" the effect in the excitation function. Obtaining the data requires special beam transport, target making, and data taking procedures designed to minimize the energy spreading mechanisms. Also, we recognize that it is not only important to minimize the energy spread but also to determine experimentally or calculate accurately the resolution effects. Efforts toward this end will be described in detail in the following chapters.

With the specialized hardware available to us and with our improvements in data taking procedures, we hope to expand the available knowledge of isotensor decays by improving the existing data for alpha decays from $T = 2$ states. More accurate and precise results for these pure isotensor decays should provide a better understanding of the extent of isospin mixing in light nuclei. Also, by choosing consecutive $A = 4n$ nuclei in the same major shell, we hope to obtain knowledge about possible systematics of the isospin mixing.

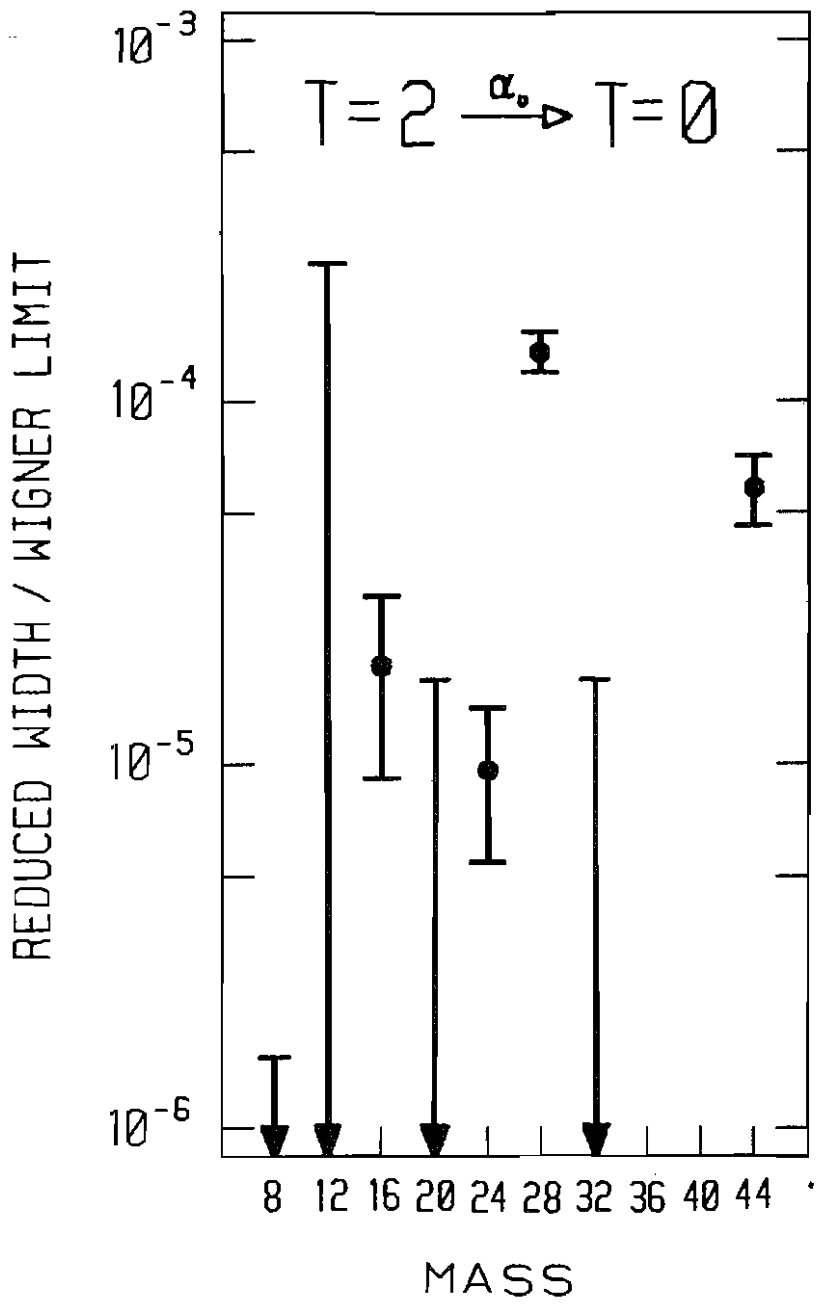


Figure [1.5]: Single particle fraction vs. compound system mass for decays from T=2 states to T=0 ground states (from [Fre79]).

II. DATA ACQUISITION WITH THE TUNL HIGH RESOLUTION SYSTEM

By virtue of the narrow energy widths for the states involved, obtaining isospin-forbidden resonance reaction data requires specialized hardware and procedures designed to minimize and determine the effects of various energy spreading mechanisms. This chapter examines the physical system and procedures used to obtain high energy resolution excitation function data. Also, it will outline the general methods of the offline data analysis and describe the programs used to calculate cross sections from the raw data.

Physically, the main ingredients of any scattering experiment of this type are ion source, accelerator, beam transport system, target, detectors, and the necessary signal processing hardware and software. Following is a description of the basic system as it exists at the Triangle Universities Nuclear Laboratory (TUNL) with some emphasis on the many improvements and additions made for use in this experiment over the last few years. Figure [2.1] shows the overall layout of the TUNL laboratory.

2.1 Ion Sources [1]

For the proton elastic and (p,α) scattering data, proton beams were produced with the TUNL Direct Extraction Negative Ion Source, known at TUNL as DENIS II. (DENIS I, the original direct extraction source at TUNL, has been dismantled.) DENIS II is a product of the High Voltage Engineering Corporation (HVEC) and has been implemented at TUNL with few modifications. As the name suggests, H^- ions are

[1] An excellent overview of negative ion sources for tandem accelerators is given by [Mid74] (included in the collection of [Bro74]).

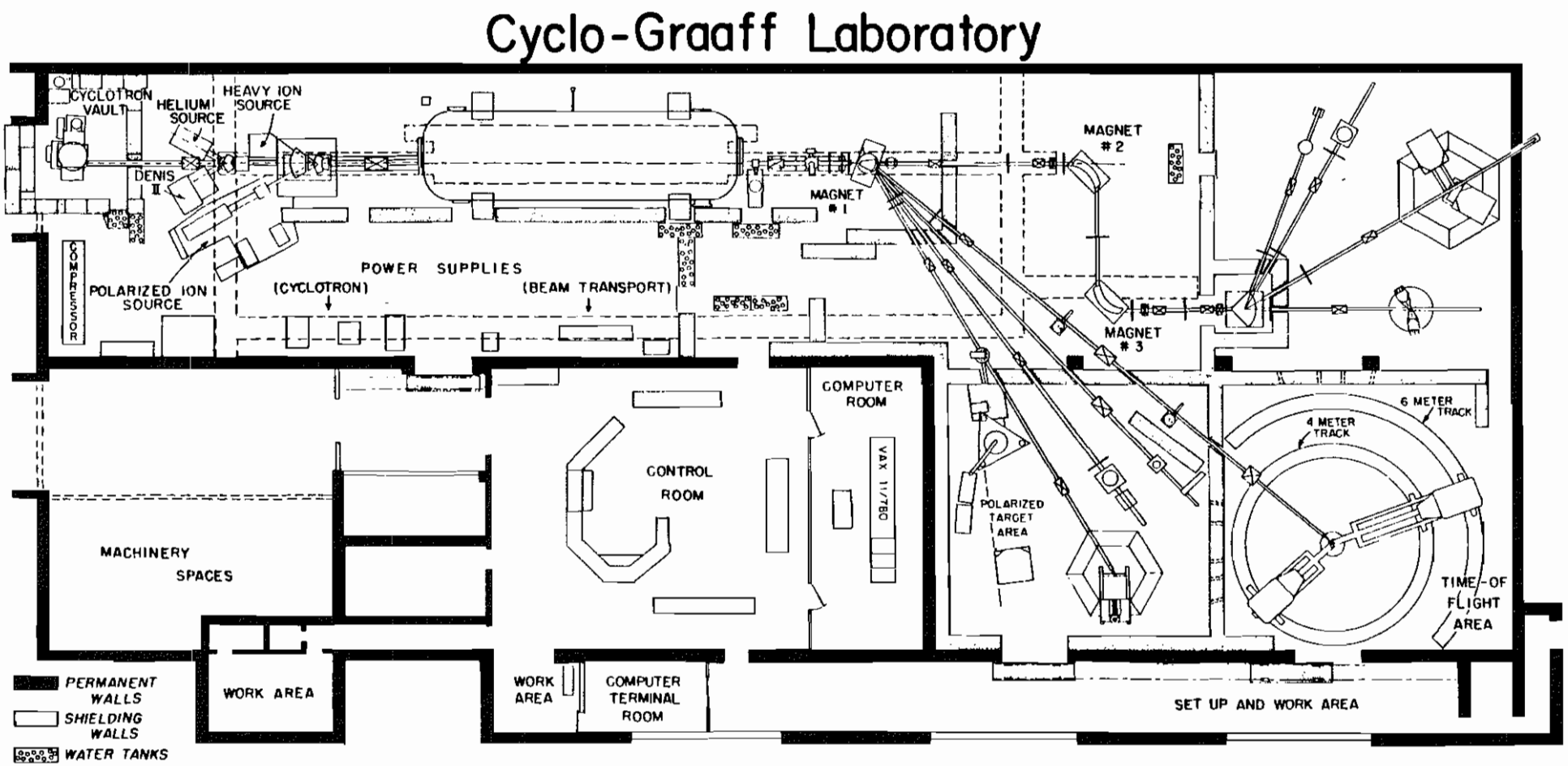


Figure [2.1]: Overall floorplan of the TUNL laboratory (1988).

extracted from a standard duoplasmatron assembly[2] [Law65] and accelerated to ~50 keV for injection into the tandem Van de Graaff accelerator. DENIS II is the main source used to produce unpolarized protons and deuterons at TUNL and is used by many different experimental groups. Maintenance and care of DENIS II is provided mainly by the TUNL technical staff, and the source usually operates with few or no problems with beam currents of up to 30 microamperes.

2.1.1 Helium Source

Alpha particle beams for the alpha elastic scattering data were produced by the TUNL helium source, sometimes referred to as the alpha source or the exchange source. Although capable of producing protons, deuterons, and a variety of heavy ions from different exchange gasses, the source is used almost exclusively for the production of $^4He^-$ and $^3He^-$ beams. The TUNL helium source is a High Voltage Engineering Corporation (HVEC) product (assembly # D-EN-SO-66), and a complete description of the original HVEC source can be found in the original HVEC manual (part # HVI1137). Unlike the DENIS II source, the helium source is not used extensively by different experimental groups, and the responsibility for its upkeep and maintenance rests mainly with its few users rather than with the TUNL support personnel. Indeed, prior to our experiment, the TUNL helium source had fallen into disuse and needed a complete realignment and extensive cleaning. Also in contrast to the direct extraction source, the helium source is an exchange source; that is, positive ions are extracted "on axis" from the duoplasmatron source head and then pick up two electrons in the exchange canal.

---

[2] The direct extraction of negative ions requires that the duoplasmatron source head be positioned slightly off-axis since that is the region of the greatest negative ion concentration. This differs from an exchange source which removes positive ions from a duoplasmatron which is aligned exactly with the anode or extraction aperture.

The resulting negatively charged particles are then suitable for injection into the TUNL tandem Van de Graaff. Vapor from a heated alkali metal usually provides the charge exchange medium, and the original HVEC design used lithium for this purpose. Since the probability of charge exchange is much less than unity and the helium atom must pick up not one but two electrons, the beam currents are much lower than those expected for direct extraction sources. Prior to our involvement with the TUNL helium source, normal α^- beam currents at the Faraday cup just after the first inflection magnet (the so-called "source cup") were on the order of one or two microamperes.

Previous work [Iko79] on the lowest $T = 2$ state in ^{28}Si via alpha elastic scattering indicates that the cross section for ^{24}Mg(α,α)^{24}Mg at backward angles in the appropriate energy range is very small. Also, experience tells us that beam transmission through the 90-90 magnet system is generally poor ($\lesssim$ 30 %) when the system is operating in a manner conducive to optimum energy resolution. Furthermore, to improve existing data, the target must be as thin as possible to minimize energy straggling effects. Because of these factors, an effort was made to increase the alpha beam current from the helium source at TUNL. Following the advice and technical drawings from McMaster University [Ontario, Canada], parts of the source were rebuilt and the charge exchange medium was changed from lithium to sodium vapor. Sodium has a higher cross section for charge exchange at the energies (~20 keV) of the alpha particles in the exchange canal. The major disadvantage of using sodium stems from the volatile nature of the material, making handling and cleanup more hazardous. However, with careful handling procedures, the advantage of a significant increase in the beam current outweighs the inconvenience involved in loading and cleaning the sodium oven.

As additional aids to improve the beam intensity, the plasma cup aperture size was increased from the original 13 mils to 18 mils, and, for the final data, a gauze filament was used in place of the usual tungsten wire. The nickel gauze filament, coated

with a carbonate material, operates at a much lower current than does the tungsten wire and seems to allow cooler and more stable operation of the duoplasmatron. Further improvements include the installation of a new main source diffusion pump and freon baffle, an almost completely new source water cooling system, and a liquid freon cooling system for the duoplasmatron source head. Other changes such as a direct contact heater for the sodium oven and a thermocouple temperature monitor were found to increase source instabilities and were not used for the final experiments. With the improvements made, α^- beams of >7 μA have been achieved at the source cup, although normal currents are usually about 3 to 4 μA, still an approximately 300% improvement in beam intensity. More complete instructions on operation, as well as diagrams, typical operating parameters, and filament design, may be found in the source operations manual available at TUNL.

2.2 Accelerator

After passing through an inflection magnet to select the desired particle species and charge state, the negatively charged ions from the ion source are focused into the tandem Van de Graaff accelerator. The TUNL accelerator is a model FN tandem Van de Graaff, the tenth produced by HVEC, and was delivered to Duke University on September 28, 1968. For more information on the model FN tandem as well as some other very interesting historical notes, an excellent collection of articles is provided by [Bro74]. Major upgrades pending, at this time the TUNL machine still retains its belt charging system and uses a nitrogen and carbon dioxide mixture as its insulating gas. Negative ions enter the tandem and are accelerated by a positive voltage at the center terminal. Upon arriving at the terminal, they pass through the stripper canal where they encounter either a thin carbon foil or a carefully regulated amount of oxygen gas.

Interaction with either gas or foil causes all or some of the electrons to be stripped away, leaving the ion with a net positive charge. The positive charge causes the particles to be accelerated away from the terminal to the other, "high energy," end of the machine at ground potential. Later, we will discuss the stripping phase and its contributions to the beam energy spread.

Of major concern for high resolution data taking is the regulation of the tandem terminal voltage. The voltage on the accelerator terminal may vary irregularly as with inhomogeneous belt charging or with a definite frequency as for the ~4 Hz "belt flop" (the rotation frequency of the charging belt) or the ~400 Hz noise from the belt drive motor. Because of the varied causes of instability, the terminal voltage is controlled by correction signals from several different sources. Feedback signals for short term fluctuations known as "terminal ripple" are produced by the capacitive pickup (CPU) plates mounted inside the tank. These plates and the associated circuitry sense the change in capacitance (after all, the tandem accelerator is a very large capacitor) resulting from a change in the terminal voltage. Under normal running conditions a difference signal generated by parts of the analyzed beam's hitting a narrow control slit can provide information on short-term fluctuations as well as long-term drifts. In the absence of beam current, signals for long-term drifts are derived from a device known as a generating voltmeter (GVM). The GVM is mounted in the tank wall and senses changes in the electric field caused by a voltage drift on the terminal.

Normally, correction signals for both long and short term instabilities affect the actual terminal voltage via the corona current feedback circuitry. That is, during the machine's routine operation, there is a current, resulting from ionization of the insulating gas, from the positively charged terminal to a set of sharpened metal pins near ground potential. These pins, called corona needles, are mounted inside the tank on a movable rod whose distance from the terminal can be adjusted by the user. Usually, the needles

are positioned to maintain an average "corona" current of about 100 microamperes. The circuitry connected to the corona needles contains a triode which allows the current to be changed in response to a varying grid voltage. Thus, by varying the voltage on the grid of this corona tube, one can control the amount of current from the terminal to ground and thus effectively regulate the charge on the accelerator dome. The nature of this feedback circuit makes it obvious that the time scale for voltage corrections is limited by the transport time of ions and electrons from the corona needles to the dome. Indeed, for the normal system of CPU, GVM, and slit feedback signals through the corona stabilization circuitry, only corrections for instabilities of $\lesssim$ 10 Hz are possible.

In order to correct for fluctuations of higher frequency than allowed by the corona current assembly, the so-called "fast feedback system" has been installed at TUNL [Wes84]. This system includes a 6kV amplifier mounted inside the accelerator terminal and capable of applying a correction voltage directly to the stripper canal. Since the correction signals are not limited by particle transport times, adjustments in the kHz range are possible.[3] The feedback signals necessary to drive the amplifier are provided by utilizing the high frequency information from the same control slits which produce the signals for the corona system. The signals are optically transmitted through a lucite rod spanning the low energy portion of the accelerator to a photomultiplier tube which is, in turn, coupled to the terminal amplifier.

Every effort must be made to eliminate noise and false feedback signals which may cause the accelerator to erroneously adjust the beam energy. One particularly troublesome attribute of the slit feedback system is that a purely positional change of the beam on the control slit produces signals indistinguishable, to the tandem, from those

[3] The terminal amplifier, in its present incarnation, has a linear range of at least 6 kV and is capable of swinging 7 kV peak-to-peak. It is capacitively coupled to the stripper canal and thus has no dc response. The response function (gain versus frequency) is approximately linear from about 10 Hz to 1 kHz. [Wes89]

produced by a true beam energy shift. Adjustment of some beam transport elements can produce very noticeable (several hundred eV) erroneous energy shifts in the excitation function data, and therefore, during the final data acquisition, beam tuning must be minimized. One attempt to minimize random positional fluctuations from the tandem utilizes feedback signals from sets of slits before the analyzing system magnets. These signals are sent back to electrostatic steerers which are supposed to keep the beam centered on the slits. In practice, however, the existing steerer power supplies are very noisy and introduce additional problems for the tandem regulation. In its present condition, this positional stabilization system is of dubious value and was, in general, not used during the final data acquisition.

As a whole, the fast feedback system has been shown to be very effective in reducing fluctuations (as reflected in the control slit difference voltage) in the 10 Hz to 1 kHz range for ~3 Mev proton beams [Wes84]. When functioning properly, the system is believed to contribute substantially to lowering the overall beam energy spread. The entire feedback system is shown schematically in figure [2.2].

2.3 Beam Transport System

In addition to the tandem's feedback system, the other major factor which helps determine the average beam energy as well as its energy spread is the high energy beam transport system. After the particles have been accelerated to the proper energy, they enter the TUNL dual 90° magnet system. Also known as the dual monochromator system or the 90-90 system, this collection of slits, magnets, steering and focusing elements, in conjunction with the fast feedback circuitry, form the heart of the high energy resolution data taking system. Usage of the phrase "high resolution" refers implicitly to the particular mode of beam transport from the tandem through the 90-90 magnet system.

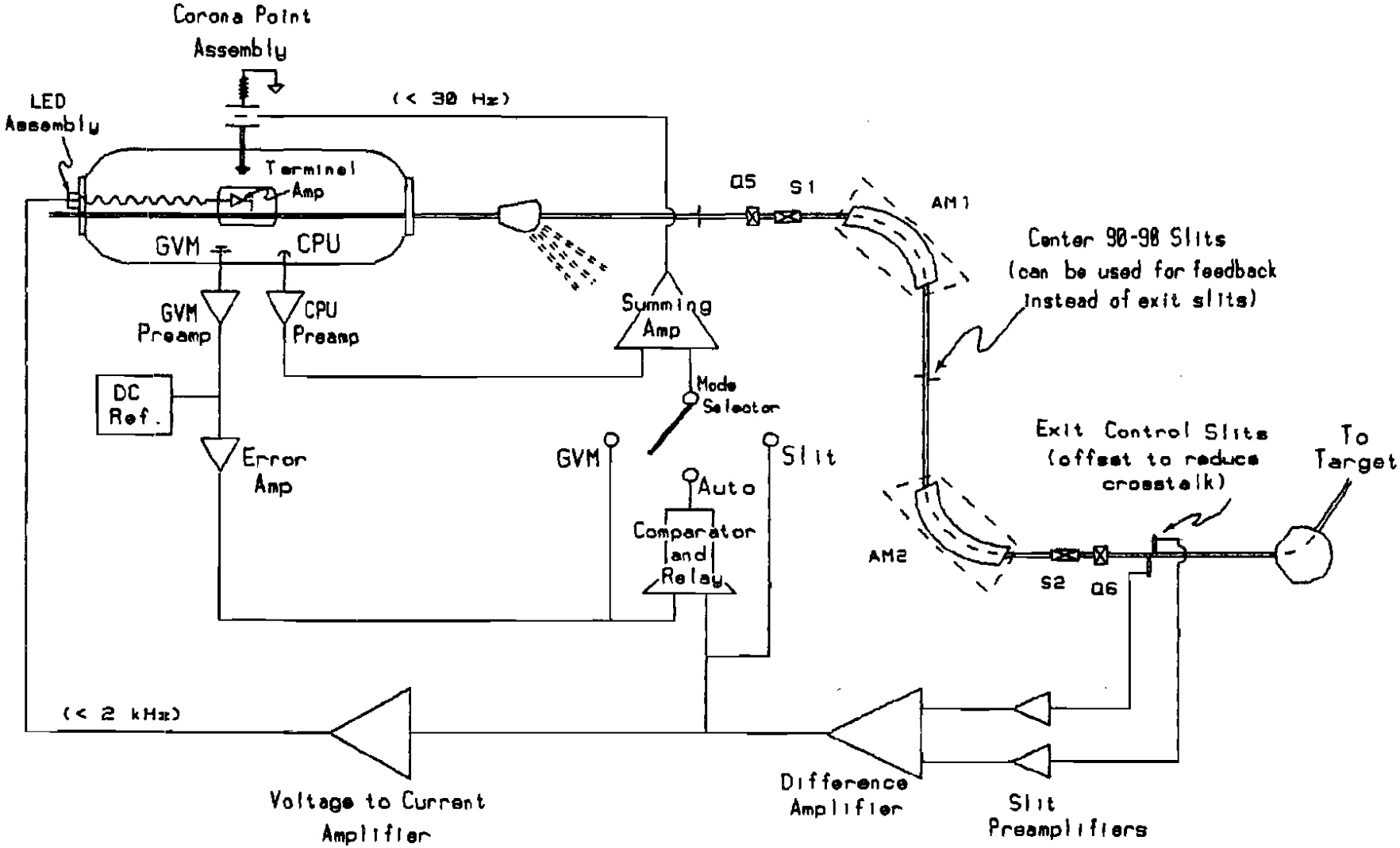


Figure [2.2]: Schematic of 90-90 magnet system and associated tandem control feedback circuitry. (see text for details)

Physically, the system consists of two oppositely bending 90 degree analyzing[4] magnets (AM1 and AM2), two quadrupole singlet (Q5 and Q6) and two sextupole (S1 and S2) magnet lenses, and these components are depicted as part of figure [2.2]. The particular configuration of elements is designed to transport beams of $ME/Z^2 \leq 60$ where M is particle mass in amu, E is particle energy in MeV, and Z is the net number of elementary charges per particle. The individual components were manufactured by the Varian corporation and detailed descriptions and specifications can be found in the original instruction manuals kept at TUNL.

The analyzing magnets, AM1 and AM2, are uniform field (so-called "field index" = 0), 90 degree bending magnets each with a central path radius of curvature of about 101.6 cm. The magnet poles are angled at the entrance and exit faces to allow first order[5] point-to-point radial imaging and to provide some focussing in the axial direction. The two magnets share a common power supply, and the field in the first magnet is regulated by a standard NMR[6] type feedback circuit. A small "trim" power supply connected to the second magnet makes up for any dissimilarities between the two magnets. The

---

[4] In general, any change in direction of an ion beam implies a separation of momenta and vice versa. We shall refer to an *analyzing* (short for momentum-analyzing) magnet as one whose main purpose is the separation of particles with different momenta and to which the necessary change in direction is unimportant. Futhermore, an analyzing magnet will usually have only one exit port at a fixed angle. Contrastingly, a *switching* magnet functions to steer particles into a particular one of several available beam lines.

[5] Approximations of first order theory include neglecting space charge forces, complete independence of the radial and axial motions, and a small angle approximation where $\tan\theta = \theta$. In general, in the equations of motion, all second- and higher-order terms are considered negligible. For a more detailed discussion, see [Liv69].

[6] The NMR (Nuclear Magnetic Resonance) circuit uses the signal from the spin flip of protons in a probe inserted in the magnet cavity. A tunable reference field inside the probe causes the spin flip to occur for well defined values of the external field. A feedback signal, induced by the spin flip occurrence, controls the magnet current to maintain the particular external field setting. [Sli78] [Fuk81]

quadrupole singlets, Q5 and Q6, are symmetrically placed before and after the analyzing magnets and operate in unison from a common power supply. Arranged to focus in the vertical plane and to defocus horizontally, they provide for maximum resolving power in the radial direction while adding the axial focussing necessary for proper beam transmission. The sextupole magnet lenses, S1 and S2, positioned just "inside" the quadrupoles, are included to minimize various second order aberrations. Output from beam optics computer codes such as TRANSPORT may be used as a guide for setting many of the optical elements of the beam transport system.

The charge species and absolute energy are determined in the 90-90 system through carefully selected magnet settings. The considerable resolving power of the system allows an extremely accurate absolute energy calibration. When proper procedures are followed, the energy reproducibility of the system[7] is $\pm$ 2 keV [Wil83]. These procedures involve "recycling" the magnets before the energy is set. Here recycling simply means taking the magnets through the same hysteresis cycle as in the calibration. The recycling routine, followed before each final data run, is as follows:

1) Saturate the magnets by turning the current to the maximum value and leave for two minutes.
2) Turn the current down to zero and turn the supply off (DC off) for two minutes.
3) Switch on the supply and increase the current to its proper setting, taking care not to overshoot. It is **not** permissable to go over and then come back down with the current setting.

Note that after this procedure we are permitted only to increase the energy, to step over

[7] Recently there has been some debate as to the effect on the calibration of an aborted attempt to realign the magnet system. It has been suggested that the calibration may be off by more than 10 keV [Moo89], but we find very good agreement, within errors, between our resonance energy determinations and most published values.

the excitation function, without recycling.[8]

2.3.1 90-90 System Resolution

Although providing a very accurate value for the mean energy, it is because of its importance in determining the beam resolution function that the 90-90 magnet system merits more careful consideration. The high resolution mode of beam transport is shown schematically in figure [2.3] and is characterized by its radial image point between the two oppositely bending magnets.[9] Details of the optics of this type of system, as well as complete descriptions of the optics of single dipole magnets are given by [Liv69].

As part of this high resolution mode, narrow slits are placed at the object and final image positions as well as at the intermagnet radial crossover. These slits, along with the dispersive properties of the magnets, are used to define the momentum resolution of the system:

$$R_p = \frac{\Delta p}{p} .$$

Here Δp is defined by the slits to be the allowed range of momenta for transmitted particles. That is, all particles with momenta less than $p - \Delta p$ or greater than $p + \Delta p$ are intercepted by the slits in the system (see figure [2.4]).

Similarly, we can define an energy range ΔE and an energy resolution

---

[8] We recognize that the usual recommendation for setting of magnetic elements is to saturate and then decrease the current to the desired value. However, for energy ranges near the maximum operating point of the tandem, it is best to proceed in increasing energy steps to allow for "conditioning" of the accelerator.

[9] There is also a mode of beam transport through the 90-90 system that does not require an intermagnet focus. This high transmission or zero dispersion mode is, as the name suggests, useful for larger transmitted beam intensities when the beam energy resolution is not important.

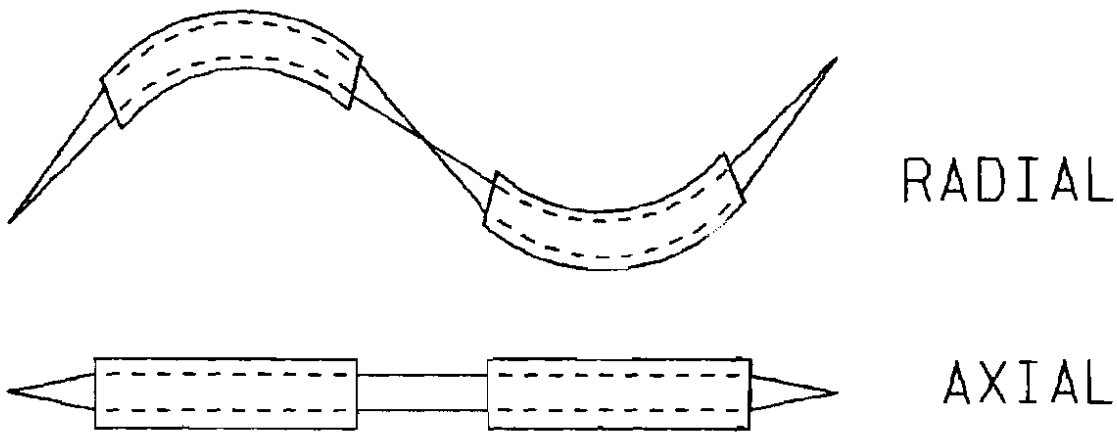


Figure [2.3]: The high resolution system consists of two oppositely bending magnets with an intermagnet radial image.

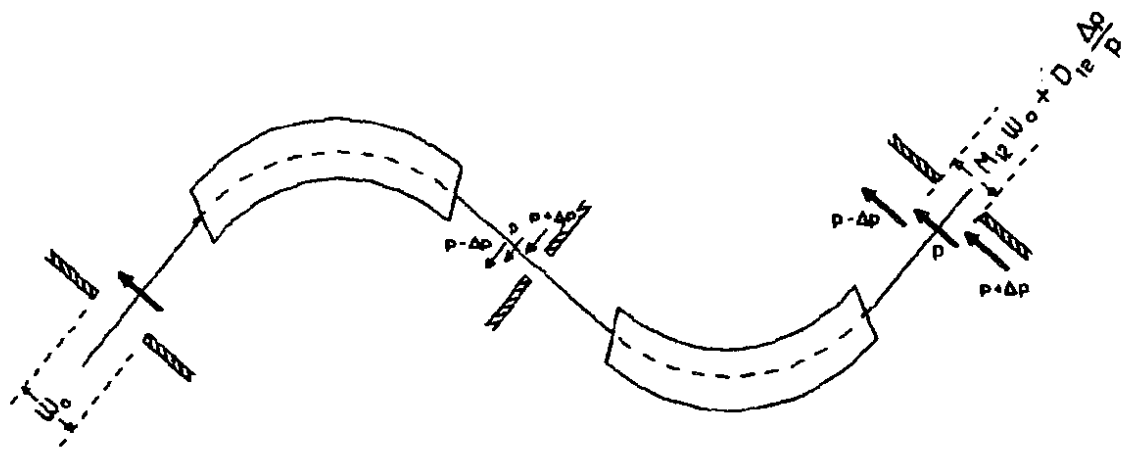


Figure [2.4]: Magnet dispersion and slit settings determine Δp for the 90-90 system.

$$R_E = \frac{\Delta E}{E}$$

which, for non-relativistic particles (energy << rest mass), is approximately equal to twice R_p. For the normal settings of the slits for high resolution (given in figure [2.5]), 100% transmission (for an extended source) is guaranteed only for those particles whose energy is exactly E. The transmission function applicable to the high resolution mode of the 90-90 system is given in figure[2.6] [Liv69]. Note that the full width at half the maximum value (FWHM) is given by ΔE.

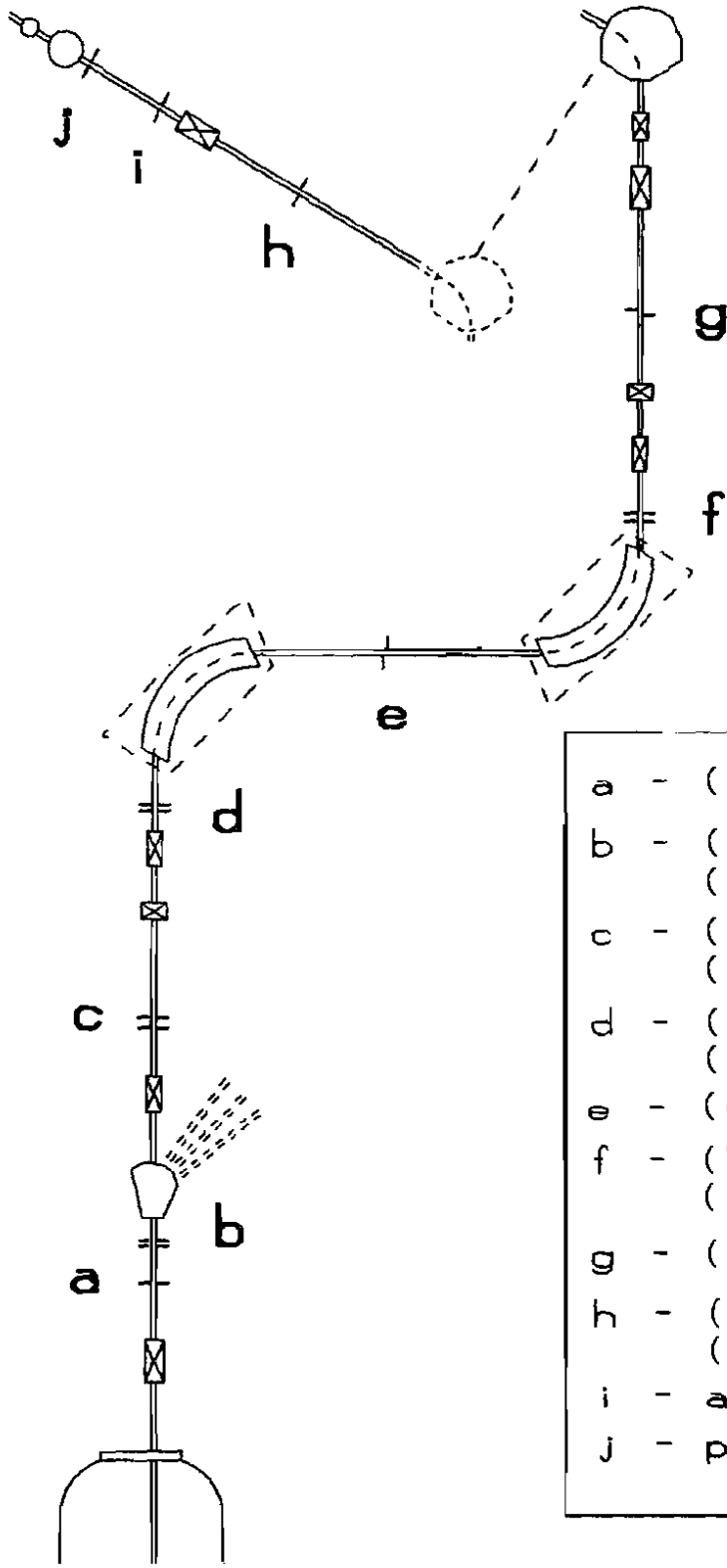
The resolution, R_E, indicated by this transmission function can be related to the properties of the system as follows:

$$R_E = -\frac{2M_{12}w_0}{D_{12}}$$

where M_{12} is the magnification of the two magnet system, w_0 is the object slit full width, and D_{12} is the dispersion coefficient. All of the factors can be related to physical properties of the system and for the TUNL configuration[10] $M_{12} \sim 1$, $D_{12} \sim -16.73$ meters, and $w_0 = 2$ mm. This gives a first order system energy resolution, R_E, of $\sim 2.4 \times 10^{-4}$. This calculated value enables us to estimate the system energy width for any particular beam energy. However, this estimated width is the ΔE value for the magnet system based on the transmission function and should not be confused with the FWHM of the beam profile. The beam profile or resolution function is a time average of accelerator output distributions **multiplied** by the 90-90 system transmission function. As we shall see, only for very poor input beam resolution will the resulting post 90-90 beam profile have a FWHM which is close to the magnet system energy width.

Further insight into the final shape and width of the beam profile can be obtained by considering some extreme cases for the 90-90 system operation and the input beam

[10] Note that these values for the system are valid only when the slits, magnets, and quadrupoles are set to their proper values.



a	-	(H) 500 -500
b	-	(H) 200-200 (V) 200-200
c	-	(H) 40-40 (V) 60-60
d	-	(H) 500-500 (V) 500-500
e	-	(H) 20-20
f	-	(H) 500-500 (V) 500-500
g	-	(H) 40-40
h	-	(H) 100-100 (V) 200-200
i	-	as needed
j	-	position 3

Figure [2.5]: Typical slit settings for high resolution experiments (numbers in mils).

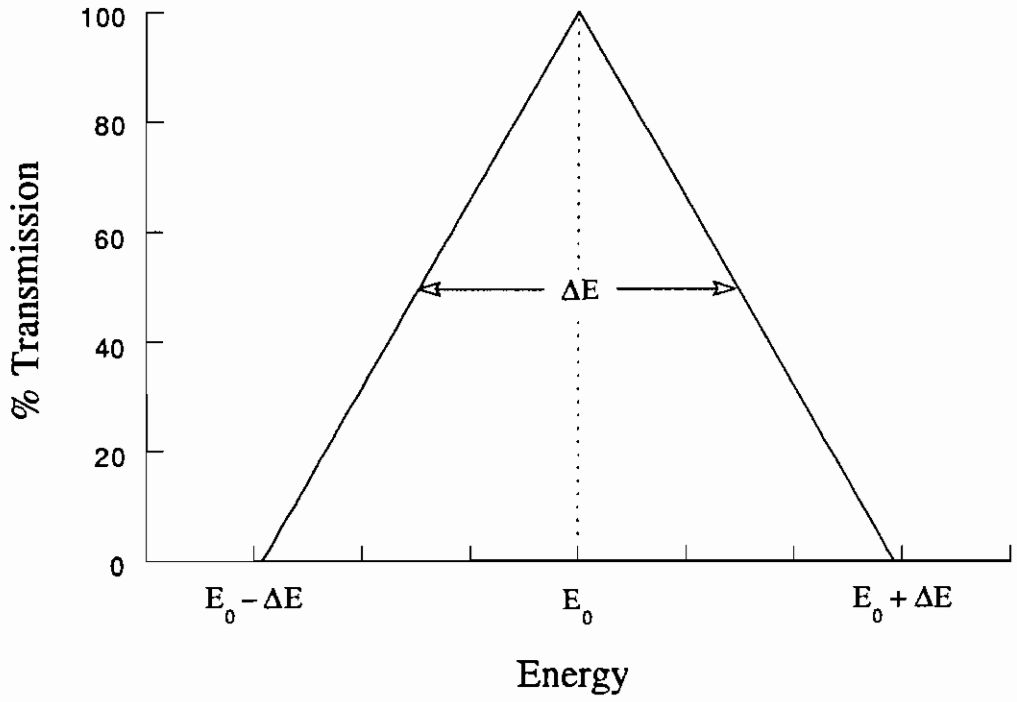


Figure [2.6]: Beam transmission function applicable to the TUNL 90-90 system in high resolution mode.

resolution functions:

1) Consider the input beam as having very poor intrinsic resolution.[11] The width of the energy distribution function is much greater than the width of the system's transmission function. Thus, over the range of ΔE_{system}, the input beam function is essentially flat. The output produced, the product of the two functions, has the shape and width of the system resolution function. Figure [2.7a] illustrates this case.

2) The input beam has very good energy resolution represented by a narrow (approaching a delta function) distribution centered at a particular mean energy, E (can be viewed as a perfect feedback system). The FWHM of the beam distribution is very much less than that of the system transmission function. Multiplying the input function by the transmission function has little or no effect on the shape or width, and output equals input. Figure [2.7b] represents this scenario.

3) The input beam has a very narrow ($<<\Delta E_{system}$) instantaneous resolution function, common for ion beams from the tandem. However, let us also assume that there is poor or no feedback to the accelerator so that no particular mean value of energy is preferred over another; that is, all centroid energies are equally probable. Now the time averaged[12] (sum of

---

[11] Previously the analyzing system had been used to transport beams from the now dismantled TUNL cyclotron. A cyclotron output would be an example of a particle beam with very poor energy resolution.

[12] Although the slit system itself is not discriminating about the causes of the energy spread, we note the distinction between energy fluctuations, occurring over time, and energy spreading resulting from energy loss mechanisms. Even though there may be some difference in the shape (fluctuations should be symmetric about the mean while loss mechanisms usually produce a low energy tail), the resulting distributions may look identical over a small enough range of energies.

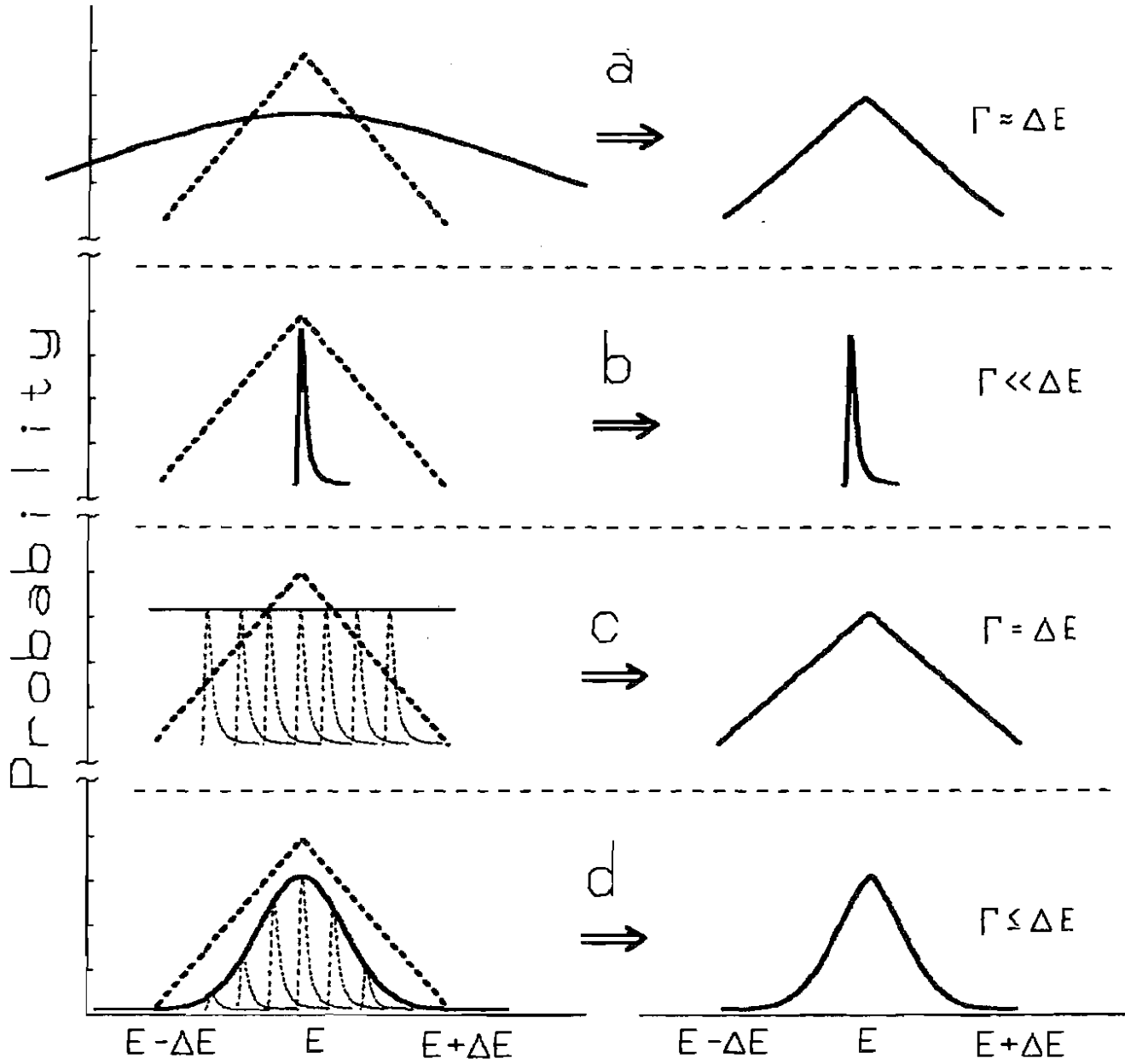


Figure [2.7]: Effects of the TUNL high resolution system. The bold dotted (triangular) function is the system transmission function. The solid curves represent the beam energy distribution before (left-hand column) and after (right-hand column) passing through the 90-90 system. See text for details.

instantaneous) input energy distribution will be essentially flat over the range of ΔE_{system}, and the situation will be like that of a). Figure [2.7c] illustrates this.

4) The input beam has a fairly narrow instantaneous beam energy distribution. The centroid beam energy fluctuates over time as in 3). However, the accelerator slit feedback system is functioning so as to make a particular energy, E, more probable than any other energy. Thus, the initially narrow instantaneous distribution "moves" inside some probability envelope whose width is determined by the effectiveness of the feedback system. Note also that the shape of the envelope will depend on the response of the slit feedback electronics. This situation, illustrated by figure [2.7d], is, of course, somewhere between b) and c), and the shape of the output beam resolution function will vary accordingly.

It is most probable that the above situation d) best describes the circumstances as they exist for the acquisition of high resolution data at low energies. This suggests that the beam resolution function is strongly dependent on the proper functioning of the fast feedback system. Furthermore, it suggests that for gas stripping and good ion source energy resolution, the fast feedback system, when working well, should produce better resolution beams than could the 90-90 system alone. However, we see also that the 90-90 system resolution, R_E, should always provide an approximate upper limit to the FWHM of the beam resolution function. Of course the dispersive properties of the magnet system are responsible for producing the separation of energies which is necessary to originate the slit current feedback signals -- a kind of "second order" effect on the beam

resolution.[13] Whatever the mechanisms involved, the spread of ion energies as well as the absolute beam energy is determined in the dual magnet system from a combination of carefully adjusted slits[14] and lenses and proper use of the aforementioned slit feedback system.

2.4 Target Chamber

After leaving the 90-90 system, the beam is bent by 60 degrees in a switching magnet (the 70-70 magnet) and is guided by several optical elements and slits on its way to the scattering chamber. Since the energy information has already been determined, this part of the ion's journey is simply a matter of beam transport and, although experimentally non-trivial, will not be discussed.

The scattering chamber is cylindrical (61 cm diameter) with detector mounting tracks on the top and bottom plates. The top of the chamber is removable to allow positioning of the individual tracks to any desired angle within a tenth of a degree. Once the chamber is sealed and evacuated, each set of tracks (top and bottom) can be rotated independently without disturbing the vacuum. Pressures in the 10^{-6} torr range are maintained inside the scattering chamber by a turbo-molecular pump mounted below the adjoining beam line. The turbo pump produces a relatively clean, oil-free environment, important for the prevention of contaminant build-up (by "cracking" of the vapors in the presence of beam heating) on the target surface. Also, to further reduce the risks of contamination, the chamber vacuum is "isolated" from the upstream beam-transport

---

[13] If the slit signals from the center 90-90 slits cause feedback circuit operation equivalent (either good or poor) to that produced by signals from the exit slits, then the beam resolution at the target should be nearly independent of the choice of control slits. This situation has been experimentally verified and noted by Wilkerson [Wil82].

[14] High resolution slit settings are tabulated in figure [2.5].

system by a liquid nitrogen cold trap in the beam line.

In the center of the scattering chamber, the particles impinge on the target prepared for the particular experiment at hand. The targets used in all our final experiments are solid targets produced by vacuum deposition onto thin carbon foils. The particulars of target selection and fabrication for each experiment will be presented in detail in chapter IV. Targets, up to five at a time, are mounted on an electrically isolated stainless steel rod inserted in the middle of the top plate of the scattering chamber. By vertically repositioning the target holder, any of the five targets may be selected without disturbing the vacuum system. Hardware has been developed (a detailed description may be found in [Moo89]) to allow a varying voltage (–5 kV to +5 kV) to be applied directly to the target holder. When this voltage is applied in steps of several hundred volts with a cycle time of several seconds, an energy excitation function can be obtained without adjusting the accelerator or magnet system. Because of the general shape of the applied waveform, the term "ramping"[15] is used to describe the system and the particular mode of data acquisition.

2.5 Detectors

The scattered particles are detected by standard silicon surface barrier detectors arranged at several angles inside the scattering chamber. Detector distances from the target can range from ~ 5 to 20 centimeters, but lengths of 10 or 12 centimeters are typical. Exact setups and detector configurations for each particular experiment are tabulated in chapter VI. However, a typical setup would include detectors at five or six different angles with the backward (>90 degrees) angles having detectors on each side of

[15] Actually, this term is a slight misnomer since the applied signal is actually a triangular wave and not a sawtooth or ramp.

the target in order to effectively double the count rate. The various detectors used had an active area of 150 mm^2, a thickness of 300 or 500 microns, and a rated alpha energy resolution of less than 1%. The best possible detector resolution is achieved by cooling the detectors with freon lines running inside the scattering chamber.

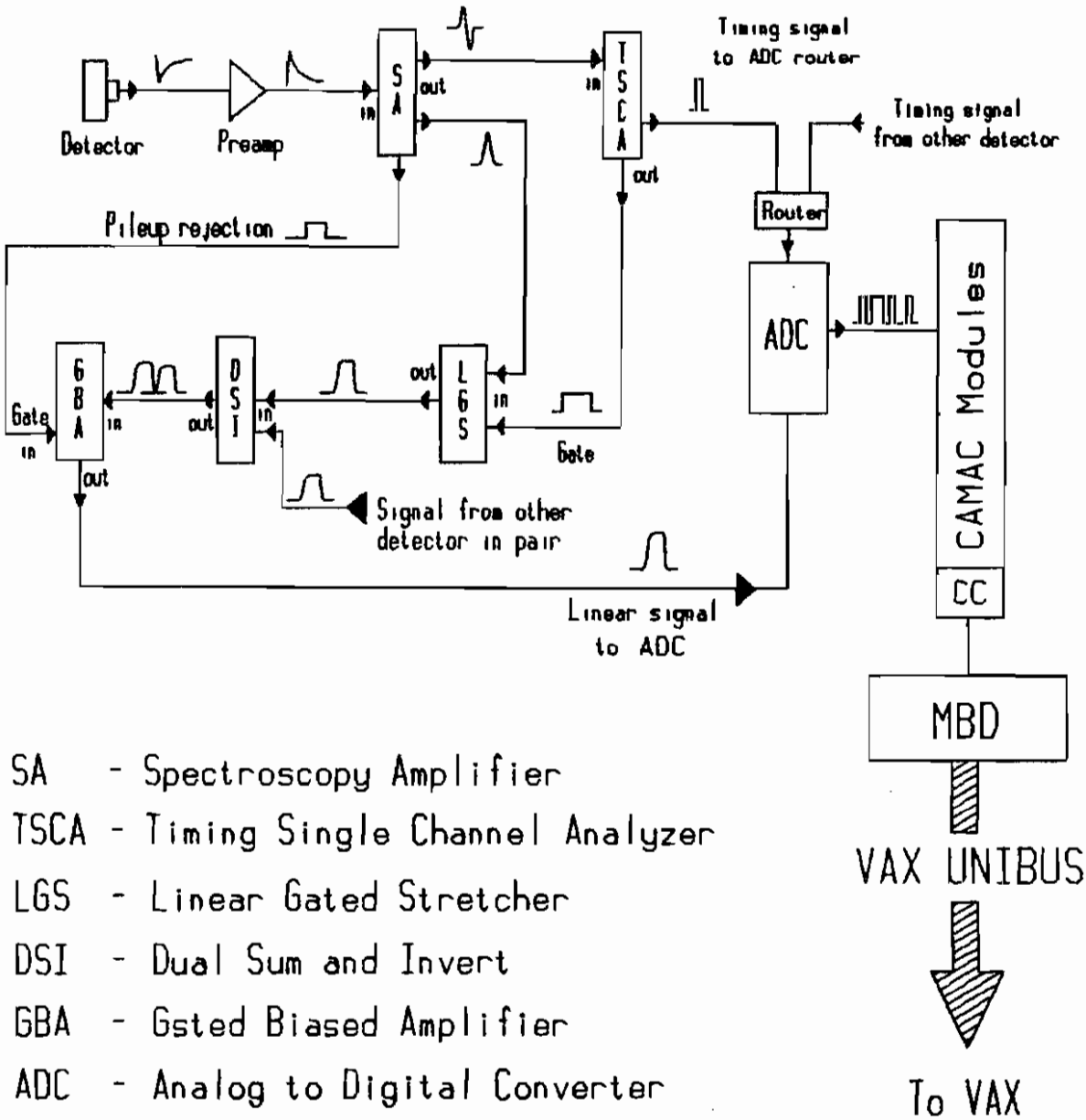
Dual rectangular collimator slits (typically 0.635 x 1.27 centimeters) mounted in front of the detector served to define the detector solid angle geometry and keep out unwanted particles which may be scattered from other parts of the scattering chamber. For our experiments, we replaced the old 15.9 mm thick tantalum collimator slits with thin (7.9 mm) brass ones to eliminate secondary scattering from the slits. This replacement provided cleaner spectra, especially noticeable for alpha particles. Also, magnets mounted to the collimators insured that stray electrons would not strike the detector face. Because of low cross sections at the back angles, it is desirable to increase the solid angle of the detectors by moving the detectors as close to the target as physically possible. In order to allow minimum distances from the target, as well as more backward angles and less angular distance between detectors, more compact detector holders were designed and built for these experiments. New collimators and electron suppressing magnet supports were also fabricated.

Unscattered particles exit the scattering chamber and impinge on a standard charge collecting plate or Faraday cup. The value of the integrated beam current is needed in the calculation of differential cross section. To prevent electrons from being ejected and producing an erroneous reading, the Faraday cup is fronted by a cylindrical suppressor held at about –60 volts. For several of the experiments an additional permanent magnet was placed outside the scattering chamber (near the target rod) to reduce fluctuations in the observed charge collection caused by electrons accelerated by the target voltage ramp.

2.6 Electronics

Scattered particles which strike the detectors generate small signals which must be amplified, shaped, and digitized for digestion by the TUNL VAX 11/780. The electronic circuitry used to process the analog signals from the detectors consists of several commercially available modules utilized in a more-or-less standard manner. From detector to computer, the signal processing hardware is shown schematically in figure [2.8]. Showing the short rise time and long tail characteristic of the rapid collection of charge and subsequent discharge through a resistor, the detector signal passes into a charge sensitive preamplifier. The preamplifier inverts the pulse and serves as a cable driver allowing the signal to be transmitted the considerable distance from the target area to the control room.

In the control room, the spectroscopy amplifier (SA) amplifies and shapes the signal and produces a Gaussian-like "unipolar" pulse and a differentiated or bipolar version. Note that it is important to preserve the pulse height information, and the SA must maintain a strict proportionality between input and output. From this output, one can clearly see the different amplitude pulses which correspond to the different energy particles which strike the detector. The bipolar output, whose zero crossing is used as a timing signal, is passed on to a timing single channel analyzer (TSCA) for conversion into a logic signal gate. The TSCA output, as well as other timing or logic pulses, have no amplitude information and are used for discrimination of noise or unwanted low energy events. This logic signal is adjusted in time to overlap the unipolar pulse, and both are input to a linear gated stretcher (LGS) for the generation of a wider, almost square, output pulse shape whose amplitude reflects the detected particle energy. The timing information is necessary because signals from other detectors, usually at the same angle, are added in via the dual sum and invert (DSI) module. The signals are passed to a



SA - Spectroscopy Amplifier

TSCA - Timing Single Channel Analyzer

LGS - Linear Gated Stretcher

DSI - Dual Sum and Invert

GBA - Gsted Biased Amplifier

ADC - Analog to Digital Converter

MBD - Micro-programmable Branch Driver

Figure [2.8]: Typical electronics setup for high resolution data acquisition.

gated biased amplifier (GBA) which, operating in anti-coincidence mode, allows pileup events (those which occur too closely in time) to be discarded.

Finally, the output of the GBA is passed to the analog to digital converter (ADC), which also receives the timing signals from the TSCA. For our experiments each ADC handles the signals from two detectors, and additional circuitry is provided for routing and timing the logic signals which are regenerated as gates for the ADCs. The digitized signals from the ADCs are then passed to CAMAC input registers. Through the management of the crate controller and the microprogrammable branch driver (MBD) running under the direction of user written DAP files, the signals are collected in buffer regions of the TUNL VAX.

2.7 Online Data Acquisition

Events from the VAX buffers are sorted and processed according to user specified instructions written within the XSYSTEM (usually shortened to XSYS) software framework. XSYS provides for very general storage of data, and virtually any digitized information can be handled through the CAMAC, MBD, and VAX network.[16] Thus, a digital scattering "event" may consist of several "words" in addition to the simple energy signal from the detector. A good example of multi-word event storage is the beam line monitoring system [Moo89] wherein measured quantities such as magnet currents and slit difference signals can be digitized and stored along with the scattering data. All or any part of these event words may be stored into FORTRAN arrays and can be displayed as histograms or spectra on a graphics terminal. The XSYS notion of *data areas* provides a

[16] For more information on the ADCs, CAMAC, MBD, and XSYS, the local reader is referred to the TUNL XSYS manual, an invaluable source of such information, written and compiled by several of the TUNL personnel. (For a general discussion and further references, see [Gou83].)

convenient terminology for referring to the event storage space and will be used to refer to either XSYS arrays or their pictorial representation as spectra.

The user supplied processing instructions or EVAL codes, in their simplest form, may read an event and direct the incrementing of an array element representing a certain energy deposited in a particular detector. The resulting one dimensional array can be displayed as the familiar energy spectrum of counts versus channel number -- essentially the function of a simple multi-channel analyzer (MCA). However, as may be inferred from the previous discussion, the situation can be, and usually is, considerably more complicated. XSYS allows the user to become involved in the sorting process by setting "gates" around particular portions of some data areas. Events which fall into these gates may then be sent to other data areas or direct the sorting of other related information.

For high resolution data taking, we have developed a system which allows for maximum flexibility in the "offline" (after the data acquisition) analysis while providing enough "online" (during the course of the data acquisition) information to guide the experimenter. The data acquisition system outlined below allows full use of the previously described target ramping setup while retaining the ability to collect data in the "stepped" energy mode.

2.7.1 Excitation Functions by Stepping

The so-called "stepping" format involves incrementing the beam energy by adjusting the field of the energy selecting 90-90 magnet system. In this manner the cross section versus energy excitation function is collected one point at a time and each such step is basically a separate measurement. The independence of the points makes this method very susceptible to errors such as energy shifts or beam position movements which show up as apparent discontinuities in the excitation function. Also, because one

is attempting to make very small percentage changes in the currents of the magnets, there are apt to be problems with the uniformity of the energy step sizes.

In order to minimize some of the errors, the energy incrementing is automated through a computer controlled stepping motor connected to a multi-turn potentiometer which regulates the currents in the magnets and the associated focussing elements. The NMR control circuitry is also computer interfaced and can be set and read automatically, providing excellent management over the absolute energy as well as the energy step. However, with this scheme there is still no defense against a change in an optical element, not the 90-90 magnets, which might cause an energy shift due to some position-to-energy dependence in the beam transport system. Also, any instability which causes the beam to hit another area of the target with a slightly different thickness, will cause a jump in the calculated cross section which may appear much the same as an energy shift in the excitation function. The stepping method is, however, very straightforward and useful in scanning energy ranges larger than the extent of the target voltage ramp. Typically, this method was used in the initial location of the resonance and as a guide to setting the mean energy for the target ramp.

2.7.2 Excitation Functions by Ramping

The ramping method of applying a varying voltage to the target has the obvious advantage of providing very even energy steps for a minimum effort. Perhaps more importantly, since the entire excitation function is scanned continuously, slight energy shifts such as those described above are averaged over the entire energy range. Thus, it is highly unlikely for ramped data to possess any glaring discontinuities, although it is likely that any resonance shape will be broadened by effects of this kind. However, resonance broadening is expected from many different sources and can hopefully be correctly

accounted for in the analysis procedures. Since all of the final data were taken using the ramping scheme, we will describe the procedures and programs as they apply to that format. However, the dialogue will be generally applicable to the stepping scheme, since, as far as the analysis programs are concerned, it can be thought of as a subset of the ramping method, the ramp being fixed at only one value.

For the ramped excitation function data, a stored scattering event consists of the processed detector signal and a digitized version of the target ramping wave form. The ramping circuitry provides the latter digital information, and it is read into CAMAC through one of the available input register modules. After receiving a scattering event, the MBD is directed by the DAP file to read the appropriate value of the ramp from the CAMAC register. The detector or exit-energy information is first sorted according to the simple MCA prescription described above. Then the exit-energy peaks of interest are gated through the XSYS program DISP, and counts occurring inside these gates are sorted into other data areas as a function of incident energy. As noted, the incident energy information is obtained from the target ramping circuit while the NMR controller still provides for the energy mean value. After sufficient counts are collected, these yield curves provide information about the existence, location, and shape of the resonance, without the need for any online calculations. Typically, once the resonance is centered in the ramp energy region, data is accumulated in this manner for many hours (perhaps days) with, ideally, only occasional intervention by the experimenter to save the data to disk or tape.

2.8 Offline Data Manipulation

Although sufficient as an experimental "road map," an excitation function produced in this manner has the disadvantages of consisting of yield instead of actual

cross section and of lacking an accurate background subtraction. To remedy this situation, a wide gate (32 channels), corresponding to several hundred keV, is set around each of the exit energy[17] peaks, and this entire region is stored as a function of target ramp step in a "three-dimensional" data area. Thus, we have a plot of counts (z-axis) versus entering particle energy ($\propto$ target ramp voltage) (y-axis) versus exiting particle energy (x-axis). Since the entire exit energy peak (generally $<$10 channels wide) and the background around the peak is saved, the background subtracted sums may be used in an offline analysis to compute an accurate value of cross section for each energy step. Note that the projection of this 3-D data area onto the xz-plane is simply a portion of the exit energy spectrum. This projection is very useful for setting gates for the peak sums and backgrounds to be used in the computation of differential cross sections. The data storage method is illustrated in figure [2.9].

2.8.1 Calculating Cross Sections

Many of the programs involved in the computation of cross sections are standard routines used for both online and offline calculations. These programs are outlined in appendix A under the heading HSYS. However, the offline calculations involve more complicated manipulations of data files and data areas since they must be applied to each (y-axis) "slice" of many 3-D data areas. Specialized offline programs are required for copying the contents of the 3-D areas into standard counts-versus-energy data areas[18] to allow for accurate setting of gates and proper functioning of the HSYS programs. Thus,

---

[17] Remember that the width of the peaks in the exit energy spectrum is governed by the detector resolution and should not, in any way, be confused with the experimental or incident beam energy resolution which is important in the resonance evaluation.

[18] Although XSYS provides means of adding data areas, it does not allow data areas of different types to be combined.

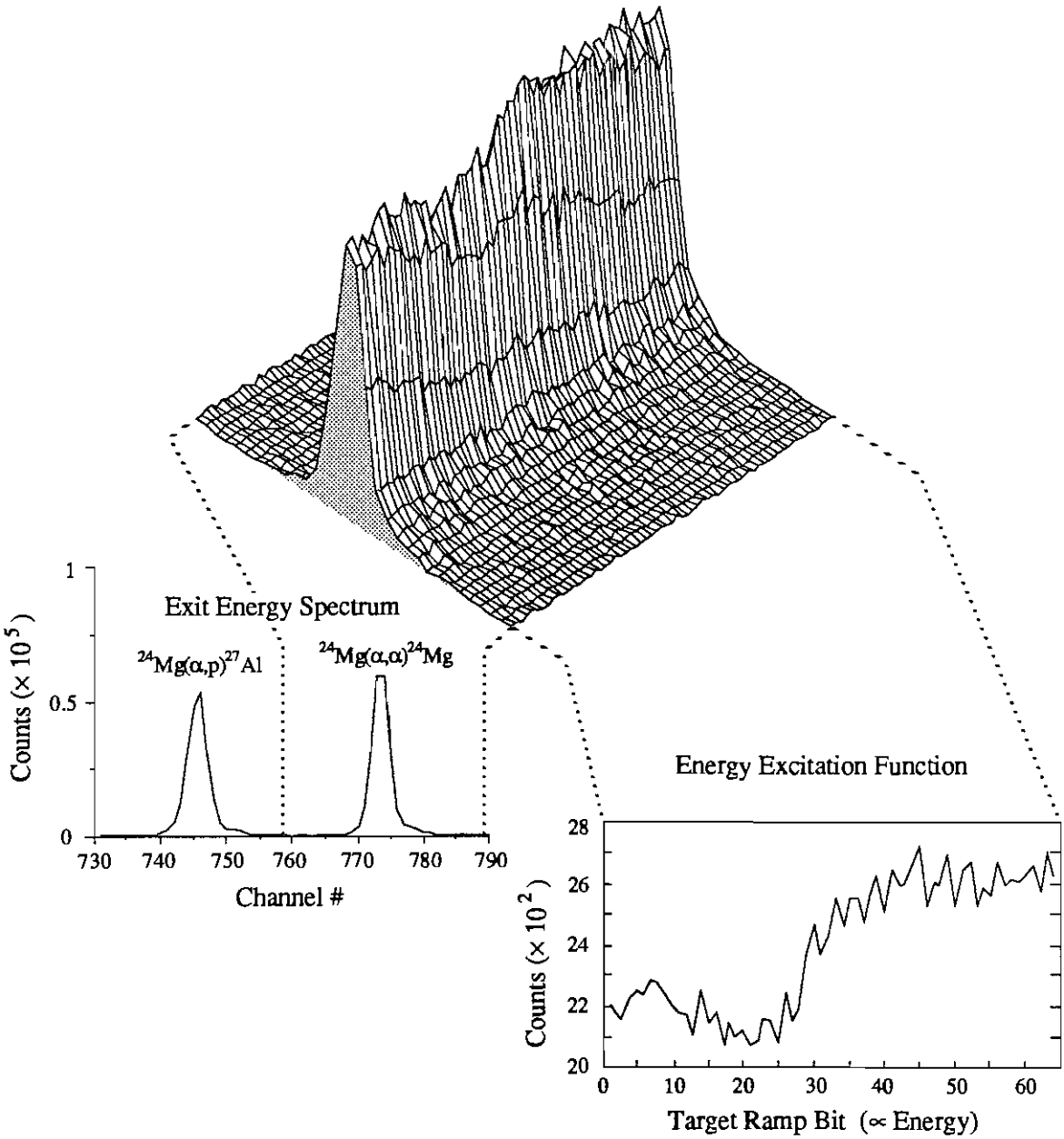


Figure [2.9]: Schematic of the high resolution mode of data acquisition. A user-defined gate selects a portion of the exit energy spectrum to be plotted versus the target voltage and stored as a 3-dimensional data area. Since the entire exit energy peak and surrounding background are stored (not just the peak sums), the offline analysis can provide an accurate background subtraction for each bit of the digital target ramp.

the m by n channel 3-D data areas are copied into $m \times n$ channel data areas by expanding along the x-axis; that is, the y-values are "side-by-side" as illustrated in figure [2.10].

Once the data are in this form, gates can be set in each segment for the peak sum and background, and the HSYS programs will calculate the cross sections. However, since there are usually 64 segments (corresponding to target ramp bits) for each of eight detectors for each reaction, this may correspond to well over 2,000 gates which must be set for each and every separate data file. We have developed programs which handle many of these chores automatically. Gates are set on the peak in the first segment, the cross section computed, the gate automatically shifted to the next segment, the energy incremented by the proper amount, the calculation repeated, and so on, until the end of the data area is reached. In this manner, files of cross section as a function of energy are generated for each selected reaction peak.

The situation is complicated, of course, by the fact that data are taken over a period of several days in a less-than-utopian environment. The fairly complex mix of hardware, software, and mental states of the different personnel produces inevitable deviations from ideal conditions. Whether purposeful or accidental, things change; sometimes for the better, often for the worse. Any type of change, however, is reason for caution, and the data are saved periodically (usually every couple of hours) in separate files constituting separate "runs."

Whether or not data-taking then continues or starts anew, the assumption is that everything stays the same **during** these runs. However, **between** runs, things often change,[19] and in the analysis, the runs must be combined to produce a single excitation function for each reaction at each angle. Of course, for high resolution resonance experiments we are most concerned with energy shifts which would alter the resonance

---

[19] Most discrete alterations result from human intervention, usually from beam tuning, resetting of software gates, changing targets, and/or adjusting electronics.

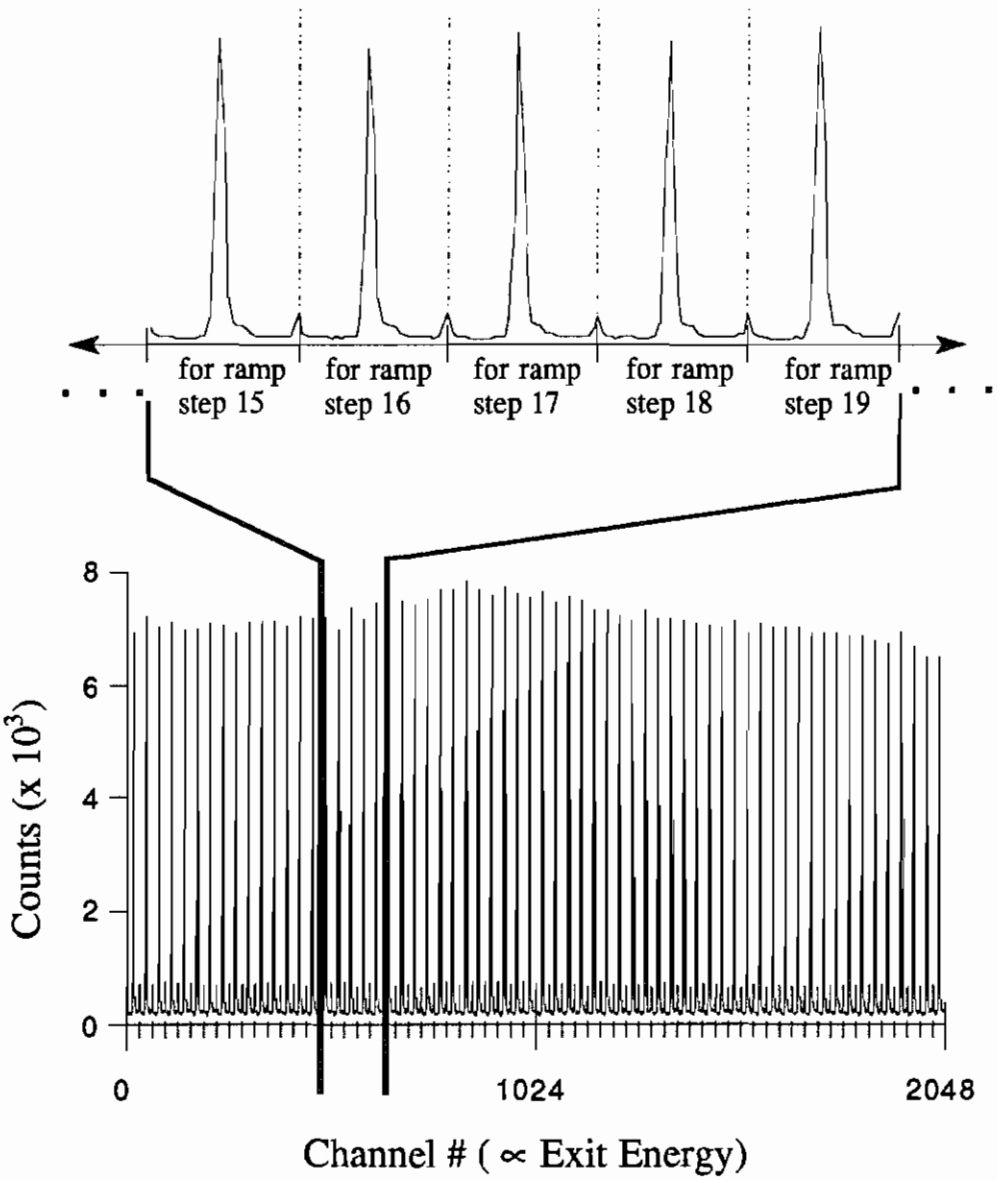


Figure [2.10]: Typical high resolution data storage illustrating the "binning" of the exit energy spectrum for different values of the target ramp. Here the exit energy slices, normally stored in a 3-D spectrum (figure [2.9]), are shown "side-by-side."

shape or even wash it out completely. At the same time we need to account for things like target thickness changes which might appear to alter the magnitude of the cross section and thus affect the extraction of the resonance parameters.

In the offline analysis, there are two obvious ways to combine data from different runs. We can either:

1) simply add the raw counts in the XSYS data areas and sum the scalers

or

2) perform the full cross section calculation for each run and then average these values and their errors.

Method 1) is the online equivalent of continuing a run (or never stopping at all), and the cross section calculation is then performed on the total spectra using the cumulative scalers. This very straightforward method based on simple addition[20] tolerates no changes between runs -- neither energy nor target thickness can be adjusted. The second method allows runs with different, perhaps even unknown, target thicknesses to be normalized to and combined with data in which the target thickness is known more accurately. Also, when runs are properly normalized, different energy regions can be combined. It is even possible to combine data of different energy step sizes by averaging with the nearest points or interpolating values from a cubic spline fit to the data. However, the computation is more involved than that of method 1), and the calculations, especially the error analysis, may not perform well on runs with very few counts (where the usual statistical approximations begin to fail).

In practice, both methods are used to extract the excitation functions from the

---

[20] Since spectra can be added, one might guess that they can also be subtracted from one another. Subtracting data areas can be a useful diagnostic for examining data from latter portions of a cumulative run. That is, if data are saved at some intermediate stage and then the run continued, the latter part of the continued run may be separated out by subtraction of the saved intermediate data. Although used for diagnostic purposes, subtraction was not involved in any of the final analyses.

experimental data. Method 1), because of its ease and simplicity, is used whenever feasible, but some situations demand the more complicated procedures of method 2). Even in the same experiment, two runs may be compatible for method 1), but method 2) might be needed to combine this total to another set of runs, taken at perhaps a slightly different energy. The package of FORTRAN programs and subroutines written to handle all of the chores of the offline analysis are grouped under the label OSYS and are listed with brief descriptions in appendix A.

These general procedures for data acquisition and analysis apply to each of the experiments described in this work. Sometimes, problems arise during a particular experiment that require slightly different procedures or variations on the general theme. Where applicable these specific anomalies will be discussed in chapter VI with the details of the individual experiments.

III. EXPERIMENTAL BEAM ENERGY RESOLUTION

The $T = 2$ levels, by virtue of the fact that isospin conservation forbids their decay by particle emission, are long-lived states and as such have very small energy widths. Widths on the order of several hundred eV are expected for these states, while their isospin-allowed neighbors normally have widths hundreds or thousands of times greater. We have already hinted at the importance of the energy resolution in our discussion of the specialized hardware necessary for measuring these narrow resonances, and we now concern ourselves with the experimental resolution effects themselves. Without careful attention, these effects may become quite large and wash out much or all of the detailed structure in the final data. The significance of this resolution smearing in the scattering data is evidenced by figure [3.1] which shows two separate observations of the same resonance in $^{24}Mg(\alpha,\alpha)^{24}Mg$. A poorer overall resolution for the first data set is the sole reason for the marked difference in the appearance of the functions.

When dealing with excitation functions of states with narrow widths, the objectives are twofold:

1) identify the origins of experimental energy broadening and minimize their consequences, and
2) determine as accurately as possible the consequences of each remaining contribution to the energy resolution.

There are several sources of beam energy spread which are of concern for the present scattering measurements. The first and perhaps easiest to understand is the regulation of the various accelerating voltages which the beam encounters in its journey from the ion source to the target. During normal operation, the ion source accelerating

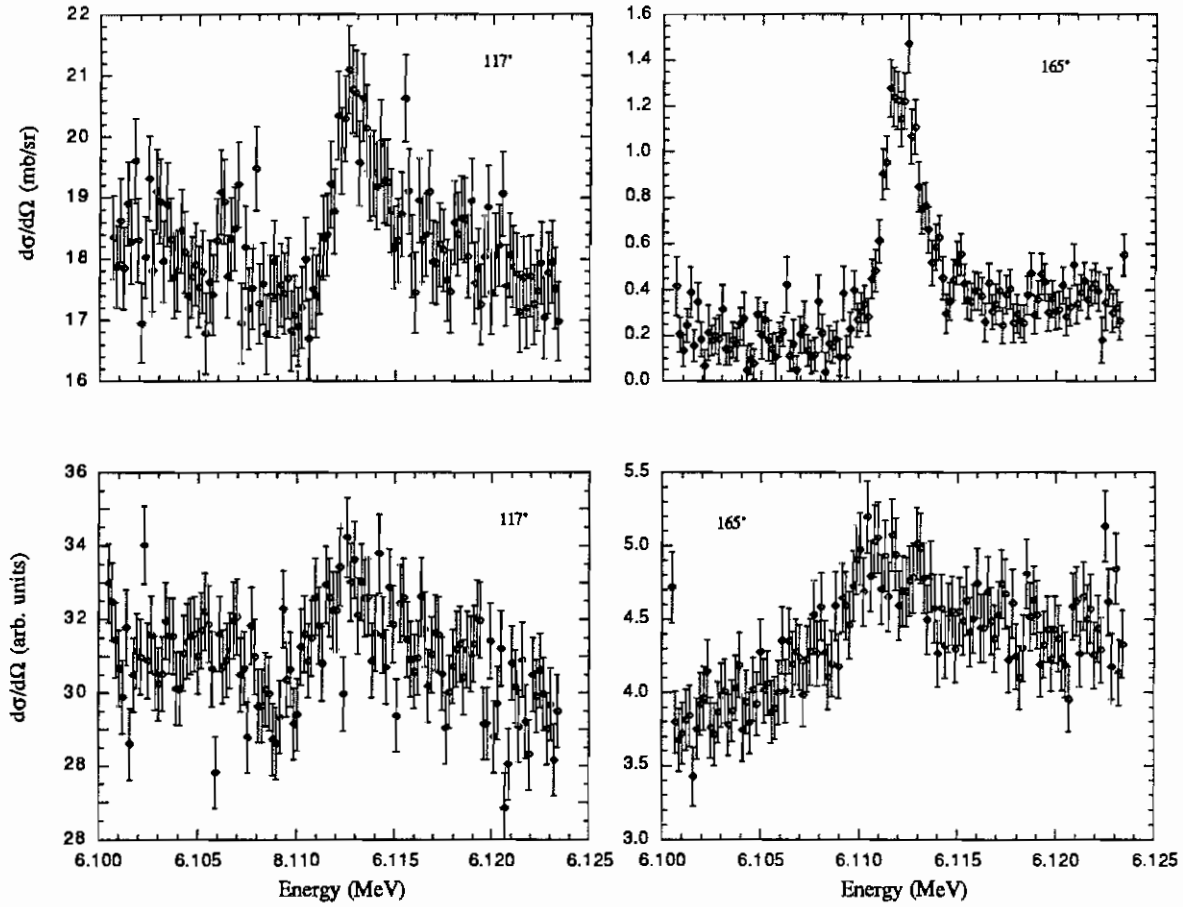


Figure [3.1]: Effects of resolution broadening for the lowest T = 2 state in ^{28}Si, seen via ^{24}Mg(α,α)^{24}Mg. The marked dissimilarity of the top and bottom resonance scans results solely from differences in the experimental energy resolution.

voltages have been found to be very well regulated, with a peak-to-peak voltage ripple (60 Hz sine wave) of 55 V at 20 kV for the alpha source attachment power supply. Correspondingly, the proton beam (from DENIS II) has a reported beam energy spread of ‹12 eV (FWHM) [How82]. Of greater concern is the regulation of the tandem accelerator voltage, and efforts to understand and minimize these fluctuations have already been expatiated in chapter II. The tandem's fluctuations are particularly troublesome, because they appear to vary considerably from experiment to experiment (even from day to day) and as such are extremely hard to estimate at any one time.

Another energy spreading mechanism, *Doppler broadening*, arises from the thermal motion of the target atoms. Because the atoms vibrate around their lattice sites, the nuclei are actually moving when the incident particle scatters from them. Although the atomic velocities are small compared to the incident particle speed, we will see that the effect on the energy spreading is not at all negligible. This mechanism affects the beam particles only at the time of the nuclear scattering event and produces a random (normal) distribution of energies with no average loss or gain of energy. Furthermore, contingent on knowledge of the target material's solid state properties, the width of the distribution may be crudely estimated for some targets.

A third source of energy spreading, one that is present whenever charged particles pass through matter, is inelastic collisions[1] with atomic electrons. The asymmetric distribution associated with these energy losses is known as *straggling*. Straggling is present in the residual gas in the beam line (normally negligible) and in the accelerator stripper canal, as well as in the target material. The consequences are particularly important in the ion source where the particles are moving relatively slowly and are much

[1] Inelastic collisions dominate at our energies and target thicknesses. Elastic collisions with atomic electrons (Ramsauer effect) are completely negligible for projectile energies greater than several hundred eV.

more likely to interact with the atoms found there. Straggling calculations for higher energy particles through thicker materials can be fairly accurate, but many of the approximations break down for very low particle energies or very thin targets. Trying to estimate the straggling for the beam through the transport system would be very difficult and probably of little value, and available target straggling calculations provide a rather poor representation of the actual straggling distribution [Moo89]. However, the average energy loss can be calculated fairly accurately, and we know that in our projectile energy range and for the traversed material thicknesses involved, the straggling distribution envelope is asymmetric with a long tail in the direction of greater than average energy loss.

It should be emphasized that these previously mentioned energy spreading mechanisms are those that are known to contribute to the overall energy resolution. As such they are relatively well understood, though in some cases "understood" means that we realize we cannot estimate them accurately. There are undoubtedly other energy spreading effects such as those from the fine structure of the target material (so-called *atomic effects*) which are not well understood and therefore nearly impossible to estimate. Also, note that we can safely ignore energy spreading mechanisms such as secondary scattering events (e.g. beam line slit scattering), since those particles should never reach the target or should be so degraded in energy as to be "gated out" in the software.

For the purposes of attempting to estimate the total resolution function, it is convenient to group the energy spreading contributions into three categories and to approximate each with a separate distribution. Thus, we consider the beam energy distribution before striking the target (so-called "incident" beam), while traversing the target material, and during the time of the nuclear collision. These three distributions will then be folded together to yield the total resolution function.

3.1 Incident Beam

The beam energy distribution from the TUNL 90-90 system contains contributions from many varied energy spreading sources (e.g. power supply regulation and collisions with gas molecules in the beam line), most of which cannot be easily quantified. Even the shape of the distribution is a complicated combination of parameters from straggling, transmission functions, and electronic responses. In the limiting case of ideal electronic responses, good feedback, and negligible straggling we would expect a generally Gaussian shape, since much of the energy spread is the result of random voltage fluctuations in the tandem accelerator and ion source. Straggling in the residual gas in the beam line, especially in the source exchange and tandem stripper gasses, would be expected to produce an asymmetry in the form of a low energy tail for the beam energy function. The many beam line slits, particularly those in the 90-90 analyzing system, will also affect the shape of the final distribution, and deviations from the ideal transport system may influence the beam energy function in unpredictable ways. We can, however, expect a beam energy function whose width is less than or equal to the resolution of the 90-90 system (section 2.3.1). The bottom line is that estimating the form of the incident beam energy distribution is indeed a risky undertaking, and undoubtedly the best solution would be to measure the energy function directly.

At TUNL we have developed two separate systems to attempt to measure this energy distribution of the beam prior to impingement on the target material. One measurement scheme, which, following historical precedent, we will designate with the adjective ΔE, utilizes specially designed hardware and the dispersive properties of the TUNL 70-70 magnet [Wil82]. The other system, developed by Mooney [Moo89] in an attempt to alleviate some of shortcomings of the ΔE system, exploits the existing high resolution hardware and mode of data acquisition.

3.1.1 The ΔE System

The ΔE system, first implemented at TUNL by Wilkerson [Wil82], is conceptually simple but in practice has some serious drawbacks. In order to obtain a useful figure for the beam resolution, the measurement needs to be taken at a point where the beam energy distribution will not be further affected by the beam transport system. At TUNL this means the ΔE hardware is best positioned after the 90-90 magnet system. Physically, the system consists of two magnetic sweeping coils positioned between the last 90 degree magnet and its output or object slits. After the output slits are another pair of slits, SL1 and SL2, the 70 degree switching magnet (used here as an momentum analyzer), a pair of electrostatic deflection plates, an image slit, SL3, and a small scattering chamber equipped with a detector at zero degrees. All components of the ΔE system (except, of course, the 70-70 magnet) are designed so that they can be physically or effectively removed from the beam line[2] to allow for normal high resolution beam transport.

Basically, the magnetic coils sweep the beam in the horizontal plane across the narrow opening provided by SL1 and SL2 thus producing a virtual point source (in the horizontal plane) which "emits" particles with energies representative of the total beam energy spread.[3] The two sweepers, driven at asynchronous frequencies, are needed to

---

[2] The slits and detector are mounted on movable air stops and can be moved in and out from the control room. The electrostatic plates and magnetic coils can, of course, simply be turned off to allow normal beam transport.

[3] SL1 is approximately 10 μm in the horizontal plane and SL2 is .41 mm (vertical) by .051 mm (horizontal). For a fixed magnet dispersion, the greatest sensitivity in energy versus position at the image can be obtained by using the smallest possible object slit. The width of SL1 represents the smallest opening which was practical to manufacture.

eliminate any angle to energy dependence of the particles entering the ΔE system.[4] After passing through the object slit, SL1, and the acceptance limiting slit, SL2, the particles are momentum analyzed by the 70-70 magnet and are focussed onto the image plane.

As a result of passing through the 70-70 magnet the particles have a horizontal position-to-energy dependence at the image plane. This position-versus-energy function can be inferred from physical properties of the magnet, and thus a measurement of the relative number of particles at each position would give us the resolution function that we seek. Positioned in the image plane is the image slit, SL3, and a solid state detector mounted directly behind. The width of SL3 is determined by the magnification of the 70-70 magnet and is just wide enough to pass the (horizontal) image of SL1 (assuming point-to-point focussing of a mono-energetic beam). Since there are practical problems with moving SL3 and its associated detector across the beam, electrostatic sweepers just upstream from SL3 move the beam across a stationary slit and detector. Distributions of counts versus voltage are collected, and the horizontal axis (voltage) is calibrated to energy by changing the 70-70 magnet by a known field (and hence known ΔE) and noting the shift in the peak centroid.[5]

The attractiveness of this system lies in its conceptual simplicity and in the proposed non-invasiveness of its operation. In theory, the ΔE elements could be switched on or inserted into the beam line at any time during an experiment and removed once the measurement is completed, all without disturbing the high resolution setup and operation. In reality, one might expect things to be slightly more complicated, and indeed they are.

---

[4] More precisely the sweeping coils provide a means for sampling the beam's entire horizontal phase space (divergence versus position).

[5] For more details on the principles of the measurement, the reader is directed to [Hin75]; for principles as well as a complete description and drawings of the TUNL apparatus, see [Wil82].

Problems arose from the initial installation of the system. Space limitations dictated that the magnetic sweeping coils be positioned upstream from the 90-90 output slit. This means that during operation the tandem accelerator's feedback control must be derived from the slits between the two 90° magnets. Also, the quadrupole lens near the 70-70 magnet has to be readjusted for the ΔE system to provide the proper point-to-point focus on which the measurement depends. Although deviating from the ideal procedures, these adjustments are minor and probably acceptable. However, in practice, alignment of the narrow slits seemed always to be a major problem, and the ΔE measurement required major retuning, usually for many hours, of the beam transport system. Although this deviates greatly from the spirit of the ideal system operation, there is still hope that the system can yield meaningful results.

Perhaps an even more serious drawback arises from the ΔE system beam optics and whether or not the devices involved can be operated without aberrations within the accuracy demanded by the theory. Errors of this kind are very troublesome, because one can never be sure of their existence or extent -- they produce effects which are indistinguishable from those produced by the beam energy spread which we hope to measure. With this latter problem in mind, we may optimistically still hope to obtain values, i.e. distribution widths, which can be interpreted as upper limits.

Because of the many problems involved with the ΔE system measurements, we do not rely on the results for the determination of the beam resolution functions for the final fits to the data. However, we did use the system to obtain values for distribution widths for a 5 MeV proton beam and a 6 MeV alpha beam.[6] These values are interesting as comparisons with previous estimations and, in the alpha case, for comparison with the

[6] For operation with alpha particles, the magnetic sweeping coils were replaced with water-cooled coils capable of operating at higher fields. The coils were designed for maximum field operation compatible with the available power supplies.

results from the "ramp-ramp" determination. Approximating a Gaussian shape in both cases, the results for the full width at half maximum (FWHM) are:

Beam constituent	Mean Energy (Mev)	FWHM (eV) *
proton	5.0	680 ± 160
alpha	6.0	~ 930

* Values for gas stripping

3.1.2 The "Ramp-Ramp" System

The second system of energy spread measurement hardware and software, which we designate as the "ramp-ramp" system, depends on varying the field of the second 90° bending magnet in the 90-90 beam transport system. Since the system is a momentum analyzer, the effect of such a changing field is to move the beam across the output slit in such a way as to allow particles of different energies[7] to pass through the slit for each particular setting of the magnet. Up to sixteen different magnet settings are used to sample the entire range of energies of the beam, and the energy range is scanned many times. The period for each step is about one minute, and the data are usually accumulated for several hours. The system is shown schematically in figure [3.2]. For more detailed information about the sweeping system hardware, see [Moo89].

During the period of the magnet's stepping, normal counts versus energy excitation function data are obtained, but data for each magnet step are stored in separate bins. The result is sixteen different excitation functions, each of the same resonance, but

[7] Here, we assume that the beam is made up of particles of the same charge and mass, and thus the magnet is also an energy analyzer.

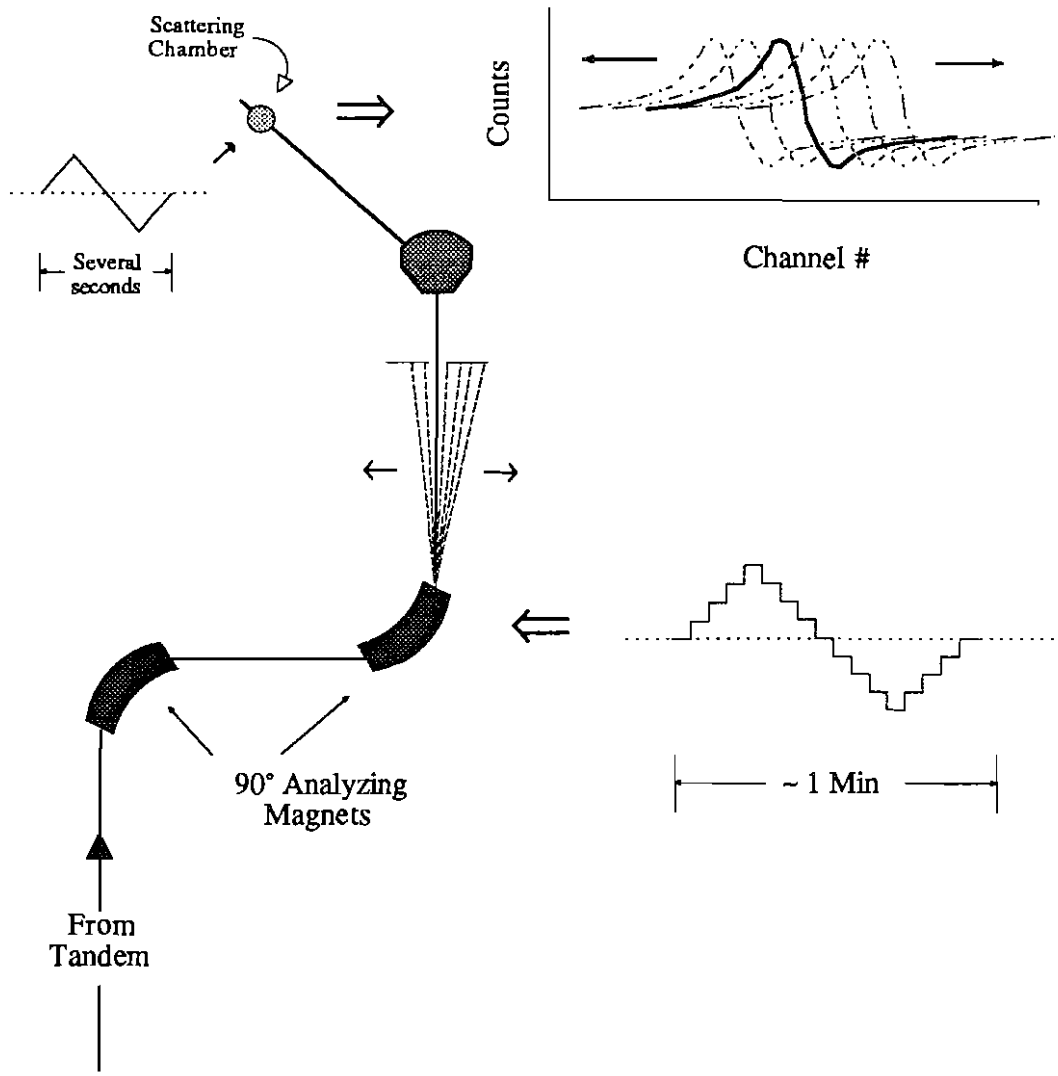


Figure [3.2]: Schematic of the principles and operation of the "ramp-ramp" system for measuring the beam energy spread of high resolution beams at TUNL (for details, see section 3.1.2).

each taken with particles of slightly different energies.[8] Since the spectrum's horizontal axis is an energy axis, the effect of using particles of different energies is evidenced by a horizontal shift in the resonance position. Thus, the resonance appears at a slightly different energy in each of the sixteen different bins. This system was used to obtain the resolution function for the 6 MeV alpha beam used in the $^{24}Mg(\alpha,\alpha)^{24}Mg$ scattering and the 4 MeV proton beam of the $^{23}Na(p,\alpha)^{20}Ne$ experiment. A sample of the data, shown as a 3-D plot, is given in figure [3.3].

Since the spectrum's *y*-axis may be calibrated to the incident beam energy from the target voltage and the vertical *z*-axis provides relative counts, we can extract a counts-versus-energy function which should represent the beam envelope. Because the analysis depends strongly on the specifics of the data, energy spread distribution determinations will be discussed in chapter VI along with the particulars of the experiments.

One particular shortcoming of this system lies in the fact that the profile measured is that of the beam as it exists at the slit **between** the two 90° magnets. This is necessary because we are using the second magnet as part of the measurement system, and it is thus not available for normal beam transport or as an aid to improving the resolution.[9] Logic dictates that the center slit beam resolution is approximately the same as that occurring at the output slit, and hence at the target, only if the fast feedback system is working well. That is, if the feedback system can produce beam energy spreads smaller than possible from the two-magnet system alone, then the contribution of the second magnet is nearly irrelevant (see section 2.3.1).

---

[8] Excitation functions are built up using the variable target voltage as described previously. The name "ramp-ramp" derives from the simultaneous use of triangular waveforms to drive the magnet current and the target voltage.

[9] The resolution, $\Delta E/E$, of the two magnet system is half that of one magnet alone (i.e. twice as good).

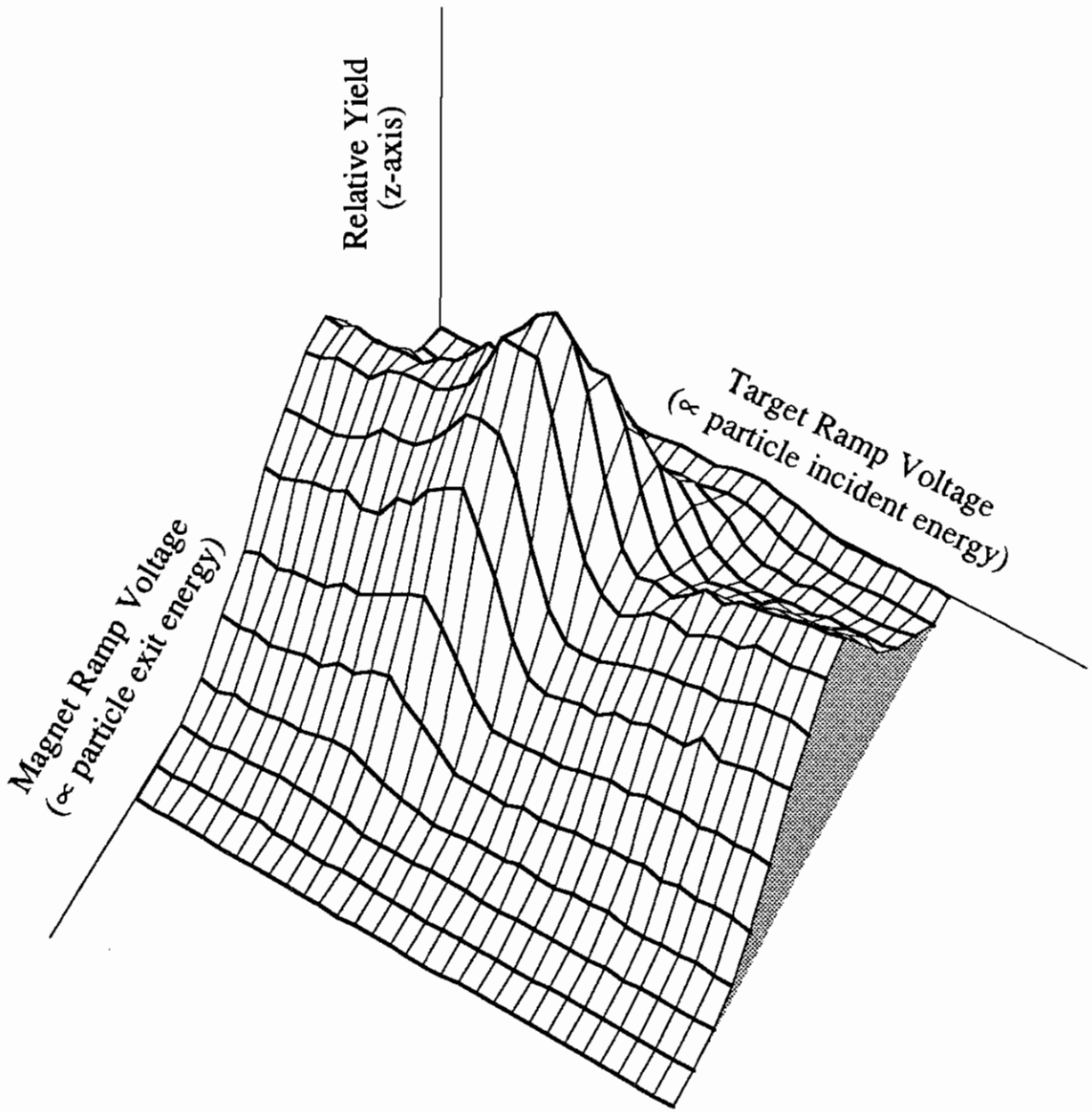


Figure [3.3]: 3-dimensional graph of the data for the direct measurement of the incident beam energy spread.

An example of when this is not the case is given by Mooney [Moo89] regarding the determination of the energy spread of a 7 MeV proton beam. The measurement (at the center slits) yields a beam profile with FWHM of 2100 eV, but the observed resonance width is only 1700 eV. From this, we can only conclude that the feedback system was working poorly, and thus the beam energy spread is definitely affected by the second 90° magnet. In the analysis of the data, Mooney finds that the resolution function width must be reduced by a factor of two. As we have noted, the resolution produced by the two magnet system is half that of a single magnet alone.

However, under normal conditions, measurement of the beam profile at the center slits is not, in itself, terribly inaccurate, especially when one considers the many approximations and statistical uncertainties inherent in the analysis and interpretation of the measurement results. Also, as we have seen, severe failures of the scheme are apt to be rather obvious, and the experimenter is not likely to be mislead by these erroneous results. Later, in the discussion of the specifics of the measurement, it will be evident that beam profiles obtained in this way can be regarded as approximations of the true pre-target beam resolution functions.

3.2 Target Straggling

After entering the target and before taking part in the nuclear scattering, the projectile ion must pass through some unknown amount of target material. We approximate the energy losses in this region as resulting solely from inelastic collisions with atomic electrons. It is impossible to predict, for any one particle, how much material is traversed and how much energy is given up in the atomic scattering(s). However, we can accurately calculate the average energy loss and in some cases describe the

distribution about this mean value.[10]

The computer program STRAGL (on the TUNL VAX) represents, for monoenergetic incident beams, thin target straggling with the Landau or more general Vavilov distributions [Sch74]. Both these distributions are centered about the average energy loss and are very asymmetric, with long tails in the direction of increasing energy loss. However, in our target thickness range (several $\mu g/cm^2$) STRAGL is successful only in providing the gross features of the energy loss function, i.e. the first moment and perhaps the shape of the envelope of the distribution. These gross features may, however, be all that is needed for our purposes, since this distribution will be convoluted with two others to form the total resolution function. Also, since the actual energy loss distribution is almost impossible to calculate ([Bic74] [Moo89]), STRAGL is basically "the only game in town." For a more detailed discussion of the validity of the STRAGL code for these energy ranges and target thicknesses, see [Moo89].

The straggling distribution gives the probability of a projectile's having a particular energy after traveling through a given thickness of material. Thus, a STRAGL calculation using the target thickness yields a distribution of energies after the particles have exited the target material. However, what we desire is the distribution of energies at the time of the nuclear collision and, for calculational purposes, scattering near the front of the target is just as likely as scattering in the back. Therefore, another program, TS, has been developed by Mooney [Moo89] to provide a better approximation to the target energy resolution function. Relying heavily on the STRAGL subroutines for calculating Landau and Vavilov functions, TS allows the user to partition the target into figmental sections, calculate the straggling for each, and sum the resulting normalized distributions to yield the final resolution function. This process is illustrated by figure [3.4]. For our

[10] For compound targets or those with impurities, we can assume that the total straggling is the same as the straggling in the two separate single elements [Chu78].

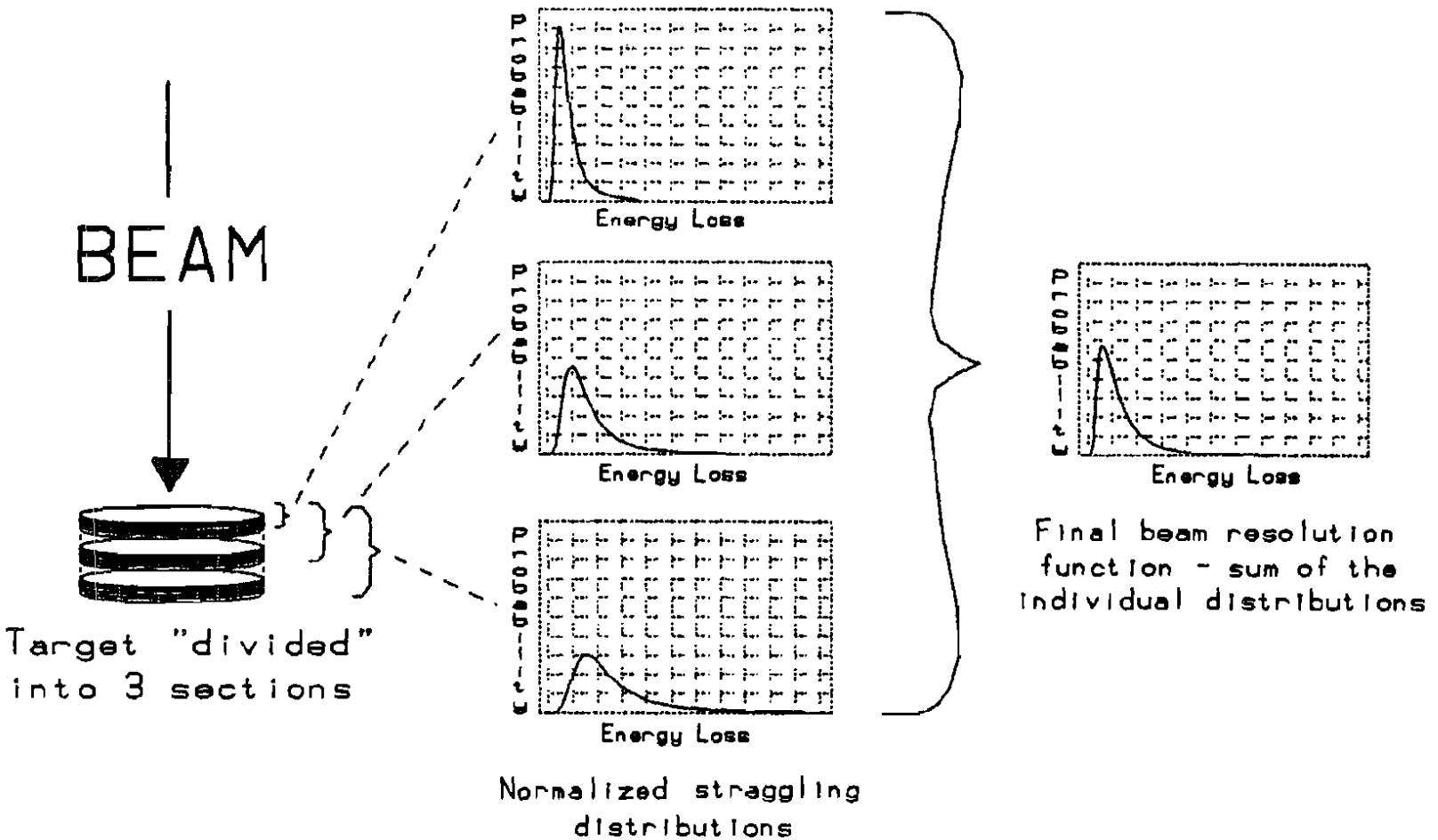


Figure [3.4]: Conceptualized drawing of the calculation of the target straggling distribution.

target thicknesses, division of the target into two or more pieces gives essentially the same result, which is markedly different from the simple STRAGL calculation (figure [3.5]). A resolution function for each target is calculated in this manner, in all cases using three target segments.

3.3 Doppler Broadening

For the final piece of our resolution function puzzle, we consider the projectile ion at the time of the nuclear collision. In many nuclear scattering experiments the target nucleus is thought of as being at rest. However, the nuclei always have some thermal motion resulting from the vibration of the atoms about their lattice sites. At first glance, the scale of the atomic vibrational energies, kT, would appear to render this effect negligible, but it is the relative velocity, not energy, which must be considered. Consequently, as we shall see, the effect is generally not at all negligible for our high resolution studies. The name given to this energy spreading mechanism, Doppler broadening, emphasizes its velocity-additive origins.[11]

Qualitatively, it is easy to see that a target's moving away from a projectile will reduce the energy available in the center of mass while a target's moving toward the projectile will provide added energy to the scattering. Furthermore, since the atomic vibrations are random, we expect the resulting energy loss distribution to be symmetric about an average value of zero.

To evaluate the Doppler broadening (following the treatment of [Lyn68]), we can relate it to the kinetic energies of the constituents. Let us first note that the relative

---

[11] Although technically the Doppler broadening is the result of scattering and not part of the beam energy spread, it affects the center-of-mass energy and thus the cross section. Like the beam resolution function, it can be accounted for through convolution of the Doppler function with the scattering cross section.

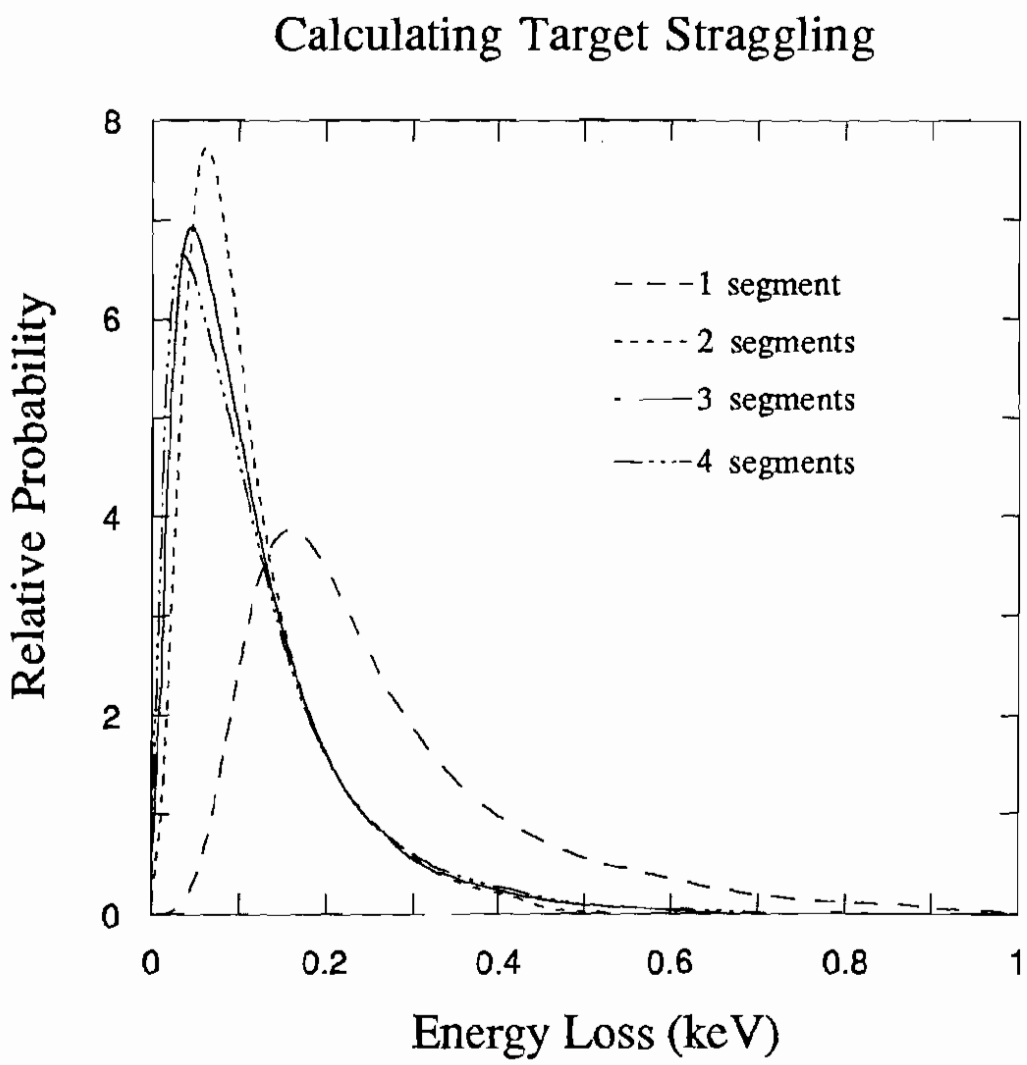


Figure [3.5]: Typical target straggling calculations by the program TS for different numbers of calculational "slices." The one segment distribution represents a simple straggling calculation for the entire target.

energy of the projectile and the target is given by

$$E_{rel} = \frac{1}{2}\mu |\mathbf{w} - \mathbf{v}|^2 , \qquad [3.1]$$

where μ is the reduced mass, $\mathbf{w}$ the velocity of the projectile, and $\mathbf{v}$ the velocity of the target nucleus. Taking the incoming beam in a particular direction, call it the x-axis, we have

$$E_{rel} = \frac{1}{2}\mu\left[(w - v_x)^2 + v_y^2 + v_z^2\right] . \qquad [3.2]$$

For projectile velocities >> thermal, we can neglect the terms in v_i^2 and we get

$$E_{rel} = \frac{1}{2}\mu(w^2 - 2wv_x) . \qquad [3.3]$$

Within the above approximation, the projectile energy, E_p, is the sum of the relative energy in the center of mass, E_{rel}, and the energy of the center of mass, E_{cm}, and we can write the following expression for the velocity of the target nucleus:

$$v_x = \frac{1}{\mu w}\left(E_{cm} - \frac{m_p}{M}E_p\right) , \qquad [3.4]$$

where m_p is the mass of the projectile and M is the sum of projectile and target masses $(m_p + m_t)$.

Classically, the velocity distribution of the target nucleus is given by the Maxwell-Boltzmann formula[12],

$$p(v_x)dv_x \propto e^{\left(-m_t v_x^2/2kT\right)}dv_x \qquad [3.5]$$

which, combined with equation [3.4], yields a nearly Gaussian energy distribution with variance,

$$\sigma^2 = 2\frac{\mu}{M}E_p kT . \qquad [3.6]$$

Quantum mechanically, the classical mean energy per mode of oscillation, kT,

---

[12] This approximation, derived for nuclei in a perfect gas, is also valid for a system of harmonic oscillators.

must be replaced by the quantal mean energy of the target, $\bar{E}_t$, and so the normal distribution we seek has variance,

$$\sigma^2 = 2\frac{\mu}{M}E_p\bar{E}_t \qquad [3.7a]$$

and a corresponding full width at half maximum, Γ, of

$$\Gamma = 4\sqrt{\ln 2\frac{\mu}{M}E_p\bar{E}_t}\ . \qquad [3.7b]$$

We can easily generate this function if we can evaluate the mean energy, $\bar{E}_t$, which is given by:

$$\bar{E}_t = \int_0^{\omega_m} g(\omega)\frac{\hbar\omega}{2}\coth\left(\frac{\hbar\omega}{2kT}\right)d\omega \qquad [3.8]$$

where ω_m is the maximum allowed frequency. (For a derivation of expression [3.8], refer to appendix B). The density of states function or phonon density spectrum, $g(\omega)$, is generally very complicated but can be fairly well approximated for monatomic targets at a particular temperature by the models of Einstein and/or Debye [Ash76].[13] Within these models, only the so-called acoustic modes of vibration are considered (there are only acoustic modes for monatomic materials). By approximating the functional form of the acoustic branches of the dispersion curve, the mean quantal energy for an elemental target can be estimated from a knowledge of the Debye temperature of the solid and the actual temperature of the target, T_t.

For polyatomic (molecular) target materials there exist, in addition to the acoustic branches, optical branches of the dispersion curve which indicate other high frequency modes of oscillation. Because these modes provide additional vibrational energy, approximations based solely on acoustical branches might be considered to be too low. However, the energy is shared between the constituents in unknown ways, and one cannot

[13] This treatment is sometimes known as the Lamb theory [Mor79].

determine how much energy is to be attributed to any particular atom of the compound.

3.3.1 Estimating Target Temperature

In order to provide a value of the target temperature, T_t, it is necessary to make some simplifying assumptions regarding the energy absorbed and lost by the target:

1) Energy deposited by the beam is given by the number of particles incident multiplied by the average energy loss per particle.
2) We ignore energy transfers to and from the target because of conduction and thus assume only radiative energy losses. This is justified because the target itself is very thin (several hundred angstroms).
3) For the final value, we consider all radiation as blackbody (emissivity = 1).

We emphasize that these assumptions, along with the difficulty in estimating some of the needed factors such as the area of the beam spot, make the determination of the target temperature susceptible to fairly large errors.

The problem now becomes a simple energy balance equation -- at some equilibrium temperature the energy in must equal the energy out. The energy deposited by the beam is given as:

$$E_{beam} = \frac{I\varepsilon}{Z} \qquad [3.9]$$

where I is the beam current, ε is the average energy loss per particle, and Z is the number of elementary charges for the incident particle. Of course there is also energy radiated into the target from the surroundings, and this is governed by Stefan's (or the Stefan-Boltzmann) law:

$$R \leq \sigma T_s^4 \qquad [3.10]$$

where R is the radiant emittance, σ is the Stefan-Boltzmann constant (5.6697×10^{-8}

IV. TARGET MAKING

The art of making of "good" thin targets for these measurements proved to be a very important and non-trivial undertaking and as such deserves careful delineation. Here we present specifics for particular targets as well as general procedures and suggestions which will hopefully benefit those trying to progress along similar lines.

Detailed descriptions of target making procedures are far enough removed from the goals of the present study that their inclusion must, at best, be regarded as a slight digression. However, good targets are an essential commodity and probably the single most important factor in the success or failure of a high resolution experiment.[1] The importance of the entire subject of thin film fabrication is evidenced by the wide variety of literature available, spanning the range from solid state to nuclear physics. A couple of the more general references relating to thin films as targets for nuclear physics are [Max67] and [Mug79].

In any case, the nature of the subject matter will allow this chapter to more or less stand on its own. The reader interested only in "real" nuclear physics may safely skip this chapter and naively assume that the targets were "gifts" from a higher power. This reader must, however, keep in mind that the targets are not at all perfect and may be the cause of some irregularities of the data. On the other hand, the reader interested only in target making may use this chapter alone, comforted by the fact that he will encounter only peripheral references to the target's nuclear physics mission.

---

[1] The $^{24}Mg(\alpha,\alpha)^{24}Mg$ experiment was repeated three times, mainly for the promise of a better target.

4.1. Target Requirements -- What is a good high resolution target?

Actual targets for high resolution excitation function data must be as thin as possible (generally on the order of 1 $\mu g/cm^2$) to reduce energy spread contributions.[2] Of course, there is a trade-off between thinness and count rate, and some of the scattering cross sections are known to be very low (e.g. <1 mb at back angles for $^{24}Mg(\alpha,\alpha)^{24}Mg$). It is important, however, that the energy spread produced by the target does not dominate the total experimental energy resolution. Thus, the aim for target thicknesses was to make the target energy spread approximately equal to the intrinsic energy spread of the beam (roughly $\lesssim$1 keV).

Concerns about thickness relate not only to the desired target material but also to various contaminants since their presence also widens the target straggling distribution. Chemical contaminants with high atomic number, and therefore many electrons, are especially damaging and should be avoided whenever possible. Note that contaminants affecting the energy resolution are those which are present in the evaporated material (e.g. as other elements in a compound), those mixed in as a result of the evaporation procedure, or those which somehow accumulate on the target surface. Any form of backing material is exempt from this type of consideration since it only affects the energy of the particle after the scattering event.[3] However, as with any target, one must avoid chemical contaminants, on, in, or behind the target, which would interfere with the

---

[2] The usual measure of target thickness is the density times the actual thickness (sometimes called the areal density), since this quantity better represents the scattering properties of the target. For a target with density 2 g/cm^3, a 1 $\mu g/cm^2$ target corresponds to an actual thickness of 50 Å.

[3] The backing can affect the target resolution "indirectly" by contributing to heating of the target and thus influencing the Doppler broadening contribution. This type of "second order" resolution effect, along with concerns about ADC dead time from high counting rates from the backing material, indicates that the backing material should be as thin as practical.

identification and selection of the peak of interest in the exit energy spectrum. For inelastic scattering, kinematic calculations tell us what to expect in the energy spectrum. For elastic scattering, dangerous contaminants include those whose atomic numbers are very close to that of the primary target atoms. This reasoning implies that the target must also be *isotopically pure*. That is, the material must consist primarily of nuclei with the same atomic weight. As we will see, in several cases nature provides us with these monoisotopic materials, but in other instances the material must be specially ordered to conform to isotopic purity standards. All of the target materials used in these experiments have very high chemical (>99 %) and isotopic (>99.9 %) purity.

Another important consideration in producing good high resolution targets is uniformity. A very nonuniform target, one with separate "clumps" of material, will have very poor resolution. Therefore, even though the measured average thickness may be quite acceptable, most of the target nuclei responsible for the scattering will be in the thicker regions, and the overall resolution will be nearly as bad as for a uniformly thick target. The homogeneity seems to be a function of how well the target material adheres to or "wets" the backing material. Unfortunately, aside from trying different backing materials and trying to keep them as clean as possible, we have little control over target uniformity.

4.2 Target Preparation

All of the targets used in the final analysis are produced in high vacuum (~ 10^{-6} torr) by evaporation onto thin (~ 5 $\mu g/cm^2$) carbon foils. These commercially produced carbon foils are floated from slides onto 1.90 cm diameter stainless steel or aluminum rings with 0.635 to 0.952 cm holes. The rings are placed in a specially designed holder about fifteen to twenty centimeters above the material to be evaporated. A bell jar covers

the entire setup, and the system is pumped down to ~1×10^{-6} torr using an oil diffusion pump with liquid nitrogen cold trap. The "boat" containing the material to be evaporated is then raised to the necessary temperature by resistive heating. During the initial heating time, before the material evaporates, the carbon substrate is protected from direct exposure to outgassing products by an aluminum shutter. When the material is hot enough to evaporate, the shutter is removed until the proper thickness is obtained. The substrate is always covered from the top to prevent any material from getting on the wrong side of the target.[4]

The thickness of the evaporating material is monitored by an oscillating quartz crystal deposition meter. However, for the very thin coatings desired, this instrument is found to be of very little use. Since we have no other reliable method of measuring material deposition at the time of fabrication, most evaporations relied on previous experience and luck for producing proper thickness targets. Luck was supplemented by quantity as various methods were employed for exposing different targets to the evaporating material for differing amounts of time. In this manner, for a constant evaporation rate, targets with a range of thicknesses are produced. Normally, several separate evaporations are performed in order to provide a full range of target thicknesses. Because it is essentially a guessing game, producing targets of the proper thickness is probably the biggest problem encountered in the fabrication of high resolution targets.

Still under vacuum in the bell jar, the targets and evaporator hardware are allowed to cool down before the system is let up to atmospheric pressure. The system is allowed to pressurize slowly to prevent air currents from breaking the fragile targets. For targets which oxidize readily, the bell jar may be filled by an inert gas such as argon. In any

---

[4] This would be particularly disastrous for high resolution measurements since the energy of particles that would scatter from the back would be lessened by virtue of the energy loss in the carbon foil.

case, once the bell jar is removed, the targets are quickly transferred to a vacuum desiccator for storage and travel to the laboratory.

Since the making of each specific target requires somewhat different procedures and equipment, each will be described separately. First the making of the sodium targets will be described since they were by far the easiest to produce. Following that will be a discussion of the phosphorus target preparation which is more complicated but still straightforward, thanks to those who preceded me. Lastly, I will try to describe the notoriously difficult task of making high quality, thin magnesium targets.

4.2.1 ^{23}Na Targets

Sodium is a soft, bright silvery alkali metal which reacts violently with water. Although it does not normally ignite in air below 115 °C, its volatile nature makes elemental sodium a poor choice for a target material. Fortunately, there exist many stable and readily available sodium compounds. Furthermore, the abundance of ^{23}Na in natural sodium and its compounds is 100%, so one need only be concerned with the chemical purity of the particular substance used in the evaporation.

As noted, one of the major concerns in choosing the compound is the minimization of target straggling effects, and, since interaction with atomic electrons is the major cause of energy loss in the target, constituents with low atomic numbers are preferable. Also, we would like to maximize the percentage of sodium in the composite material. One obvious choice, sodium chloride, has the additional advantages of being very inexpensive, readily obtainable in very pure form, very safe to work with, and easily evaporated at fairly low temperatures (melting point of 800 °C). One possible drawback to the use of sodium chloride is the fact that it is subject to evaporation from the heat produced from high beam currents ($\geq$ 3 μA in this thickness range) [Van85]. This is not a

problem for our experiment as the usual beam currents are generally about 1 μA.

Fabrication of ^{23}Na targets, by evaporating chemically pure NaCl onto thin carbon foils, is simple and straightforward. Because of the low cost and isotopic purity of the compound, the "practice" runs could use exactly the same material and setup as the final evaporation.[5] The sodium chloride is heated in a tantalum "dimple" boat to a temperature of about 1000 °C (orange boat).[6] Since the material is a powder, the heat transfer with the boat is not ideal, and one must be careful not to raise the temperature too fast. When heated rapidly, the material will "pop," throwing macroscopic pieces from the boat. When heated slowly, the crystals appear to fuse together and then become smaller and smaller as the evaporation proceeds. Even though there is an obvious decrease in material in the boat, at the desired thicknesses, little or nothing can be seen on the targets or the glass slides inserted as visual monitoring devices. However, when any of the substrate material is exposed to air for an hour or more, a white haze becomes noticeable, especially on the glass slides. This effect can be enhanced and the slides can be made almost opaque by breathing on them. Presumably this is due to moisture absorbed by the material. In any case, since it is standard procedure to quickly remove the targets to a desiccator, any water absorption should be minimal.

4.2.2 ^{31}P Targets

Phosphorus is a non-metallic element and falls into group VA of the periodic

---

[5] Actually this is not entirely true, since, for the final evaporations, we used sodium chloride with enriched ^{35}Cl. Unlike sodium, chlorine has many isotopes, and, although probably an unnecessary precaution, using $Na^{35}Cl$ simply gives us fewer reactions (from chlorine isotopes) to worry about.

[6] Once the evaporation begins, the rate can be effectively controlled by raising or lowering the temperature.

table. ^{31}P is the only stable isotope of phosphorus and occurs naturally in several different forms depending on the crystal structure and solid state properties. *White* or *yellow* phosphorus is crystalline and ignites spontaneously in air. We are most interested in *red* phosphorous which is a violet-red, stable, amorphous powder. Red phosphorus is inexpensive, readily obtainable in very chemically pure form, and fairly safe to work with if not heated in the presence of oxidizing materials. Red phosphorus is nearly insoluble in most solvents (water, CS2, ether, ammonia) and sublimes at 416 °C.

Making phosphorus targets by evaporation presents problems because, at temperatures below about 800 °C, the vapor exists as P_4 molecules and will condense as white phosphorus. In this case, unless extreme measures are taken to prevent the target's contact with air, plans for the experiment will, like the target, go up in smoke. This problem is addressed and resolved in a paper by Hooten [Hoo64]. Basically, if the phosphorus vapor is heated to a temperature above 1000 °C, then the ensuing dissociation of the P_4 molecules allows the vapor to condense subsequently as red phosphorus.

Hooten's method of passing the vapor by a heated tungsten sheet was improved upon by Tzeng *et al.* [Tze78]. The newer method consists of passing the vapor through a heated carbon chimney before it is allowed to contact the substrate. The necessary apparatus, a carbon crucible joined to the chimney by a small pyrex tube, has been manufactured at UNC, and its setup is fairly simple. Separate tantalum wires wrapped tightly around the crucible and chimney provide the resistive heating necessary to produce the required temperatures. Blank feedthroughs in the bell jar base plate are replaced by extra electrodes to accommodate the separate heater wires. An exact description of the apparatus can be found in [Tze78].

With the phosphorus in the crucible and the system under vacuum, the graphite

chimney is heated to 1000 °C (graphite glows orange). [7] With the targets shielded, the crucible is heated slowly until outgassing ceases. At this point the diffusion pump gate valve is closed since phosphorus vapors may contaminate the pump oil. The crucible temperature is raised until phosphorus evaporation occurs (< 800 °C), and the targets are uncovered long enough to obtain the desired thickness. Next, the crucible heater is turned down, and, after a few minutes, the chimney heater is also turned off. The entire system is then allowed to cool for several hours.

Since phosphorus may produce some nasty compounds when heated, the bell jar is filled with air[8] and evacuated again with a mechanical pump vented to the outside. For safety, this "flushing" procedure is repeated several times, and the targets are then transferred to a vacuum desiccator. The evaporator hardware may safely be cleaned up using gloves, soap, and water. Of course, as with the handling of any chemical, proper clothing and eye protection should be worn. For phosphorus spills the recommended action is to carefully sweep up the waste material and burn it in an iron pan under a fume hood.

This method of producing phosphorus targets works very well, and again the only problem is the judgement of the thickness. Visually, the different targets which were produced occurred in a myriad of colors, from purple to green to yellow. However, no

---

[7] In order for the chimney to be at the proper temperature, the tantalum heater wire must be quite a bit hotter since the design of the system allows for relatively poor heat transfer. In general, the heater wire must be white hot, and the current must be momentarily decreased to judge the chimney color. Usually, only the very tip of the chimney is bright orange, and the rest is a dull red. Also, an extra stationary heat shield around the chimney and its heater helps prevent re-evaporation of the phosphorus after it has reached the substrate and partially blocks the tantalum which evaporates from the heater wire.

[8] With the room lights off, one may sometimes see a flash of light from the inside of the bell jar as the air is first let in. This is, of course, due to the presence of some white phosphorus which was not converted to red before condensing or which re-evaporated outside the crucible-chimney apparatus.

definite correlation between target color and thickness was ever found.

4.2.3 ^{24}Mg Targets

Magnesium is a light silvery-white metal in group IIA of the periodic table. It oxidizes slightly in air and has a melting point of about 650 °C. The natural abundance of ^{24}Mg is 78.7% with the remainder being divided almost equally between the isotopes with $A = 25$ and 26 (10.13% and 11.17% respectively). Enriched ^{24}Mg (to >99.9%) is most readily available in oxide form (MgO) from Oak Ridge at a moderate price (about $0.75/mg). It is this form which constitutes the starting point for the procedures which will be related here. Because of its very inexpensive price, natural magnesium oxide was utilized for the many practice runs with the enriched ^{24}MgO employed for making the usable targets. The chemical properties are identical, and the descriptions and procedures related pertain to either the natural or enriched elements and compounds.

Magnesium oxide (also magnesia) is a very fine white powder which is insoluble in alcohol and only slightly soluble in pure water. It has a very high melting point of about 2800 °C and is fairly safe to work with under the usual chemical precautions. Because of the high melting point of MgO, we find ourselves working with a compound which we cannot evaporate by our standard resistive heating techniques. However, since oxygen is really an unwanted contaminant, we can use standard techniques to break apart the MgO molecule and thus allow the magnesium to evaporate at a reasonably low temperature. The process of removing oxygen from an oxygen containing compound is known as *reduction*[9] and is related to the constituents' affinity for electrons. In this case,

---

[9] This definition of reduction is a bit antiquated since the meaning has been broadened to include non-oxide compounds by defining an *oxidation number* for elements in compounds. Any element which undergoes an algebraic decrease in oxidation number is said to be reduced.

reduction of the MgO is brought about by heating the oxide together with a suitable *reducing agent*. Basically, the reducing agent, having a greater affinity for oxygen, steals it from the nearby compound and forms its own oxide. The reduction of MgO takes place at temperatures above the magnesium melting point and thus, after being freed from its oxide compound, the metal evaporates immediately. Standard reducing agents include zirconium, lanthanum, and tantalum.

There is much written [Tak66] [Wil82] and related [Wes84b] [Loz84] about how to reduce magnesium oxide and evaporate the magnesium metal, and many techniques were attempted with varying degrees of success. Over a period of several years literally hundreds of Mg targets were produced, and, while some target making methods were found to be superior to others, there seems to be no absolutely foolproof method of producing thin, uniform magnesium targets. The problems in making magnesium targets may be broadly grouped into two categories which we will examine in the following sections. The first section describes the techniques of reducing MgO and evaporating magnesium metal, while the second section is concerned with the condensation of the magnesium vapor onto a substrate as a thin film suitable as a target in our high resolution experiment.

4.2.3.1 Evaporating Magnesium Metal

Our early attempts at making magnesium targets followed the procedures and techniques of [Wil82] and [Tak66], where the MgO is mixed with zirconium powder and placed in a closed tantalum boat.[10] Using these methods, we found the results to be very

[10] The closed boat is produced from a piece (~ 8 centimeters long) of 0.635 centimeter diameter tantalum tubing. A small hole (~ 15 mm) is drilled through one wall in the middle of the tubing. Then one end is closed by crimping with pliers or a vise. Next, the mixture is loaded in the open end which is subsequently flattened to completely enclose the material.

inconsistent. The boat temperature tends to vary in each evaporation if evaporation occurs at all. This inconsistency was usually blamed on the thermal contact of the boat and the material and/or the contact between the MgO and the zirconium. The MgO and fine zirconium powder were mixed together in atmosphere before being loaded into the boat. Sometimes pure alcohol was introduced into the boat to aid in thermal conductivity, but it was unclear whether or not this procedure was beneficial.

In retrospect, it seems likely that the fine zirconium powder (-80 mesh) was oxidized from exposure to the atmosphere and the reduction, if it occurred, was initiated by the tantalum boat itself. This would explain some of the null results when using a boat for more than one evaporation. Also, this would indicate the importance of the contact of the material with the surface of the reducing agent.

With this in mind, we tried a method which had been used by another experimental group at TUNL [Wes84b]. This method involves mixing the MgO with distilled water to form a slurry which is then painted on a boat made of a suitable reducing agent (we tried tantalum and tungsten). We tried different thicknesses of paint and different methods of drying, but again inconsistency ruled. Many times the MgO only cracked and flaked off of the hot boat.

Finally, after many attempts using different sizes, shapes and materials for boats, we returned to the closed tantalum boat because of its promise of even heating, and, using a method suggested by W. R. Lozowski [Loz84], we were able consistently to reduce MgO and to evaporate Mg metal. This method required Lanthanum as the reducing agent. Lanthanum shavings, obtained from filing or cutting a lanthanum ingot, are combined with the MgO and pressed tightly together into a small pellet.[11] A special press was manufactured to produce pellets of a size to fit into our closed tantalum boats.

---

[11] In general, we found that larger shavings or slivers (we used wire cutters to trim the ingot) worked better than small pieces produced by filing.

Furthermore, since lanthanum oxidizes rapidly in air, the shaving of the lanthanum, the packing with MgO, and the loading into the boat are all accomplished in a glove bag under an argon atmosphere.[12]

The boat and its contents are then transferred to the bell jar for the evaporation. Being almost completely enclosed by the boat, the MgO and lanthanum mixture are protected from oxygen during the transfer and subsequent readying of the evaporator apparatus. Once in high vacuum, the material is heated to about 1000 °C (orange boat)[13], and the reduction and evaporation usually occur fairly rapidly. Even though this method seems to provide a relatively consistent means of evaporating magnesium metal, the ineffectiveness of the quartz crystal thickness monitor makes obtaining the proper target thickness very difficult. Indeed, the targets produced by this method for use in our scheduled final experiment were found to be either too thick or too thin, and more targets had to be prepared during the experimental running period.

Ultimately, the targets used for the final data were produced at Duke University using much the same methods as our earlier attempts [Lem87]. That is, the procedure utilized a closed tantalum boat in which MgO is mixed with a suitable reducing agent. However, in this instance, tantalum powder, instead of zirconium, was used as the reducing agent. Some targets produced in this fashion were found to be badly contaminated with tantalum. However, with care taken to hold the boat temperature below the evaporation point of the reducing agent, very good magnesium targets were produced with only minimal contamination.

---

[12] Several attempts were made using fresh zirconium powder, but it did not stay formed in the pellet as well as did the lanthanum.

[13] For this type of reduction and evaporation, in order to prevent contamination, one needs to provide temperatures below the boiling point of the reducing agent. For lanthanum (at 10^{-6} torr) this temperature is about 1200 °C.

4.2.3.2 Magnesium Condensation -- Preparing the Substrate

During the time when we were searching for a highly reproducible procedure to evaporate Mg from MgO, we were also aware of perhaps a more severe problem which has been noted by [Wil82] and many others. When magnesium is evaporated, it does not stick very well to carbon. Furthermore, when it does adhere, it often seems to do so in patches, resulting in very non-uniform targets. Although we cannot directly measure the target surface profile, we have observed (and others documented [Iko79]) poor energy resolution which is blamed on the target non-uniformity.[14] At this point we can only speculate about the causes of this erratic adhesion behavior.

Since Mg has a relatively low melting point, it seems feasible that the metal might re-evaporate from "hot spots" on the carbon foil and condense on cooler surfaces. Adding credence to this theory is the fact that the magnesium seemed often to accumulate around the edges of the carbon foil -- areas that would benefit most from heat conduction to the substrate holder. In an attempt to provide a better heat flow away from the target, we designed and built a special copper substrate holder which can be water or liquid nitrogen cooled. Even though the targets are held in place by this huge heat sink, the magnesium metal still does not adhere well to the middle of the carbon foils. Perhaps the thin foil does not allow enough heat conduction at any surrounding temperature, or perhaps it is the carbon surface itself which is somehow to blame. In any case, cooling the substrate did not seem to improve considerably the sticking of the magnesium, and the

[14] As a uniformity test, we measured the target thickness at different points on one target by raising and lowering the target by small amounts. This test would be sensitive to rather large scale variations (several mm) over the surface, and little variation was found. Also, an attempt was made to look at (an arbitrary) one of the magnesium targets under a scanning electron microscope. Although indications were of a very non-uniform surface, charging problems prevented any detailed or definite determinations of the surface features.

practice was abandoned for all of the later evaporations.

Noting that magnesium sticks slightly to the copper substrate holder and very well to the aluminum foil covering some of the evaporator hardware, we decided to provide several different backing materials upon which the magnesium could be deposited. In separate evaporations we coated some of the carbon foils with various metals -- copper, silver, gold, bismuth, and aluminum.[15] The copper, silver, and bismuth coatings seem to produce no difference in the sticking of the magnesium metal, although the bismuth-backed targets were never fully tested. The gold coating seemed to make the situation worse, with almost no magnesium present on the gold layered targets.

As expected, the magnesium seemed to adhere well to the aluminum coated foils, and these types of targets were used for some of the preliminary data. We acknowledge that the surface of this backing material is almost certainly aluminum oxide (Al_2O_3) since the backings were exposed to air when transferred to and from the evaporator. At this point we cannot say if the apparent improvement in adhesion is the result of the physical properties of the Al_2O_3 surface, characteristics of the Al bulk material, or simply the condition of the surface (cleanliness) because of the way it is prepared. Also, it is questionable whether or not the aluminum provides a significant improvement in target uniformity. We can be sure, however, that the addition of a backing substance whose atomic weight is close to that of magnesium produces problems in peak selection in the exit energy spectrum at forward angles (< 60 degrees).

Finally, we note that, although the solution to the magnesium sticking problem was never found, the aluminum backing material seemed to be a step in the right

---

[15] We used copper and aluminum since we had noticed some magnesium adhesion to the evaporator hardware. Bismuth, gold, and silver were used because they are easily obtained, stable, and simple to evaporate. (Also, bismuth had been suggested by [Loz84].) Note also that the coefficient of linear expansion for aluminum is the same as that of magnesium ($25 \times 10^{-6}\ K^{-1}$).

direction. In any case, the final targets were made by direct deposition on carbon foils, and the energy resolution indicates that the targets are probably fairly uniform.

4.2.3.3 Sputtering

As a final note to magnesium target making, the sticking problem of Mg metal to carbon led us to seek other possible forms for the target. The acquisition of hardware for producing targets by heavy ion sputtering provides the means of producing targets from almost any substance and gives us the opportunity to try an almost completely different approach in making a magnesium target. In particular, direct sputtering of MgO produces targets which seem to be fairly uniform and whose thickness can be fairly accurately controlled. However, our single attempt in using these MgO targets found them to be severely contaminated by silicon. The energy straggling produced by the silicon as well as that from the oxygen rendered these targets unusable for our high resolution work.

4.3 Conclusions and Suggestions for Future Work

In general, the biggest problem was the thickness monitor. Most times it failed to register any deposition of material, and, when it seemed to work, the results were unreliable. All our efforts in cooling, shielding, and cleaning were in vain. It seems that no amount of work can make this type of thickness monitor very useful in our desired thickness range. Although I have no specific recommendations, this problem should be carefully considered before these types of targets are fabricated by these methods.

In the final analysis, sputtering may be the answer for making all targets. The procedures for all materials are almost exactly the same, and one should be able to calculate approximate thicknesses from beam intensities and sputtering rates. At the time of most of these experiments, the sputtering apparatus was not available at UNC. Some

sputtered targets were used (section 4.2.3.3) but were found to be contaminated by silicon. Recent replacement of the oil diffusion pump with a cryopump should eliminate this problem. Once some experience is gained in making sputtered targets, I can see no real disadvantages of the sputtering method for fabricating thin targets of any material.

4.3.1 Sodium and Phosphorus

The evaporation methods for sodium and phosphorus worked well. Some extra effort should be made in the phosphorus evaporation to provide better thermal contact between the tantalum heater wires and the carbon crucible and chimney. This would almost certainly help remedy the problem of contamination from tantalum evaporating from these same wires. Except for sputtering, as mentioned above, I see no possibility of a major improvement in the fabrication of these targets.

4.3.2 Magnesium

Although coming up with few definite answers, I gained much experience and some insight from the many trials and tribulations of magnesium target making. For anyone who cannot get someone else to produce his/her magnesium targets, I have the following suggestions:

1) When reducing MgO, care must be taken to insure that the reducing agent is oxide free. Obviously, an agent which is already oxidized cannot do the job. Also, I reiterate that, to avoid contamination, the boat temperature must be kept below the evaporation point of the reducing agent.
2) Recent improvements in the sputtering system have solved the silicon contamination problem and suggest MgO sputtered targets could be reconsidered. Given the known difficulties with the reduction and evaporation

procedures, sputtering should be high on the list of anyone desiring a magnesium target.

3) The current availability of isotopically enriched magnesium metal implies that the reduction problems can be eliminated entirely. Evaporation of magnesium metal is very easy and straightforward and would almost certainly be more convenient.

4) If the charging problems could be overcome, the electron microscope (EM) may be a useful diagnostic tool for examining the target uniformity. It has been suggested that a some sort of special "target" with a conductive backing be included in the bell jar during the evaporation process. Of course this target could only be used for the EM, but it might provide some insight into the way the material is deposited during a particular evaporation.

5) A final suggestion is to try to "clean" the carbon foil surface before the material is deposited on it. One way may be to use a heavy ion beam to sputter away any contaminants and smooth the carbon surface. This method has been suggested by Crosson [Cro88] and used successfully in depositing gold onto a diamond surface [Chi89].

V. COMPARING THEORY WITH DATA

The theory of the scattering of two particles with spin may be envisioned in two parts: the so-called "formal" scattering theory with formulae based on the Schrödinger equation and the various model-dependent approximations which attempt to reproduce the observables involved in nuclear scattering. There are several excellent developments of formal scattering theory (e. g., [Mer70] [Tay83]), and we will not compete with the descriptions found there. Although we strive to remain as general as is practical, our concern focuses on adapting and applying the results of this formal framework to the derivation of formulae which lend themselves to comparison with the particular experimental data at hand.

The first part of this chapter involves the derivation of an expression for differential cross section for proton and alpha scattering through a $J^{\pi} = 0^{+}$ state in the compound system. The energy dependence is modeled by an isolated Breit-Wigner resonance interfering with an energy independent background. Although some of the formulae may appear complicated, much of the apparent complexity is the result of the angular momentum coupling[1] and the algebra involved therein. The remaining sections are devoted to the means of actually calculating the cross sections and comparing them with the experimental data. These discussions include the incorporation of the experimental energy resolution, the quest for a convenient form of the equations for efficient computer evaluation, and an outline of the programs utilized to determine the best values of the model parameters.

---

[1] We note that there are several schemes for coupling the angular momenta and we have chosen (more-or-less arbitrarily) to couple via a channel spin representation.

5.1 Theory

In general[2], the differential cross section for the scattering of two particles is given by[3]:

$$\frac{d\sigma}{d\Omega} = \sum_{\substack{M_a, M_b \\ M_A, M_B}} \frac{1}{(2I_A + 1)(2I_a + 1)} \left| f_c(\theta)\, \delta_{\beta\alpha} \delta_{M_B M_A} \delta_{M_b M_a} + A_{\beta M_B M_b, \alpha M_A M_a}(\theta, \phi, E) \right|^2 \quad [5.1]$$

with spins, I, and projections, M, where the uppercase subscripts denote the target before and after (A and B), and the lowercase subscripts refer to the projectile before and after (a and b) the scattering event. The Greek subscripts, α and β, represent the incident and exit channels respectively. Also, f_c is the Coulomb amplitude, and $A_{\beta,\alpha}$ is the transition amplitude which, in the channel spin representation, is given by:

$$A_{\beta M_B M_b, \alpha M_A M_a}(\mathbf{k}_\beta, \mathbf{k}_\alpha) = \frac{2\pi}{i\, k_\alpha} \sum_{\substack{l_\alpha, l_\beta, S_\alpha \\ S_\beta, j, m_\alpha}} \langle I_A I_a M_A M_a \,|\, S_\alpha M_\alpha \rangle \langle l_\alpha S_\alpha m_\alpha M_\alpha \,|\, j M_j \rangle \langle I_B I_b M_B M_b \,|\, S_\beta M_\beta \rangle$$

$$\langle l_\beta S_\beta m_\beta M_\beta \,|\, j M_j \rangle\, e^{i(\sigma_{\alpha, l_\alpha} + \sigma_{\beta, l_\beta})} \left(S^j_{\beta l_\beta S_\beta, \alpha l_\alpha S_\alpha} - \delta_{\beta\alpha} \delta_{l_\beta l_\alpha} \delta_{S_\beta S_\alpha} \right) Y^{m_\alpha *}_{l_\alpha}(\hat{k}_\alpha)\, Y^{m_\beta}_{l_\beta}(\hat{k}_\beta) \quad . \quad [5.2]$$

Here, k is the wave number, S the channel spin, l the relative orbital angular momentum, j the total angular momentum, S^j the scattering matrix element, and the σ's are Coulomb phase shifts.

We choose the incident beam direction as the positive z-axis (i.e. $\mathbf{k}_\alpha$ along $\hat{\mathbf{z}}$) so that

---

[2] These first two general equations are adapted from [Sat83] (equations 5.37 and 5.36) and roughly follow the notation therein.

[3] This equation holds true for unpolarized incident particle beams. In particular, we recognize the usual "average over initial states and sum over final states" nature of the equation.

$$Y_{l_\alpha}^{m_\alpha *}(\hat{k}_\alpha) = Y_{l_\alpha}^{m_\alpha *}(\theta,\phi) \rightarrow Y_{l_\alpha}^{m_\alpha *}(0,0) = \sqrt{\frac{2l_\alpha+1}{4\pi}}\,\delta_{0m_\alpha}\,, \qquad [5.3]$$

and, since the cross section is now independent of the azimuthal angle, ϕ, we have

$$Y_{l_\beta}^{m_\beta}(\hat{k}_\beta) = Y_{l_\beta}^{m_\beta}(\theta,\phi) \rightarrow Y_{l_\beta}^{m_\beta}(\theta,0) = (-1)^{m_\beta}\sqrt{\frac{2l_\beta+1}{4\pi}}\sqrt{\frac{(l_\beta-m_\beta)!}{(l_\beta+m_\beta)!}}\,P_{l_\beta}^{m_\beta}(\cos\theta)\,. \qquad [5.4]$$

For further simplification, we note that for an isolated resonance in the presence of a nearly energy independent background, there is only one term in the above sum [5.2] that is strongly energy dependent. This resonance term has particular values of l_α, l_β, and j which we shall denote by L_α, L_β, and J. Separating this term and factoring out the Coulomb phase shifts, we have, for the resonant part of the scattering amplitude:

$$A^{(res)}_{\beta M_B M_b,\alpha M_A M_a}(\theta,E) = \frac{1}{2ik_\alpha}\sum_{S_\alpha,S_\beta}\langle I_A I_a M_A M_a | S_\alpha M_\alpha\rangle\langle L_\alpha S_\alpha 0 M_\alpha | JM_j\rangle\langle I_B I_b M_B M_b | S_\beta M_\beta\rangle\langle L_\beta S_\beta m_\beta M_\beta | JM_j\rangle$$
$$(-1)^{m_\beta}\sqrt{2L_\alpha+1}\sqrt{2L_\beta+1}\sqrt{\frac{(L_\beta-m_\beta)!}{(L_\beta+m_\beta)!}}\left(S^J_{\beta L_\beta S_\beta,\alpha L_\alpha S_\alpha}(E)-\delta_{\beta\alpha}\delta_{L_\beta L_\alpha}\delta_{S_\beta S_\alpha}\right)P_{L_\beta}^{m_\beta}(\cos\theta) \qquad [5.5]$$

and now the differential cross section is given by

$$\frac{d\sigma}{d\Omega} = \frac{1}{(2I_A+1)(2I_a+1)}\sum_{\substack{M_a,M_b\\M_A,M_B}}\left|F^{(\pi)}_{M_B M_b M_A M_a}(\theta) + A_{\beta M_B M_b,\alpha M_A M_a}(\theta,E)\right|^2 \qquad [5.6]$$

where $F^{(\pi)}$ contains all the "left-over" terms in the sum, the Coulomb phase shift factor, and the Coulomb scattering amplitude. The $F^{(\pi)}$, each a sum over l's, S's, and j, are energy independent[4] background amplitudes and by themselves are uninteresting for our purposes. However, these non-resonant amplitudes cannot be ignored since they are added coherently in the above sum and thus influence the resonance parameters through the "interference" or "cross" terms. Also, in general, the background terms constitute unknown parameters adjusted to fit the data, and as such, we are concerned with the total

[4] Basically, our model considers the background as constant over the range of the data. However, since many of the data sets exhibit a noticeable slope, we allow the non-resonant cross section to have a linear dependence on energy.

number of independent values of these amplitudes (i.e. the number of terms in the sum over the M 's). We note that the M-dependence of the background amplitudes is contained in the Clebsch-Gordan coefficients and the spherical harmonics, and we can use various symmetry considerations to show, under quite general conditions[5] from [5.2], that

$$F^{(nr)}_{M_B M_b M_A M_a} = (-1)^{I_A+I_a+I_B+I_b+n}(-1)^{M_A+M_a-M_B-M_b} F^{(nr)}_{-M_B-M_b-M_A-M_a} \quad \begin{cases} n=0\,, \text{ integer } j \\ n=1\,, \text{ half-integer } j\,. \end{cases} \quad [5.7a]$$

Since the cross section is made up of the squares of the amplitudes, the possible minus sign is, for our purposes, irrelevant; that is,

$$\left|F^{(nr)}_{M_B M_b M_A M_a}\right|^2 = \left|F^{(nr)}_{-M_B-M_b-M_A-M_a}\right|^2 , \quad [5.7b]$$

and this relationship effectively cuts the number of independent amplitudes in half. Note, however, that the amplitudes are, in general, complex quantities and as such would require two real search parameters for each.

Unlike the non-resonant terms, we can simplify the form of the resonant amplitudes from the specifics of the experiments. One thing our measurements have in common is that the resonant part of the reaction proceeds through a state which has $J^\pi = 0^+$. This means that we can substitute $J=0$; $M_J=0$; $S_\alpha = L_\alpha$; $S_\beta = L_\beta$; $M_\alpha = 0$; and $M_\beta = -m_\beta$ into the equation for $A^{(res)}$ and, after simplifying, we get

$$A^{(res)}_{\beta M_B M_b, \alpha M_A M_a}(\theta,E) = \frac{1}{2ik_\alpha}\langle l_A l_a M_A M_a | L_\alpha 0\rangle\langle l_B l_b M_B M_b | L_\beta - m_\beta\rangle$$
$$(-1)^{L_\alpha + L_\beta}\sqrt{\frac{(L_\beta - m_\beta)!}{(L_\beta + m_\beta)!}}\left(S^0_{\beta L_\beta, \alpha L_\alpha}(E) - \delta_{\beta\alpha}\delta_{L_\beta L_\alpha}\right)P^{m_\beta}_{L_\beta}(\cos\theta) \quad . \quad [5.8]$$

Furthermore, we note that the Clebsch-Gordan coefficients enforce the equalities $M_A = -M_a$ and $M_B = -(M_b + m_\beta)$ although we did not explicitly make those substitutions. For

---

[5] We use symmetry of Clebsch-Gordan coefficients and spherical harmonics under a sign change of the M's along with parity conservation such that parity is the same for the initial and final states; i.e. $(-1)^{l_\alpha + l_\beta} = 1$.

notational convenience we separate the energy dependent parts[6] from the angular and simple "geometric" terms by setting

$$g(\theta) = g_{M_\beta M_b M_A M_a}(\theta) = \frac{-1}{2k_\alpha}\langle I_A I_a M_A M_a | L_\alpha 0\rangle \langle I_B I_b M_B M_b | L_\beta - m_\beta\rangle \sqrt{\frac{(L_\beta - m_\beta)!}{(L_\beta + m_\beta)!}}\, P_{L_\beta}^{m_\beta}(\cos\theta) \qquad [5.9]$$

and

$$h(E) = -\frac{1}{i}\left[S^{0}_{\beta L_\beta \alpha L_\alpha}(E) - \delta_{\beta\alpha}\delta_{L_\beta L_\alpha}\right] \qquad [5.10]$$

so that

$$A^{(res)}_{\beta M_B M_b \alpha M_A M_a}(\theta, E) = g_{M_B M_b M_A M_a}(\theta)\, h(E). \qquad [5.11]$$

Since the values of the spins, I, are fixed for a given system (two partitions), and the values of relative orbital angular momentum, L, are fixed for a particular reaction channel, the value of $g(\theta)$ can be easily calculated for a particular scattering angle and set of spin projections (M's).

The real physics, of course, as far as the resonance reaction is concerned, is in the term $h(E)$. For an isolated resonance, this energy dependent term can be represented by the Breit-Wigner single level formula which, to within an arbitrary phase, is given by (see, e. g., [Pre62], chapter 16)

$$h(E) = \frac{\sqrt{\Gamma_\alpha \Gamma_\beta}}{E - E_R + \frac{i\Gamma}{2}} \qquad [5.12]$$

where E_R is the resonance energy, Γ is the total energy width of the resonant state, and Γ_α and Γ_β are the partial widths for the incident and exit channels, respectively. The partial widths, resonance energy, and total width constitute the "resonance parameters" which are adjusted to produce fits to the experimental data. Note that writing this equation in general terms for the incoming and outgoing partial widths allows the description of

[6] Note that we consider k to be approximately constant over the energy range of the resonance.

elastic ($\Gamma_\beta = \Gamma_\alpha$) as well as inelastic ($\Gamma_\beta \neq \Gamma_\alpha$) scattering.

One problem in calculating fits to the cross section arises from the possibly large number of terms in the sum over the spin projections. If each term were treated separately, for systems with many possible combinations of spin projections the relatively high number of unknown parameters (compared to the number of relationships among those parameters) might present a problem for the optimization routine in finding the best solution. It almost certainly would provide a problem in the interpretation of the results. Of course, the relationship [5.7] reduces the number of independent amplitudes by a factor of two, but even for a system whose members all have spin-½, there are still eight independent complex background amplitudes (corresponding to sixteen real search parameters) at each angle, and searching on each individual amplitude remains quite impractical. However, we see that it is neither necessary nor in most cases advantageous to search for the values of each separate amplitude. With our definitions of $g(\theta)$ and $h(E)$, we can write the cross section as

$$\frac{d\sigma}{d\Omega} = \frac{1}{(2I_A+1)(2I_a+1)} \sum_{\substack{M_a,M_b \\ M_A,M_B}} \left| F^{(\mathrm{nr})}_{M_B M_b, M_A M_a}(\theta) + g_{M_B M_b, M_A M_a}(\theta)\, h(E) \right|^2 , \qquad [5.13]$$

and expanding the squared terms we get:

$$\frac{d\sigma}{d\Omega} = \frac{1}{(2I_A+1)(2I_a+1)} \left[\sum_{M'_B} |F(\theta)|^2 + 2\mathrm{Re}\left\{ h(E) \sum_{M'_B} F^*(\theta) g(\theta) \right\} + |h(E)|^2 \sum_{M'_B} |g(\theta)|^2 \right] \qquad [5.14]$$

where, for clarity, we have dropped the subscript notation on F and g, and $h(E)$, which is not dependent on the spin projections, has been factored out of the sums.

The first sum in [5.14], that of the non-resonant amplitudes, may be regarded as one real search parameter which is proportional to the background cross section and therefore strongly constrained by the "wing" or off-resonance data. The second sum, containing the so-called "cross terms" can be treated as one complex parameter, denoted

as F', to be multiplied by the energy dependent resonance term. Note that in some cases the $g(\theta)$ may be calculable, but since they must be multiplied by the unknown F's, it is sensible simply to search for the value of the complete sum. The final sum in [5.14] may be evaluated from the form of the $g(\theta)$. Using equation [5.9], we can write

$$\sum_{M's} |g(\theta)|^2 = \frac{\pi}{k_\alpha^2(2L_\beta+1)} \sum_{M_A M_a} \left|\langle I_A I_a M_A M_a | L_\alpha 0\rangle\right|^2 \sum_{M_B M_b} \left|\langle I_B I_b M_B M_b | L_\beta - m_\beta\rangle\right|^2 \left|Y_{L_\beta}^{m_\beta}(\theta,0)\right|^2 \quad . \quad [5.15]$$

From the orthonormality of the Clebsch-Gordan coefficients and the completeness of the spherical harmonics[7] this reduces to the shamelessly simple[8]

$$\sum_{M's} |g(\theta)|^2 = \frac{1}{4k_\alpha^2} \quad . \quad [5.16]$$

Thus, finally, the cross section can be written:

$$\frac{d\sigma}{d\Omega}(\theta) = \frac{d\sigma}{d\Omega}^{(nr)}(\theta) + \mathrm{Re}\left[h(E)\,F'(\theta)\right] + \frac{1}{4k_\alpha^2(2I_A+1)(2I_a+1)} |h(E)|^2 \quad [5.17]$$

where the F' is a complex search parameter proportional to the sum of $F^*(\theta)g(\theta)$ in equation [5.14]. This equation is very general, providing for three unknown, real, "non-resonant" parameters at each angle, and may be applied to any of the experimental data at hand.[9] In practice, this form is the basis of the calculations used in the analysis of all our experimental data.

Although this aforementioned scheme also applies to the ^{24}Mg(α,α) ^{24}Mg data,

---

[7] Since $m_\beta = -M_B - M_b$, we can rewrite the second sum as a sum over M_B (or M_b) and a sum over m_β. The square of the Clebsch-Gordan coefficient then sums to one (over M_B) leaving the completeness relationship (sum over m_β) for the spherical harmonics.

[8] Note that, in general, it is only the sum of the **squares** of the resonant amplitudes which reduces to such a simple, angle independent form. The amplitudes themselves are isotropic only when the associated exit channel orbital angular momentum is equal to zero ($L_\beta = 0$).

[9] This , of course, under the condition that there is a single resonance state with $J^\pi = 0^+$.

the elementary nature of that system allows us to simplify the cross section calculation in such a way as to have only two real background amplitude search parameters at each angle. The $^{24}Mg(\alpha,\alpha)^{24}Mg$ system contains nothing but spin zero constituents; that is,

$$I_A = I_B = I_a = I_b = 0 \;\Rightarrow\; M_A = M_B = M_a = M_b = 0 .$$

We do not need any of the symmetry relations between the amplitudes, nor do we encounter any complicated sums since there is only one term in the cross section sum over spin projections:

$$\frac{d\sigma}{d\Omega} = \left| F^{(nr)}(\theta) + g(\theta)h(E) \right|^2 . \qquad [5.18]$$

Furthermore, for the resonant portion, $L_\alpha = L_\beta = 0$, and the geometric part of the resonant amplitude, $g(\theta)$, is similarly easy to evaluate as

$$g(\theta) = \frac{1}{2k_a} \langle 0000 | 00 \rangle \langle 0000 | 00 \rangle \sqrt{\frac{0!}{0!}}\, P_0^0 (\cos\theta) = \frac{1}{2k_a} . \qquad [5.19]$$

We see that this amplitude is, in fact, isotropic, as one would expect for pure S-wave scattering.[10] We see now that for $^{24}Mg(\alpha,\alpha)^{24}Mg$ there is, indeed, only one complex non-resonant amplitude (corresponding to two "search" parameters) at each angle. The non-resonant cross section, which is searched on in the more general spin systems analysis, is now easy to evaluate as the absolute square of this single complex background amplitude. This simple scheme was very useful in debugging and understanding the results from the other, more general parts of the resonance fitting program.

5.2 Resolution Broadening

The cross section of equation [5.17] is different from that which makes up the

[10] Note also that this value agrees with the previous result [5.16] for the sum of the squares of the amplitudes (for a one term sum).

experimentally determined excitation function. The experimental or "observed" cross section is the result of the "actual" resonance shape (from the formulae in the previous section) broadened by the experimental resolution effects outlined in chapter III. It is easy to ideate this smearing process if we remember that the pre-scattered particles have a distribution of energies about some mean value. Thus, the point on the excitation function which corresponds to this mean energy value also contains contributions from other points within the range of the beam energy function. For high resolution resonance scans this can be a very significant and noticeable effect.[11] To compare theory to the data, the actual cross section must be smeared out by all the energy spreading mechanisms.[12] Alternately, the distributions from these broadening effects can be combined to form a single total beam resolution function which can then be folded in with the theoretical cross section. The mathematical construct which represents this folding process is the shift-invariant convolution, defined for arbitrary functions $f(E)$ and $g(E)$ by [Arf70]:

$$y(E) = f * g \equiv \int_{-\infty}^{+\infty} g(E')f(E-E')\ dE' \quad . \qquad [5.20]$$

Note that, although the definition appears somewhat asymmetric in f and g, such convolution is commutative : $f * g = g * f$. Also, it is easy to show that convolution is

---

[11] We can see that if the excitation function is relatively flat in the energy range of the beam resolution function then this smearing out effect is completely unimportant. Similarly, if the resonance width is much greater than the width of the resolution distribution then the effect is also negligible. However, for the present work the resonance widths are very small, and the cross sections may vary significantly over the energy range of the resolution distribution width.

[12] As in chapter III, we consider basically three separate energy spreading distributions -- incident beam, target straggling, and Doppler broadening. However, as we shall see, to account for any other unknown spreading mechanisms the programs allow an adjustable width parameter for the total resolution function.

associative as well. Associativity insures that the order of the operation is unimportant, and thus, we can convolute the resolution distributions to produce a total beam resolution function, $R(E)$, which is in turn convoluted with the theoretical cross section. That is,

$$\frac{d\sigma_{obs}(E)}{d\Omega} = \int_{-\infty}^{+\infty} \frac{d\sigma_{th}(E')}{d\Omega} R(E-E')\, dE' \qquad [5.21]$$

This is, in fact, the manner in which the resolution effects are accounted for in the analysis programs. The function $R(E - E')$ is produced by the stand-alone program CONVOL[13] and written to a file which serves as input for the main resonance fitting program, described in section 5.4. More information about the implementation of the convolution for discrete data appears in appendix C.

5.3 Fitting

In the analysis or data fitting procedures, the formulae in the previous sections are utilized to calculate a value of observed cross section corresponding to each data point. This calculated set of points or "fit" is valid for a particular group of resonance and background parameters and is compared with the data via a weighted least squares or chi-squared statistic, which, in our analysis,[14] is given by:

$$\chi^2 = \sum_{i=1}^{n} \frac{(d_i - f_i)^2}{e_i^2} \qquad [5.22]$$

where n is the number of data points, d_i and e_i are the values of the data (cross section) and the statistical error at each point, and f_i are the corresponding theoretical values or

---

[13] This program was checked for accuracy and consistency with functions whose convolution results are known analytically (Gaussians, Lorentzians, and delta functions).

[14] For more information about our particular usage of a chi-squared statistic, see appendix D.

fits. Some or all of the parameters are varied, and new fits are generated until the best fit, given by a minimum value of chi-squared, is obtained. This process is known as *point estimation* since its objective is to locate a particular set of parameters constituting a point, e.g. a chi-squared minimum, in a multi-dimensional parameter space. The quoted values of the partial width and other resonance variables are taken from this best-fit parameter set. Estimating the quality of the fit, so called *goodness-of-fit testing*, is accomplished by examining the chi-squared per degree of freedom.[15] Generally, a value of χ^2/ν close to unity is considered a very good fit, but even values as high as five or ten may indicate reasonable results. On a more quantitative level, in a process known as (*confidence*) *interval estimation*, estimates of parameter errors for specified confidence limits are obtained by considering the change in chi-squared produced by a variation of each particular parameter.

5.3.1 Point Estimation

In general, point estimation consists of finding the point in parameter space at which the test function or statistic has a minimum[16] value. Since the parameter space is usually many-dimensional[17] and the test statistic is often a non-linear function of the parameters, the process can be very complex and difficult to visualize. Furthermore, the space is not guaranteed to be well behaved in the sense that there may be many local

---

[15] Here, the number of degrees of freedom, ν, is given by the number of data points minus the number of free parameters minus one.

[16] Many of the terms used in these descriptions carry with them rigorous statistical or mathematical meanings which may not be explicitly brought to light. However, for our purposes, the more common intuitive meanings will usually suffice.

[17] Since we fit all angles simultaneously, it is not unusual to have 20 or 30 free parameters, although typically only ~ 10 vary during any one search.

minima or minima which are "shallow" such that some parameters may take on a wide variety of values without greatly affecting the test statistic value. For any type of non-linear search involving several variables, factors such as speed and convergence to the true global minimum become important considerations.

A straightforward search which calculates the value of the function at every point at selected intervals in the space, i.e. a grid search, has the advantage of being absolutely convergent. However, finding the true minimum may depend on the chosen size of the grid, and the process can be painfully slow for higher dimensional spaces. Ideally, to save time, the program should be capable of making a somewhat intelligent decision about which direction is toward the minimum and then converge rapidly to the minimum value. While any one search algorithm may not provide all the desirable attributes for a particular problem, our program's organization allows the use of several different search routines, either separately or in conjunction.

5.3.2 Interval Estimation

Although free to chose any suitable test statistic for interval estimation, we employ the same chi-squared function as is used for point estimation. However, we are now mainly concerned with the change in chi-squared, $\Delta\chi^2$, in the local parameter space about the minimum. Analogously, knowledge of this change in chi-squared may be regarded as information about the shape of the parameter space about the minimum value. For single-parameter errors,[18] the usual one standard deviation error or 68.3% confidence interval is given by considering $\Delta\chi^2 = 1$. That is, if the value of the parameter is changed by one standard deviation then the value of chi-squared will increase by one. Note that

---

[18] Single-parameter errors refer to errors calculated for a particular parameter, regardless of the values of all the other parameters.

this is true for single parameter errors, without regard for the values or errors of the other parameters. For multi-parameter confidence regions (simultaneous values of several parameters), one must consider the area enclosed by the $\Delta\chi^2 = 1$ contour in the particular parameter space. Alternately, for normally distributed errors, one can determine multi-parameter errors from a table of chi-squared values [Jam80].

5.4 Computer Analysis -- RESMIN

The analysis program, a collection of library and user-written subroutines written in C and FORTRAN, runs on a Ridge 32 workstation at the University of North Carolina, Department of Physics and Astronomy. Given the name RESMIN, the program may be run interactively or as a detached background process. RESMIN uses the basic procedures outlined above to produce values and errors for the desired resonance parameters.

RESMIN is made up of two main parts. The first part provides the primary user interface and handles the statistical analysis involved in point and interval estimation. For this we use MINUIT -- a general purpose optimization routine originally available from the program library at CERN [Jam77]. To insure maximum flexibility, MINUIT is designed to have very little knowledge of the particular problem at hand and requires only values of the function to be minimized.[19] These values are provided by the second part of RESMIN, a collection of user-written subroutines which deal with the specifics of the data and theory and their comparison via a chi-squared test. For our case, this means generating theoretical cross section excitation functions, convoluting them with the

---

[19] Note that since MINUIT uses only the **values** of the minimization function, it does not require any one particular form for the test statistic. Although we have chosen a chi-squared test, there are several possibilities for useful statistics, including log-likelihood and more general least-squares functions.

resolution function, calculating the value of the chi-squared statistic, and managing several other mundane chores such as input and output of the data and fit. Since RESMIN is initiated via the MINUIT interface, we will consider that part first and then examine user-supplied routines that govern the cross section and chi-squared calculations. After these essential components are described, we will review some of the special options (bells and whistles) designed to make the program more flexible and user-friendly.

5.4.1 Optimization with MINUIT

Applicable to a wide range of optimization problems, MINUIT consists of a package of interrelated FORTRAN subroutines designed to perform the various tasks involved in function minimization. We will be chiefly concerned with those subroutines which may be called explicitly by the user and which deal with point and interval estimation as they relate to the present analysis. The reader interested in more details or information about other features of the MINUIT package is referred to the MINUIT user's manual [Jam77] or any of several MINUIT-compatible articles on function optimization [Jam72] [Gro81] [Jam81] [Jam80] (or, for local readers, [Var86]).

5.4.1.1 SIMPLX and MIGRAD as Point Estimators

For point estimation, MINUIT has two separate subroutines,[20] SIMPLX and MIGRAD, which provide different searching methods and different amounts of information about the parameter space. Each has its own strengths and weaknesses, and

[20] Although unused in the present analysis, there is a third subroutine, SEEK, which uses a Monte Carlo type search algorithm and is useful when there is no reasonable estimate for the starting values of the parameters (far away from minimum).

either can be run separately, although they are typically utilized "back-to-back" -- SIMPLX is allowed to converge and followed by a call to MIGRAD. Each subroutine works by calling for the minimizing function value after adjusting any of the variable parameters within user specified limits and according to the particular searching prescription. This procedure of varying parameters and obtaining a new function value is iterated hundreds (even thousands) of times until the particular minimization criteria for that search method are fulfilled.

SIMPLX utilizes the simplex technique of Nelder and Mead [Nel65]. A complete description of the algorithm can be found in [Jam72] or in the MINUIT manual "long write-up" [Jam77]. The method relies on a figure with n+1 vertices in an n-dimensional parameter space. This construction, called a simplex, requires the test function to be evaluated at each of the n+1 vertices. The vertex with the highest value of the test statistic is replaced by reflection about the hyperplane formed by the other vertices. This process is repeated many times, forming new simplices and continually "stepping" toward the function minimum. SIMPLX is fast and usually converges rapidly even when started far away from the minimum. However, it accumulates little knowledge about the shape of the parameter space and produces only crude estimates of parameter errors.

MIGRAD employs Fletcher's "switching" variation of the Davidson-Fletcher-Powell variable-metric method [Fle70]. The algorithm is much more complicated than the simplex one, and the industrious reader is referred to [Jam72] and the references therein. Ignoring the obvious complexity of the algorithm, we note that the variable-metric method (VMM) is a so-called "gradient" method since it utilizes first and higher order derivatives of the test function with respect to the free parameters. During its journey through the parameter space, MIGRAD calculates and uses the gradients and the covariance matrix (the inverse of the second derivative matrix) to make decisions about the direction of and distance to the test function minimum. Thus, unlike SIMPLX,

MIGRAD accumulates information about the shape[21] of the nearby parameter contours. Since this information requires calculating derivatives of the chi-squared function, in many cases the VMM algorithm is slower than the basic simplex one. However, the acquired knowledge of the local parameter space makes MIGRAD very useful for confidence interval estimation.

As a final note to point estimation, we consider the possibility of the existence of several distinct minima in the chi-squared function. Obviously, one would usually prefer to find the global minimum, that one which has the lowest value of the test function. However, it is unlikely that SIMPLX or MIGRAD would abandon a particular minimum to look for another. MINUIT provides a user-callable subroutine, IMPROV, which actually "removes" the current minimum and then calls SIMPLX from a number of new random starting positions. For our problems, if the user is careful to allow searching only over physically reasonable ranges, IMPROV indicates no other minima in the parameter spaces of the final searches.

5.4.1.2 Interval Estimation with MIGRAD, HESSE, and MINOS

While working its way toward the test function minimum, MIGRAD continually improves its estimate of the covariance or error matrix, which has as its diagonal elements the squares of the individual parameter errors. After each step the matrix is updated, and as MIGRAD nears the minimum, the calculated matrix converges to the true covariance matrix. In this way, MIGRAD can provide good error estimates for n free parameters if the number of calls is large compared to n^2, the approximate number of terms in the covariance matrix.

---

[21] MIGRAD assumes a locally parabolic shape in terms of each parameter; therefore the function value and first two derivatives provide a complete description of the chi-squared contours.

As an alternate means of acquiring parameter errors, the subroutine HESSE may be invoked by the user to independently calculate and invert the second derivative matrix as a means of estimating the covariance matrix. Since it calculates the covariance matrix at the point from which it is called, HESSE is most often used after the minimum has been established by SIMPLX or MIGRAD. HESSE is particularly useful when MIGRAD converges too quickly, before gaining enough information to calculate all the matrix terms.

Errors obtained from the covariance matrix are based on the assumption of a locally parabolic shape for the parameter space near the minimum. For most cases the locally parabolic shape assumption is a good one, and the acquired symmetric error interval accurately represents the true parameter errors. However, if the χ^2 contours about the minimum are asymmetric, MIGRAD and HESSE will fail to account for this irregularity. In these cases, the subroutine MINOS can be called upon to calculate the asymmetric error intervals for any or all of the free parameters. MINOS works by varying the parameter(s) and examining values of the test statistic until a prescribed confidence region is determined. For our analysis, one standard deviation errors are obtained by examining changes in chi-squared until the region of $\Delta\chi^2 = 1$ is established. This method can produce accurate, asymmetric errors and is particularly useful for certain pathological (highly non-linear functions of the parameters) situations. However, for our purposes, the MINOS results are nearly symmetric errors which are very close to those of MIGRAD and HESSE.

5.4.2 Interfacing with MINUIT -- User Subroutines

The discussion so far has centered on the statistical aspects of point and interval estimation. These tasks are capably handled by MINUIT, but not without support from

the user. Remember, MINUIT knows nothing about the specifics of the problem, operating only from the value of the test statistic, which needs to be recalculated many times during the course of a minimization search.

To obtain the test function value MINUIT passes the parameter values along with an integer control flag to a single user-supplied subroutine which must be named FCN. Of course, this subroutine can and does call other subroutines and functions[22] to perform specific tasks as dictated by the value of the MINUIT flag. FCN and its supporting functions make up the remainder of RESMIN, our resonance analysis program. Our user-supplied functions are written in C, compiled separately, and linked together with the FORTRAN subroutines of MINUIT. Since there are an infinite number of possible ways to divide the computational chores among different functions, we will describe the general operation of the program package, and the reader interested in the chosen division of labor and descriptions of the individual functions is directed to the documented program listings in appendix F.

In general, except for the initial and final entry, each call to FCN from a MINUIT subroutine requests a return value of the chi-squared test statistic as defined in equation [5.23]. Here, d_i and e_i, are the cross section data and errors provided by the procedures described in chapter II, and the fit values, f_i, represent the observed cross sections whose calculation poses the real work for the analysis program. For comparison with the data, the theoretical cross section (equation [5.18]) has to be folded with the resolution function via the convolution (equation [5.22]). Furthermore, the f_i must be prepared such that they correspond in energy with the data points, and they must be recalculated each time

---

[22] In the C-programming vernacular a function is a group of statements designed to perform a specific task and for all practical purposes is analogous to the subroutine (or function) of FORTRAN. We regret the ubiquitous usage of the term "function" and hope the reader will not be confused by references to the functions of C programming and mathematical functions (of independent variables).

any parameter is adjusted by MINUIT. Generally, many hundreds of calculations are necessary during a search, and this frequency of calculation of the observed cross section places a premium on program efficiency.

5.4.2.1 Calculating Observed Cross Section

The first step in the determination of the fit values involves using the particular values of the resonance parameters (supplied by MINUIT) to calculate the real and imaginary parts of the Breit-Wigner amplitude given by equation [5.12]. Note that these amplitudes are energy dependent, and thus an array of points is calculated for each set of parameters. Note also, that these values are angle independent and one array is sufficient for each data set.

Next, we use the Breit-Wigner arrays (real and imaginary), along with the values of the background parameters, to calculate the actual cross section given, for example, by equation [5.17]. Now, the background parameters are angle dependent, so the result is an array of cross section points corresponding to each data array. This is, of course, the format that we require for the comparison of data and theory. However, for a meaningful comparison we must smear out the theoretical cross section by convolution with the experimental resolution function.

As mentioned earlier (section 5.2), the total resolution function is calculated outside RESMIN and available as an input file. Once this file is read into an array, it is straightforward and convenient to perform the convolution represented by equation [5.21] using the same program as that which produced the total resolution function. However, as is often the case, convenience takes a back seat to speed when it comes to programming, and the convolution portion of RESMIN is streamlined to allow faster execution. We recognize that the convolution is the most time-consuming aspect of the

cross section calculation, and, while the basic convolution algorithm remains the same as described in appendix C, time is saved by calculating only those points in the energy range of the actual data.[23] Results from the streamlined subroutine version of the convolution algorithm were checked against those of the fully tested stand-alone version outlined in section 5.2.

Perhaps the greatest increase in speed results from noting that the convolution affects only the energy dependent parts of the cross section (equation [5.17]). Thus, if the proper intermediate results are stored, then we should not have to reconvolute after energy independent parameter changes. In particular, this allows very fast searches over the background or non-resonant parameters. We can see how this works by substituting equation [5.17] into equation [5.21]:

$$\frac{d\sigma_{obs}(E)}{d\Omega} = \int_{-\infty}^{+\infty} \left\{ \frac{d\sigma^{(nr)}(\theta)}{d\Omega} + \mathrm{Re}[h(E')F'(\theta)] + \frac{|h(E')|^2}{4k_\alpha^2(2I_A+1)(2I_a+1)} \right\} R(E-E')\,dE' \quad . \quad [5.23]$$

If we separate the real and imaginary parts of $F'(\theta)$ and $h(E)$ so that

$$F'(\theta) = F_r'(\theta) + iF_i'(\theta) \text{ and } h(E) = h_r(E) + ih_i(E) \ ,$$

and, taking into account the Breit-Wigner form for $h(E)$ (equation [5.12]), we find

$$|h(E)|^2 = -\frac{2\sqrt{\Gamma_\alpha \Gamma_\beta}}{\Gamma} h_i(E) \quad . \quad [5.24]$$

Therefore, the observed cross section can be written

---

[23] Although the convolution results consist only of points within the data energy range, points calculated outside of this range must be included. More specifically, to insure reasonable behavior within the energy range of the data, "wing" points should be calculated beyond the data range extremes by an amount greater than the half-width of the resolution function. Also, to provide more accuracy for the convolution (a better approximation to the integral), the functions to be convoluted are prepared in step sizes smaller than that of the data. Step sizes two to four times smaller than the data step are used; smaller intervals produce no change in the results (< 1%). A spline interpolation insures that the points in the final observed cross section arrays correspond in energy to the data values.

$$\frac{d\sigma_{obs}(E)}{d\Omega} = \frac{d\sigma^{(nr)}}{d\Omega}(\theta) + F_r'(\theta)\int_{-\infty}^{+\infty} h_r(E')\,R(E-E')\,dE' + \left[F'(\theta) - \frac{\sqrt{\Gamma_\alpha\Gamma_\beta}}{2k_\alpha^2\Gamma(2I_A+1)(2I_a+1)}\right]\int_{-\infty}^{+\infty} h_i(E')\,R(E-E')\,dE' \quad [5.25]$$

where

$$h_i(E) = \frac{-\frac{\Gamma}{2}\sqrt{\Gamma_\alpha\Gamma_\beta}}{(E-E_R)^2+\left(\frac{\Gamma}{2}\right)^2} \quad \text{and} \quad h_r(E) = \frac{(E-E_R)\sqrt{\Gamma_\alpha\Gamma_\beta}}{(E-E_R)^2+\left(\frac{\Gamma}{2}\right)^2}\,. \quad [5.26]$$

Even more to the point, substituting for h_i and h_r and simplifying, we have:

$$\frac{d\sigma_{obs}(E)}{d\Omega} = \frac{d\sigma^{(nr)}}{d\Omega}(\theta) + F_r'(\theta)\sqrt{\Gamma_\alpha\Gamma_\beta}\int_{-\infty}^{+\infty} \frac{(E'-E_R)\,R(E-E')}{(E'-E_R)^2+\left(\frac{\Gamma}{2}\right)^2}\,dE' + \left[\frac{\Gamma_\alpha\Gamma_\beta}{4k_\alpha^2(2I_A+1)(2I_a+1)} - \frac{\Gamma\sqrt{\Gamma_\alpha\Gamma_\beta}}{2}F_i'(\theta)\right]\int_{-\infty}^{+\infty} \frac{R(E-E')}{(E'-E_R)^2+\left(\frac{\Gamma}{2}\right)^2}\,dE' \quad . \quad [5.27]$$

Thus, as long as we save the results represented by the above two integrals, we need not perform the convolution again for changes in the background amplitudes or the partial widths. While the coding of the above equations produces some loss of programming clarity due to extra flags and conditional branches, the loss is more than offset by the gained increase in execution speed (by a couple of orders of magnitude), especially noticeable in the interactively running environment.

Each calculation of the observed cross section arrays and corresponding value of the total chi-squared takes a few seconds, and thus entire minimization searches (several hundred calls) may be completed in a matter of minutes. As the reader eager for more

details scrambles toward the program listings in appendix F, we turn our attention to other miscellaneous program details not directly involved in the minimization or calculation of chi-squared.

5.4.3 Housekeeping, Bells, and Whistles

Shying away from details, we note that MINUIT has numerous other functions and capabilities and ably performs its own housekeeping chores. Topics such as selecting which parameters to vary, keeping track of the parameter values and errors, and estimating the distance to the minimum are probably best left to the MINUIT experts ([Jam72] or [Jam77] and the references therein). One thing the user does need to know is that MINUIT begins its execution by accepting initial parameter values and error estimates from the keyboard or from a specified file. As part of this initial information the user may specify intervals in which the parameters are allowed to vary. These intervals may correspond to actual physically meaningful ranges (e.g. branching ratios between zero and one) or boundaries selected by the user to decrease the space of the search. MINUIT allows either constant or variable parameters. Note that variable parameters may be later fixed (by the FIX command in MINUIT) for a particular search, but initially constant parameters must remain invariant through the entire MINUIT session.

One consequence of this input format is that MINUIT treats each parameter independently, and multi-parameter constraints are difficult to apply. Thus, it is impossible to constrain the value of the branching ratio if we have as parameters the total and partial resonance widths. In order to work around this difficulty, we define the MINUIT parameters in such a way as to allow the constraints to be directly applied to them. For example, in the previous case we might (and in fact do) select the parameters

to be the branching ratio and the partial width, leaving the total width to be extracted in the calculational stages of the program.

After accepting the parameters, MINUIT makes its initial call to the user function FCN. Along with the parameter values, MINUIT passes the particular value of a flag which signals FCN that it is the first call. The flag directs FCN to call the proper functions to read in the data and the resolution function, set certain flags and variables, and calculate and store constants which will enter into the cross section calculation. The data, cross sections, errors, and energies in the laboratory frame are read from a file into arrays in the designated data structure. Data for up to six different angles can be used, and RESMIN produces simultaneous fits at all angles, with the total chi-squared being the sum of those at each angle. The cross sections and scattering angles are converted to the center-of-mass system for comparison to the theoretical predictions. The resolution function, probability versus energy, is also read from a file into designated arrays. A spline interpolation insures that the resolution function steps are compatible with those of the calculated cross section. Since there are fairly large uncertanties associated with the overall width of the resolution function, we include as a search parameter an adjustable width factor, so that the curve may be expanded or contracted about the mean value without affecting the relative shape of the distribution.

A normal exit from a MINUIT session is signaled by the EXIT command which necessitates a final call to FCN with the predefined flag exit value. Upon receiving this flag, FCN directs the writing of the final parameter values, chi-squared, and fit values to a file or, a seldom used option, to the terminal. Actually, the user can demand that MINUIT call FCN with any value of the control flag. Thus, the user can perform many specific tasks inside of MINUIT and without MINUIT's knowledge. Care must be taken that these "behind-the-back" changes do not confuse MINUIT if it is to continue its minimizing actions. Among the options available in the current version of RESMIN are:

1) Smooth wings of data -- use standard smoothing technique [Bev69] to average over adjacent data points. It was hoped that smoothing would make the fit rely less on the wing data, but, in fact, this option served only to lower the overall χ^2 and thus was not used in any of the final searches.
2) Truncate wings of data -- by deleting points on the wings, we were able to increase the sensitivity of the search to the resonance region. This procedure is used only after the background parameters are determined and fixed.
3) Shift resonance energy -- adjust resonance energy to value determined by analytically differentiating χ^2 to find its minimum value. This method is not used extensively since MINUIT has no trouble pinning down the resonance energy and has the advantage of providing error estimates.
4) Normalize data to minimize χ^2 -- also an outgrowth of the analytical expression for the chi-squared first derivative. This is very dangerous since the partial width is strongly affected by this procedure. For our data it is not needed and not used.
5) Determine the background slope -- The slope of the off-resonance background cross section is determined from data points at each end of the energy range by a least-squares fit to a straight line. Data points in the resonance region are not included in the calculation. This procedure was used in all the final analyses.
6) Calculate χ^2/ν for goodness-of-fit -- The calculation of chi-squared per degree of freedom, at a particular χ^2 minimum, provides an indication of the goodness-of-fit of that parameter set. Note that this calculation is only qualitative, with the total chi-squared used for point and interval estimation.
7) Change parameter values -- used extensively to perturb the fit and see if MINUIT returns to the same point.

8) Graph the data and fits -- Simple graphs were very useful in directing the course of the minimization. They also proved invaluable for recognizing gross errors in data and parameter input.

9) Write intermediate results to a file or terminal using the same flag as used by the EXIT command.

10) Turn calculations on/off -- The ability to disregard a particular data set proved very useful in both the preliminary runs and the final analysis. It was especially useful for the $^{31}P(p,p)^{31}P$, $^{31}P(p,\alpha)^{28}Si$ data where experimental difficulties resulting in very large error bars forced several data sets to be omitted from the final fits.

VI. EXPERIMENTAL DETAILS

This chapter presents the specific details involved in data acquisition for each of the three separate experiments. In particular, we will relate any deviations in the standard high-resolution scheme outlined in chapter II. The determination of the constituent resolution distributions and their consolidation into the total resolution functions for each experiment will also be described. Although many of the calculations involved with the resolution function are subject to fairly large errors, we will endeavor to include each contribution as accurately as possible. Furthermore, there are instances in which a particular resolution ingredient is essentially negligible but will nevertheless be included in the final result since, in many ways, the computational methods are as important as the values themselves.

Since the experimental techniques were refined over the years during which the data were obtained, it is advantageous to describe the experiments in chronological order.[1] Thus, we will begin with an explication of the $^{31}P(p,p)^{31}P$, $^{31}P(p,\alpha)^{28}Si$ experiment, designed to study the lowest $T = 2$ state in ^{32}S. After that, we examine the lowest $T = 2$ state in ^{24}Mg via the $^{23}Na(p,p)^{23}Na$ and $^{23}Na(p,\alpha)^{20}Ne$ reactions. Finally, we look at $^{24}Mg(\alpha,\alpha)^{24}Mg$ to investigate the lowest $T = 2$ state in ^{28}Si.

6.1 Experimental Common Ground

Before discussing the details of particular experiments, it is worth commenting on

---

[1] Although all the experiments were studied in preliminary investigations, we generally consider only the final experiments for each system, since they are used to obtain the results for the resonance parameters. However, where appropriate and interesting, we will compare and contrast problems and results from previous experiments performed by us and by others.

some general attributes and philosophy which are common to all the experiments. The reader of chapters I and II will be familiar with the basic idea of the high resolution system and its associated hardware as well as the general format for the online data acquisition and storage. We recall that the main part of the stored online data consists of three-dimensional data areas which have scattered particle counts (z-axis) versus target ramp voltage (incident particle energy -- y-axis) versus particle exit energy in the region of interest (x-axis).

At the risk of repeating some of the material which was outlined at the end of chapter II, we recall that the objective of the offline analysis is to convert the scattered particle counts to differential cross sections. This is accomplished by summing the exit energy peaks, subtracting the appropriate background, making corrections for things such as ADC dead time, and utilizing the necessary experimental parameters such as target thickness and integrated beam current. The resulting cross sections are then plotted versus energy obtained from the 90-90 magnet system and the variable voltage target ramp. These excitation functions, one for each reaction at each angle, are then numerically convoluted with the total experimental resolution function and compared with results from the theoretical model by the resonance fitting programs described in chapter V.

Several experimental details common to all the experiments are worth reiterating here (and will not be repeated in each section). All the experiments follow the general procedures outlined in chapter II, and all use the high resolution system with all its implied beam transport, gas stripping, and standard slit settings. All experiments employ thin detector collimators and thin slits to define the beam as it enters the scattering chamber. Also, to reduce carbon buildup on the target, all experiments use a liquid nitrogen cold trap between the scattering chamber and the main beam transport system. All the $T = 2$ state resonance scans and resolution function measurements use the target

ramping method to produce the varying energy necessary to scan the resonance region. Finally, all resonance scans were preceded by the standard magnet system recycling procedures as outlined in section 2.3.

6.2 Lowest $T = 2$ State in ^{32}S

Because of the small α-branching ratio (4 ± 4 %) [Fre79], formation of this state is accomplished through the proton channel, and the data consist of excitation functions for $^{31}P(p,p)^{31}P$ and $^{31}P(p,\alpha)^{28}Si$. An earlier, complementary experiment by Vernotte, *et al.* [Ver73] used the $^{31}P(p,\gamma)^{32}S$ reaction to locate a narrow resonance at $E_p = 3.289 \pm 0.003$ MeV, but the experimenters were unable to identify a corresponding structure in the (p,p) or (p,α) cross section curves. Based on angular distribution and γ-correlation measurements, they were able to assign $J = 0$ to this level. Furthermore, their data from the γ-decays and energy information from $^{34}S(p,t)^{32}S$ [Cer68], allowed them to identify this resonance with the lowest $T = 2$ state in ^{32}S. Because of the absence of the resonance in the proton channel, they assign an upper limit of 230 eV to the level's total width. In addition to the $T = 2$ resonance, Vernotte was able to associate a large resonance in both the (p,γ) and (p,p) reactions with a J^{π}, $T = 4^-$, 1 level in ^{32}S. This resonance appears at $E_p = 3.283 \pm 0.003$ MeV, approximately 6 keV below the $T = 2$ state. Owing to its relatively large proton elastic scattering cross section and large excursion from the background (400% at 162°), this resonance provides an unmistakable marker for the desired energy region. In fact, with our beam currents and a target thickness of 12 $\mu g/cm^2$, this resonance appears in the back angles after only a few minutes of data collection.

For the present experiment, utilizing the 10 kV target ramping circuit, the initial energy scan covered $E_p = 3.275$ to 3.285 MeV, the region of the aforementioned 4^- state. Confirming our energy calibration, the resonance appears at about $E_p = 3.282$ MeV; the

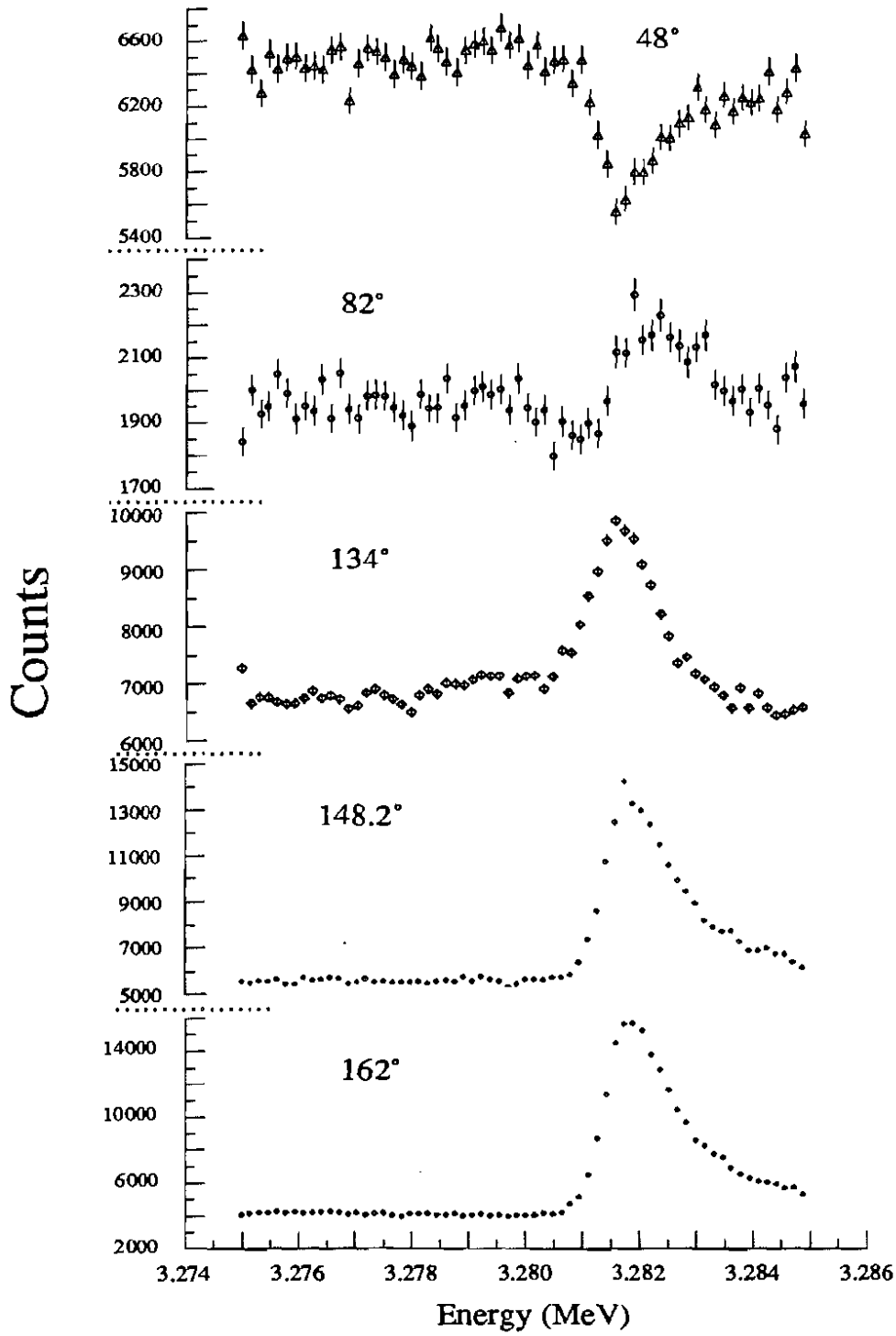


Figure [6.1]: ^{31}P(p,p)^{31}P resonance scans for the J^{π}, $T=4^{-}$, 1 state in ^{32}S. This level has Γ= 470 eV and is about 6 keV below the T = 2 state of interest.

yield curves for proton elastic scattering are shown in figure [6.1]. After increasing the beam energy with the magnet system and changing to a thinner target, the $T = 2$ resonance scan covers the energy range from about 3.284 to 3.292 MeV in 160 eV intervals. The data were accumulated over many hours, and the $T = 2$ state resonance appears in both the (p,p) and (p,α) data at about $E_p = 3.288$ MeV.

In the $T = 2$ resonance region, data were obtained for two different sets of angles, with slightly different energy ranges and energy increments. Our experimental format, showing each set of angles and some relevant parameters, is given in figures [6.2a] and [6.2b]. The two sets of data were analyzed independently, albeit using identical methods and programs. The 48° data from angle set one were not used because of poor statistics and a relatively large instantaneous jump in the elastic scattering cross section. This could be caused by incompletely resolved exit energy peaks resulting in less than adequate background subtraction. For the second data set, perhaps resulting from a partially blocked detector, abnormally low cross sections and poor statistics render the 87° data useless.

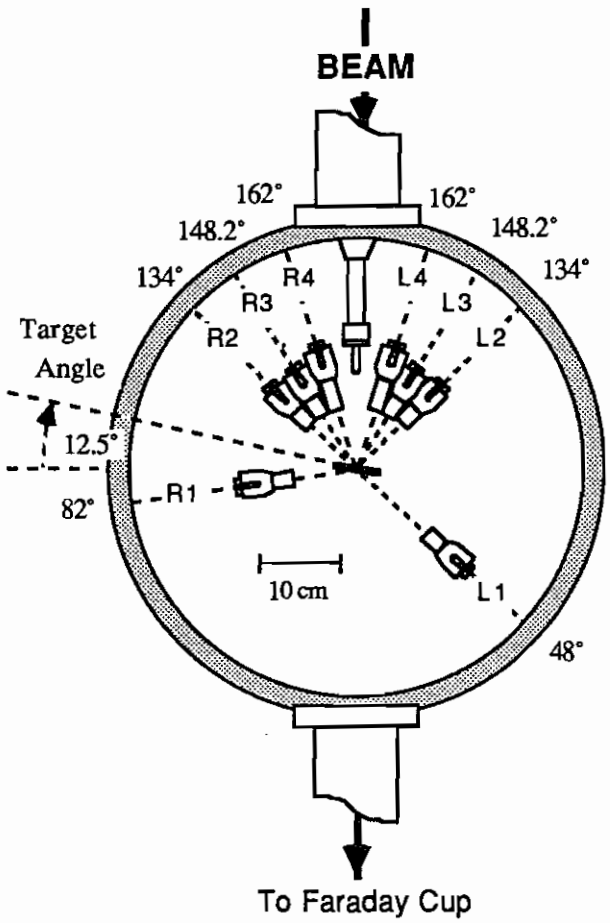
The phosphorus targets were of good quality with many of a usable thickness[2] (1 to 3 $\mu g/cm^2$). Showing some signs of evaporation under our microampere of proton beam, the targets were replaced periodically during the experiment to avoid the possibility of serious deterioration or of major carbon buildup which might adversely affect the target resolution. Except for the expected tantalum[3] and unavoidable oxygen impurities, the targets were relatively contaminant free, and the exit energy spectra were fairly clean. A sample spectrum is shown in figure [6.3].

If the targets were some of the best for high resolution work, then the

---

[2] Thicknesses were determined by Coulomb scattering at the most forward angles and by comparison to the cross sections of [Ver73].

[3] From the chimney heater wire, see section 4.2.2.

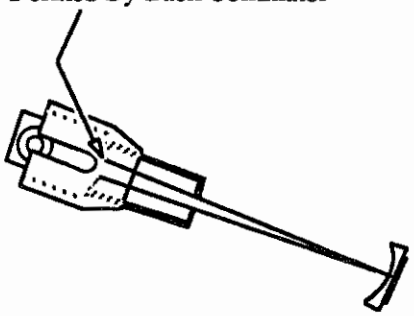


$^{31}P(p,p)^{31}P$ / $^{31}P(p,\alpha)^{28}Si$

Angle Set #1

Detector	Lab Angle	dΩ (msr)	Horizontal Angular Acceptance
R1	82°	1.?	0.80°
L1	48°	0.65	0.60°
R2 + L2	134°	6.2 + 6.2	3.2°
R3 + L3	148.2°	6.2 + 6.2	3.2°
R4 + L4	162°	5.0 + 5.0	2.9°

Angular Acceptance And Solid Angle Defined By Back Collimator



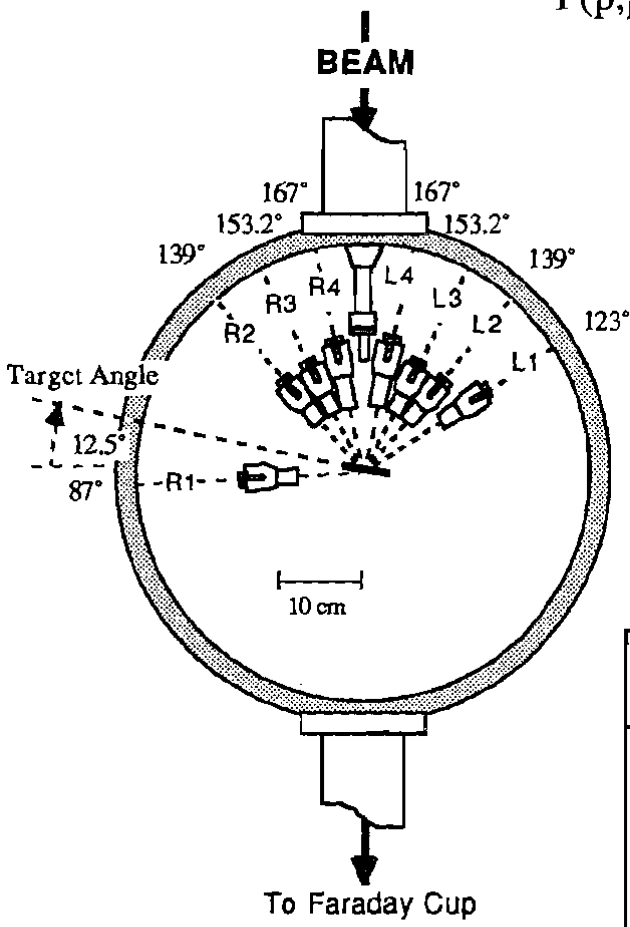
Total Energy Range (MeV)	3.275 - 3.293
T = 2 Resonance Scan (MeV)	3.283 - 3.293
Target Ramp Voltage (kV)	± 4.96
Number of bits in ramp	64
Energy Increment (eV)	157
^{31}P Target Thicknesses (μg/cm^2)	2 - 3 *
Ta Contamination (μg/cm^2)	0.04 - 0.1
Average Beam Intensity (μA)	1.0 – 1.5
$^{31}P(p,\alpha)^{28}Si$ Q-value (MeV)	1.916

* A 12 μg/cm^2 target was used outside of the T = 2 resonance region

P_ℓ Zeros (lab angles)	
E_p = 3.28 MeV	P_4
$^{31}P(p,p)^{31}P$	148.4°
$^{31}P(p,\alpha)^{28}Si$	147.7°

Figure [6.2a] : First experimental setup to study the lowest T = 2 state in ^{32}S

$^{31}P(p,p)^{31}P$ / $^{31}P(p,\alpha)^{28}Si$



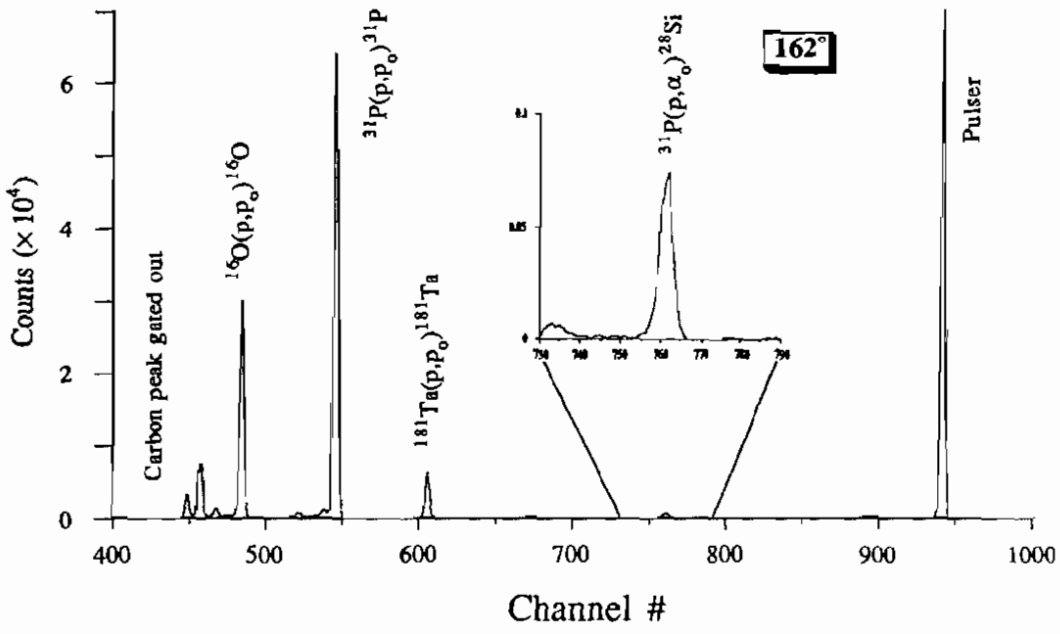
Angle Set #2

P_ℓ Zeros (lab angles)			
E_p = 3.28 MeV	P_{odd}	P_2	P_3
$^{31}P(p,p)^{31}P$	88.2°	124°	140°
$^{31}P(p,\alpha)^{28}Si$	86.9°	123°	139°

Detector	Lab Angle	dΩ (msr)	Horizontal Angular Acceptance
R1	87°	6.2	3.2°
L1	123°	3.5	2.4°
R2 + L2	139°	6.2 + 6.2	3.2°
R3 + L3	153.2°	6.2 + 6.2	3.2°
R4 + L4	167°	5.0 + 5.0	2.9°

Total Energy Range (MeV)	3.284 - 3.293	^{31}P Target Thicknesses (μg/cm^2)	2 - 3
T = 2 Resonance Scan (MeV)	same	Ta Contamination (μg/cm^2)	0.04 - 0.1
Target Ramp Voltage (kV)	± 5.15	Average Beam Intensity (μA)	1.0 – 1.5
Number of bits in ramp	64	$^{31}P(p,\alpha)^{28}Si$ Q-value (MeV)	1.916
Energy Increment (eV)	164		

Figure [6.2b] : Second experimental setup to study the lowest T = 2 state in ^{32}S .



Typical Target Composition

(target C33)

Active Layer:	2.2 μg/cm^2 of ^{31}P
	0.1 μg/cm^2 of Tantalum
Backing Layer:	4.3 μg/cm^2 of Carbon

Figure [6.3]: Typical exit energy spectrum and target composition for the $^{31}P(p,p)^{31}P$ / $^{31}P(p,\alpha)^{28}Si$ experiment. The incident projectile energy is 3.29 MeV and the spectrum energy calibration is 5.3 keV/channel.

experimental conditions were some of the worst. This particular experiment was plagued by magnet instabilities and feedback problems. The NMR feedback circuit was often operating near the limits of its error signal to keep the 90-90 field, and hence the energy, regulated. We later learned that some changes in the master reference system[4] had the effect of loading down the 90-90 magnet power supply. This was also one of our early encounters with the beam line steerer feedback's affecting the NMR regulation system through a still unknown mechanism.[5] Without fully considering the consequences for the data analysis, we compounded our problems by periodically attempting to adjust the magnet current by a negligible amount (programmed a 1 eV energy shift) in order to recenter (or "relax") the NMR error signal for each new run.

In retrospect, it is doubtful whether this procedure really helped avoid any more serious energy errors. What it did do, however, was to shift the beam energy by zero to about a hundred eV for almost all of our forty different runs. Although the amount of energy shift is known for each run, the problems of trying to account for this in the offline analysis were substantial. After considering many interpolation schemes to correct for shifts which are smaller than one step, we finally calculated cross sections for each energy-shifted run and averaged them together to the nearest point. Thus, there is a built-in error of half a step in the energy axis of the final function.

To add to the energy woes of this experiment, near the finish of data taking for the first angle set, we discovered the target ramp was not making a complete first step,

---

[4] The master reference is a way of tying together the optical elements of the high resolution system. The idea is that changing the 90-90 magnet setting with the master reference alters the optical elements in such a way as to maintain the beam focussing and transport at all energies.

[5] It appears that certain of the steerer feedback power supplies were somehow tied in with the master reference. Although its origins are not understood, the problem has been documented and is a real effect. In the latter part of this run and subsequent runs the offending feedback units were simply not used.

perhaps missing this step entirely. This could have had the effect of seemingly narrowing the resonance, but, since we do not know how many of the runs were affected, the extent of the damage could not be estimated in advance. From the analysis of the two angle sets, the effect of this lost step appears to be negligible.

6.2.1 Resolution Function

In order to be handled accurately by the resonance fitting program (RESMIN), the cross section excitation curves must be accompanied by an estimation of the total resolution function. Without repeating the general discussion of resolution concerns covered in chapter III, we now turn our attention to the specific makeup of the resolution function for this particular experiment (illustrated in figure [6.4]).

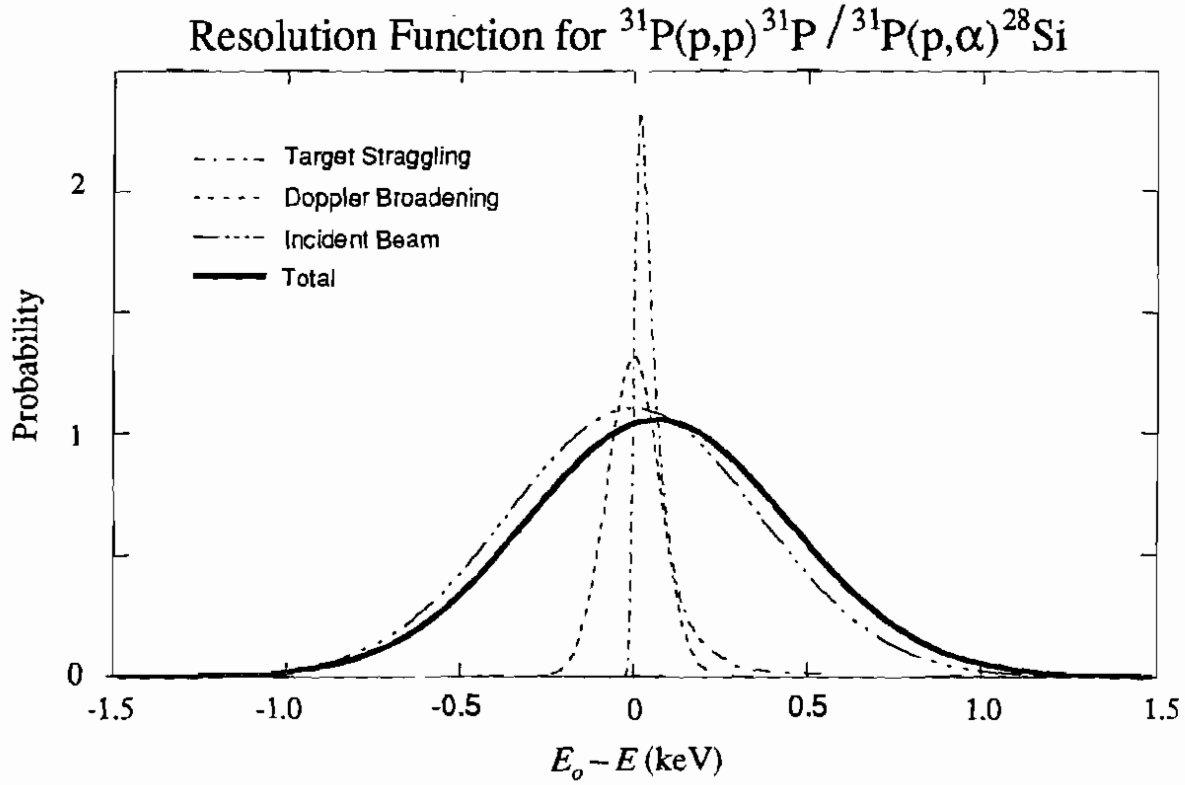
6.2.1.1 Target Straggling

The target resolution function is generated by the program TS, as described in section 3.2. Table [6.1] column [a] gives the relevant data for the straggling calculation, and the distribution itself is shown as part of figure [6.4]. Note that, although included in the analysis, the target energy distribution does not contribute significantly to the overall resolution function.

6.2.1.2 Intrinsic Beam Resolution for the 3.3 MeV Proton Beam

Not nearly so straightforward is the estimation of the incident beam energy function. The experiment was performed at a time before the full development of the "ramp-ramp" system[6] for direct measurement of the beam distribution, and thus there

[6] See section 3.1.1 for a description of this direct measurement system.



	Incident Beam	Target Straggling	Doppler Broadening	Total Resolution Function
Description	Joined half-Gaussians	Sum of Vavilov Distributions	Gaussian	Convolution Result
Mean $[\mu_1']$ (keV)	0.00 *	0.0722	0.00 *	0.0722
Variance $[\sigma^2 = \mu_2]$ (keV^2)	0.130	0.00676	0.00572	0.143
Skewness $\left[\gamma_1 = \mu_3/\mu_2^{3/2}\right]$	0.00	2.51	0.00	0.0257
Kurtosis $\left[\gamma_2 = \left(\mu_4/\mu_2^2\right) - 3\right]$	–0.000285	8.11	0.00	0.0178

$\mu_n' \Rightarrow n^{th}$ algebraic moment $\mu_n \Rightarrow n^{th}$ central moment *set to this value

[For Normal Distribution, FWHM, $\Gamma = 2\sqrt{2\ln 2}\,\sigma = 2.355\sigma$]

Figure [6.4]: Resolution function and its constituents for the E_p = 2.288 MeV proton beam for the $^{31}P(p,p)^{31}P$ / $^{31}P(p,\alpha)^{28}Si$ experiment. For an explanation of the distribution properties and a description of the notation used, see [Ead71].

TARGET STRAGGLING DATA			
	[a]	[b]	[c]
Target Thickness	3.0 $\mu g/cm^2$ ^{31}P	2.3 $\mu g/cm^2$ ^{23}Na	1.35 $\mu g/cm^2$ ^{24}Mg
Major Contaminant Thickness	0.1 $\mu g/cm^2$ Ta	3.4 $\mu g/cm^2$ ^{35}Cl	0.48 $\mu g/cm^2$ Ta
Projectile	proton	proton	alpha ++
Projectile Energy	3.288 MeV	3.905 MeV	6.112 MeV
Number of Target "Slices" for Calculation	3	3	3

Table [6.1]: Data for target straggling calculations by the program TS for the three different targets (columns [a], [b], and [c]).

exists only circumstantial evidence insinuating the width of the resolution function. In particular, this evidence is in the form of excitation functions containing a very narrow resonance in $^{32}S(p,p)^{32}S$, representing presumably the lowest $T = 3/2$ level in ^{33}Cl. This state has a total width, Γ, of about 100 eV, and the corresponding resonance occurs at approximately the same proton energy ($E_p = 3.379$ MeV) as the ^{32}S $T = 2$ state resonance in $^{31}P(p,p)^{31}P$ [Wil82]. Since the actual width is much smaller than the resolution effects, the observed width of the resonance should provide a fairly good indication of the **total** resolution function. For our very thin target (2.8 $\mu g/cm^2$ of Sb_2S_3), the straggling contribution is relatively negligible, and the observed width is a measure of the incident beam plus Doppler broadening functions. Furthermore, by estimating the Doppler broadening width, we can extract an estimate of the width[7] of the incident beam energy distribution. Unfortunately, it is not simple to estimate the extent of Doppler broadening for a this Sb_2S_3 (or any other compound) target, and we simply approximate the mean energy of the target atoms (section 3.3) by the classical limit value, kT, to get a Doppler FWHM of < 200 eV for our typical target temperatures.

The excitation function data are given in figure [6.5], and the only remaining obstacle is the estimation of the observed resonance width. For resonance shapes that are symmetric about the resonance energy, the widths can easily be obtained by actually measuring the full width at half the maximum value. However, for resonance structures with a marked asymmetry about E_r ("S-shaped"; caused by interference with the background), the widths are much more difficult to estimate accurately. From experience and examination of fits for several other narrow resonances with similar shapes and known widths ([Wil82]), we note that an approximate width may be obtained by

---

[7] Note that we cannot gain information about the shape of the distribution from this simple estimation scheme, so a Gaussian form is assumed.

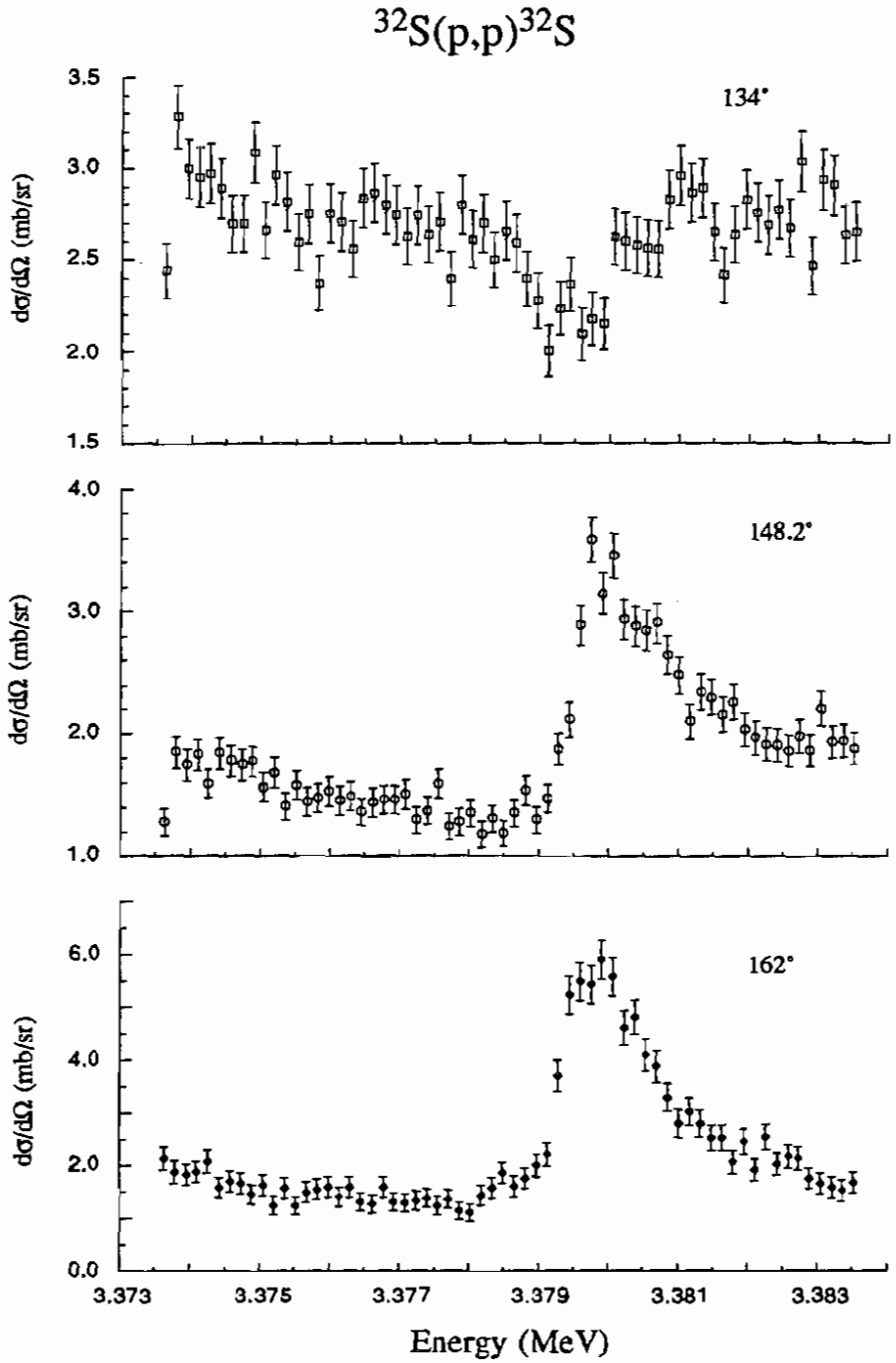


Figure [6.5]: Excitation function data for $^{32}S(p,p)^{32}S$ used to estimate the incident beam energy resolution for the proton beam used in the $^{31}P(p,p)^{31}P$ / $^{31}P(p,\alpha)^{28}Si$ experiment.

considering the 10% and 90% values in the region of the maximum slope.[8] We acknowledge that this method of determining widths has very tenuous justification and holds only approximately for certain forms of the resonance amplitudes [Tho89]. Also, we note empirically that this "10% - 90%" value is almost always an underestimate of the FWHM, in extreme cases by as much as 30%, but more typically 10% to 20%. Based on the apparent widths of the $^{32}S(p,p,)^{32}S$ resonances and considering both the uncertainty in the data as well as the inherent error involved in representing the true FWHM, we estimate the width of the incident beam energy distribution to be 900 ± 200 eV, after unfolding the estimated Doppler broadening.

There is, however, another piece of information pertinent to the beam energy spread -- the energy resolution of the 90-90 magnet system itself. Running in the high resolution mode with its associated slit settings, the magnet system (energy resolution $\Delta E/E \approx 2.4 \times 10^{-4}$) gives an energy range (FWHM of the triangular system resolution function) of about 800 eV. Errors for this value are very difficult to estimate, but we acknowledge the fact that this value should be an approximate upper limit.[9]

Without any real knowledge of the relative precision of the two values, we simply average them together and represent the incident beam energy function by a Gaussian distribution with FWHM of 850 eV. In any case, this estimated value is higher than might be expected and represents the dominant contribution to the total resolution

---

[8] In order to make the "10% - 90%" estimation, one must envision a fit to the data in order to extract the energy width. With the fairly marginal statistics available, trying to pick out data points at the 10% and 90% positions can be misleading. For our determination, we also considered the data after adjacent points were averaged to give a slightly smoother function. As might be imagined, even when done with much care there is room for substantial error in this estimate.

[9] Although this value should provide an absolute upper limit for the beam energy spread, we note that this value assumes specific settings for the various optical elements in the system and thus is susceptible to errors resulting from deviations in ideal running conditions.

function. However, the magnitude of this value is not surprising given the aforementioned problems with the energy regulation and feedback systems. Despite the possibility of fairly large errors, this value should provide a reasonable starting point for the calculations. The resulting incident energy distribution and its relevant parameters are given as part of figure [6.4].

6.2.1.3 Doppler Broadening for p +^{31}P

Methods for the calculation of the Doppler broadening distribution have been outlined in section 3.3. The first step is estimating the target temperature based on the energy deposited in the target by the incident beam as a result of energy losses in the phosphorus, any contaminants, and backing materials. This calculation is straightforward, using equation [3.11] and the parameters shown in the first column of table [6.2]. The temperature is, as expected for our small target thicknesses and moderate beam energy, only slightly higher than room temperature. Using this value of the target temperature, we can estimate the mean energy of the target atoms using the models outlined in the appendix B. For monatomic targets, such as the phosphorus ones used here, these calculations may be fairly accurate and are simple if the Debye temperature of the substance is known. However, a search of the available literature finds no listed values of the Debye temperature of phosphorus, and it is likely that this value has never been determined.

Without a readily obtainable value of the Debye temperature, there are several approaches to estimating the width of the Doppler broadening distribution. First, we could simply ignore the Doppler contribution, considering it negligible. This would be valid since the estimated width of the incident beam resolution is so large. In particular, if we use one of the highest known Debye temperatures, that of diamond (Θ_D = 2230 K),

Data for Doppler Broadening Calculations			
Target	^{31}P	^{23}Na	^{24}Mg
Projectile	proton	proton	α^{++}
Projectile energy	3.288 MeV	3.905 MeV	6.112 MeV
Average Beam Intensity	1.5 μA	1.0 μA	300 nA
Area of beam spot on target	5 mm^2	5 mm^2	5 mm^2
Mean energy loss in target (ε)	670 eV (1)	850 eV (2)	3.74 keV (3)
Surrounding temperature (T_s)	300 K	300 K	300 K
Estimated target temperature (T_t)(eqn. [3.11])	328 K	325 K	320 K
Debye temperature (Θ_D)	204 K (4)	321 K (5)	318 K (6)
Value of Debye integral (I) (eqn. [B.14])	0.786	0.678	0.675
Mean quantal energy ($\overline{E_t}$) (eqn. [3.8], using Debye approximation)	28.9 meV	29.4 meV	28.9 meV
FWHM of Doppler broadening normal distribution	179 eV	225 eV	500 eV

(1) based on target composition of figure [6.3]
(2) based on target composition of figure [6.7]
(3) based on target composition of figure [6.13]
(4) for phosphorus, from elastic constants -- see Appendix E
(5) for sodium chloride [Ash76]
(6) for magnesium [Ash76]

Table [6.2]: Data and results for Doppler broadening calculations. See section 3.3 and appendix B for details.

then the calculation yields a Doppler FWHM of only 286 eV. This "worst case" Doppler distribution convoluted with the 850 eV (FWHM) beam Gaussian produces an 897 eV wide result, well within the errors of the original incident beam distribution. Although ignoring the Doppler broadening is certainly adequate for the results, it violates our philosophy of including all contributions as accurately as possible.

Another approach would be to approximate the mean energy in equation [3.7] by the classical limit, kT, as was done in the previous examination of the $^{32}S(p,p)^{32}S$ data. This is equivalent to assuming a low Debye temperature, and we have shown that this limit gives results within 5 % of those from the Debye model when $\Theta_D \leq T$ (appendix B).

Alternately, we can attempt to approximate the Debye temperature from the elastic properties of the target material. In particular we can, with approximations relating the transverse and longitudinal velocities, relate the Debye temperature to the compressibility and density [Sei40]. The density of red phosphorus is known (2.2 gm/cm^3) and its initial compressibility, κ, has been measured by Bridgman [Bri27] to be 5.45×10^{-6} cm^2/kg or 5.56×10^{-6} bar^{-1}. The result of this approximation is $\Theta_D = 204$ K, which leads to a mean energy of 1.02 kT, or 28.9 meV, for our estimated target temperature. (See appendix E for the details of the calculation.) The essential final calculation parameters and results for the Doppler broadening distribution are given in table [6.2].

6.3 Lowest $T = 2$ State in ^{24}Mg

The lowest $T = 2$ state in ^{24}Mg has been studied via resonances in the $^{23}Na(p,p)^{23}Na$ and $^{23}Na(p,\alpha)^{20}Ne$ reactions. A previous experiment [Rie67] used γ-γ correlation measurements to confirm the $J = 0$ assignment of the resonance state in $^{23}Na(p,\gamma)^{24}Mg$ at $E_p = 3.906 \pm 0.003$ MeV. Another experiment with $^{23}Na(p,\gamma)^{24}Mg$

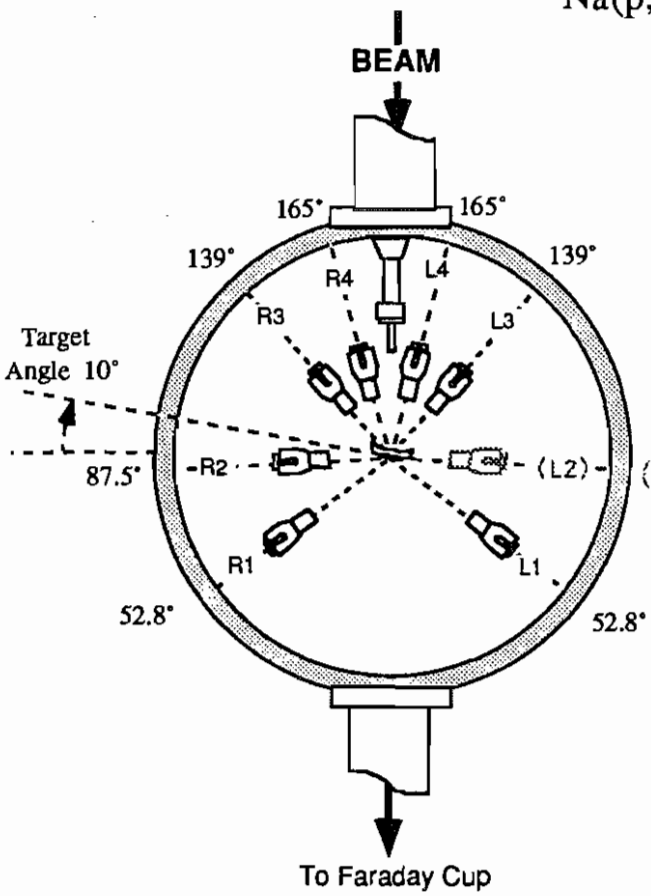
locates the $T = 2$ resonance at $E_p = 3.906 \pm 0.002$ MeV [Osb78], and an experiment similar to our own [McD78] gives the energy as 3.903 ± 0.010 MeV. Both of the latter experiments rely, as we will, on this well known energy position and the relatively narrow width to establish the observed resonance as corresponding to the lowest $T = 2$ state in ^{24}Mg. The branching ratios have been established by the ^{26}Mg(p,t)^{24}Mg reaction [Fre79] and are given in table [1.2]. The experiment of McDonald, *et al.* [McD78] uses the ^{23}Na(p,p)^{23}Na and ^{23}Na(p,α)^{20}Ne reactions to observe the $T = 2$ resonance in excitation functions at a scattering angle of 173°. They note that instabilities in the analyzing magnet field stabilization produce somewhat poorer resolution than they had anticipated. Nevertheless, they quote a value for the total width of $300 \text{ eV} < \Gamma < 700 \text{ eV}$ (using the branching ratios given by [Fre79]).

For the present experiment, data were obtained for two different sets of angles with four angles in each set; the experimental setups and details are given in figures [6.6a] & [6.6b]. By virtue of much better statistical precision, data for the first angle set are used for the extraction of the resonance parameters, and the second angle set data are used only as a check. The targets used were NaCl on carbon with some having a backing layer of gold to act as a heat sink.[10] Over the course of the experiment several different targets were used to guard against possible evaporation and/or carbon buildup. Target thicknesses were based on Rutherford scattering at the most forward angles and comparison to other known thickness targets. A typical exit energy spectrum and target composition are presented in figure [6.7].

Initially, an excitation function was obtained in coarse steps by incrementing the field of the 90-90 magnet system. Utilizing this "stepping" mode, a resonance at about E_p

[10] We were aware of a problem with NaCl targets evaporating under several microamperes of proton beam [Van85]. However, for our ~ 1 μA of beam we noticed only small changes in target thicknesses over several hours.

$^{23}Na(p,p)^{23}Na$ / $^{23}Na(p,\alpha)^{20}Ne$



Angle Set #1

Detector	Lab Angle	dΩ (msr)	Angular Acceptance
R1 + L1	52.8°	3.0 + 3.0	2.2°
R2 (÷ L2)	87.5°	6.2 (÷ 6.2)	3.2°
R3 + L3	139°	6.2 + 6.2	3.2°
R4 + L4	165°	6.2 + 6.2	3.2°

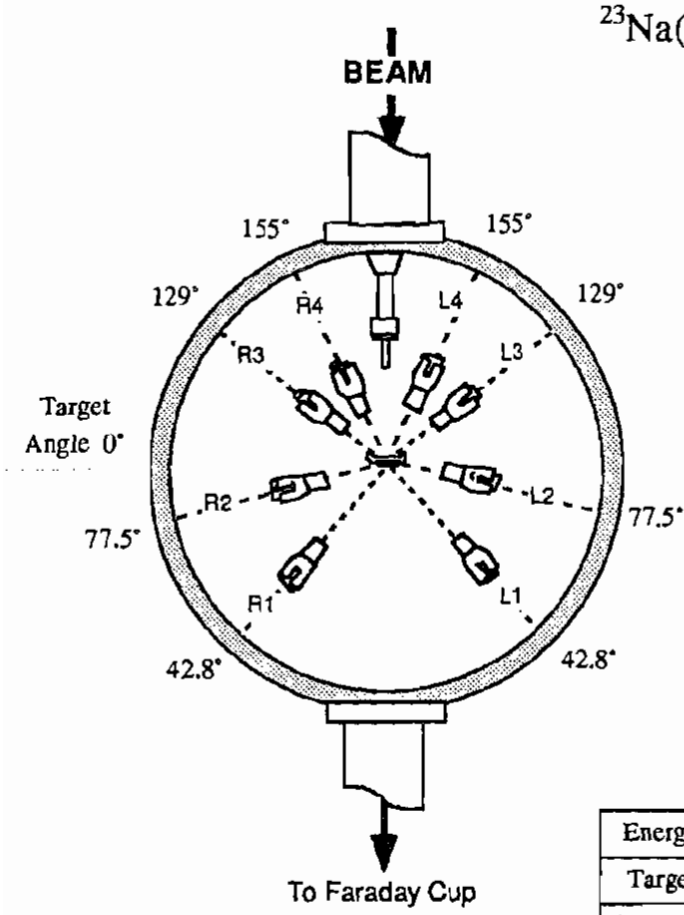
[L2 blocked at this position]

P_ℓ Zeros (lab angles)			
E_p = 3.90 MeV	P_{odd}	P_2	P_3
$^{23}Na(p,p)^{23}Na$	87.5°	52.8°	139°
$^{23}Na(p,\alpha)^{20}Na$	85.8°	51.4°	138°

	Step	Ramp	
Energy Range (MeV)	3.896 - 3.915	3.900 - 3.911	3.905 - 3.911
Energy Increment (eV)	1,000 & 500	89	54
Target Ramp Voltage (kV)	off	0 to -5.63	0 to -3.40
Number of bits in ramp	N/A	64	64
^{23}Na Target Thicknesses ($\mu g/cm^2$)	2.8	1.1 - 2.5	
^{35}Cl Contamination ($\mu g/cm^2$)	4.2	1.6 - 3.7	
Average Beam Intensity (μA)	1.0		
Q-value for $^{23}Na(p,\alpha)^{20}Na$ (MeV)	2.377		

Figure [6.6a] : First experimental setup for the study of the lowest $T = 2$ state in ^{24}Mg.

$^{23}Na(p,p)^{23}Na$ / $^{23}Na(p,\alpha)^{20}Ne$

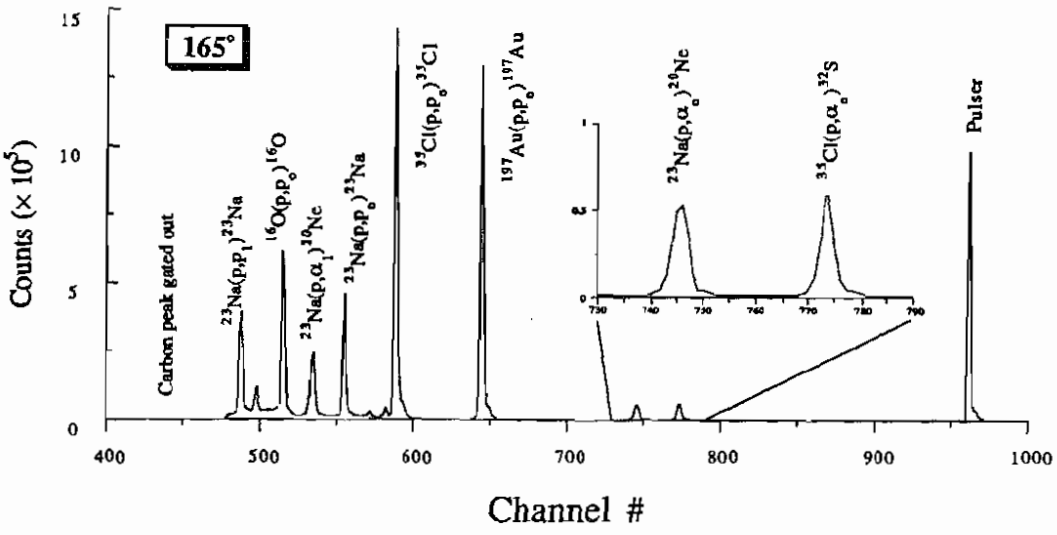


Angle Set #2

Detector	Lab Angle	dΩ (msr)	Angular Acceptance
R1 + L1	42.8°	3.0 + 3.0	2.2°
R2 + L2	77.5°	6.2 + 6.2	3.2°
R3 + L3	129°	6.2 + 6.2	3.2°
R4 + L4	155°	6.2 + 6.2	3.2°

Energy Range (MeV)	3.902 - 3.908
Target Ramp Voltage (kV)	0 to -5.63
Energy Increment (eV)	89
Number of bits in ramp	64
^{23}Na Target Thicknesses ($\mu g/cm^2$)	1.1 - 1.3
^{35}Cl Contamination ($\mu g/cm^2$)	1.6 - 1.9
Average Beam Intensity (μA)	1.0
Q-value for $^{23}Na(p,\alpha)^{20}Na$ (MeV)	2.377

Figure [6.6b]: Second experimental setup for the study of the lowest $T = 2$ state in ^{24}Mg.



Typical Target Composition	
Active Layer:	2.3 μg/cm^2 of ^{23}Na
	3.4 μg/cm^2 of ^{35}Cl
Backing Layer:	4.1 μg/cm^2 of Carbon
	2.5 μg/cm^2 of Gold (for some targets)

Figure [6.7]: Typical exit energy spectrum and target composition for the $^{23}Na(p,p)^{23}Na$ / $^{23}Na(p,\alpha)^{20}Ne$ experiment. The bombarding energy is 3.9 MeV and the spectrum energy calibration is 6.0 keV/channel.

= 3.905 Mev is the only significant structure observable in the $^{23}Na(p,p)^{23}Na$ reaction in the E_p = 3.896 - 3.915 MeV energy range. We also note the presence of a resonance structure in $^{35}Cl(p,p)^{35}Cl$ at approximately the same energy. Both these resonances result from true variations in the respective cross sections,[11] and both were observed by McDonald, *et al.* [McD78]. Several ramp scans of approximately 5 keV each were obtained in this energy region. The total energy region covered, step sizes, and other pertinent information are summarized in figure [6.6a].

The overlapping ramp scans proved to be somewhat of a nuisance since at least one region of overlap was centered on the resonance structure. However, the data were normalized together over their common energy ranges, and the resulting curves appear free from major discontinuities. A more serious aggravation resulted from several runs being taken in energy steps which differ significantly from the bulk of the data. Fortunately, most of these data were on the upper wings of the resonance region, and they were simply averaged with the nearest points[12] of the primary resonance scan data. Since only points of the primary scan (larger energy steps) were considered, this procedure may result in the loss of some of the data points of the finer energy scan. The noticeable effect of adding several scans is that the errors are unequal for different energy regions. However, this should not be a problem for the analysis program, and the increased precision in the resonance region should result in greater sensitivity of the search to the resonance parameters.

---

[11] The $^{23}Na(p,p)^{23}Na$ and $^{35}Cl(p,p)^{35}Cl$ peaks are of course well separated in the exit energy spectra, and the counting rates were fairly low. Thus, one resonance should not be artificially produced from electronic dead time or background variations resulting from the other.

[12] Actually, we also attempted a cubic spline interpolation of the data to the standard energy step size. We found very little difference in the results of this interpolation and the simple averaging procedure.

6.3.1 Resolution Function

The total resolution function is the convolution of the incident beam energy spread distribution, the target energy loss function, and a Gaussian energy distribution which may approximate the effect of Doppler broadening. Each of these contributions is discussed in general in chapter III. For the $^{23}Na(p,p)^{23}Na$, $^{23}Na(p,\alpha)^{20}Ne$ experiment, estimations of the specific functions will be discussed in the following sections, and the individual distributions, as well as the convolution result, are depicted in figure [6.8].

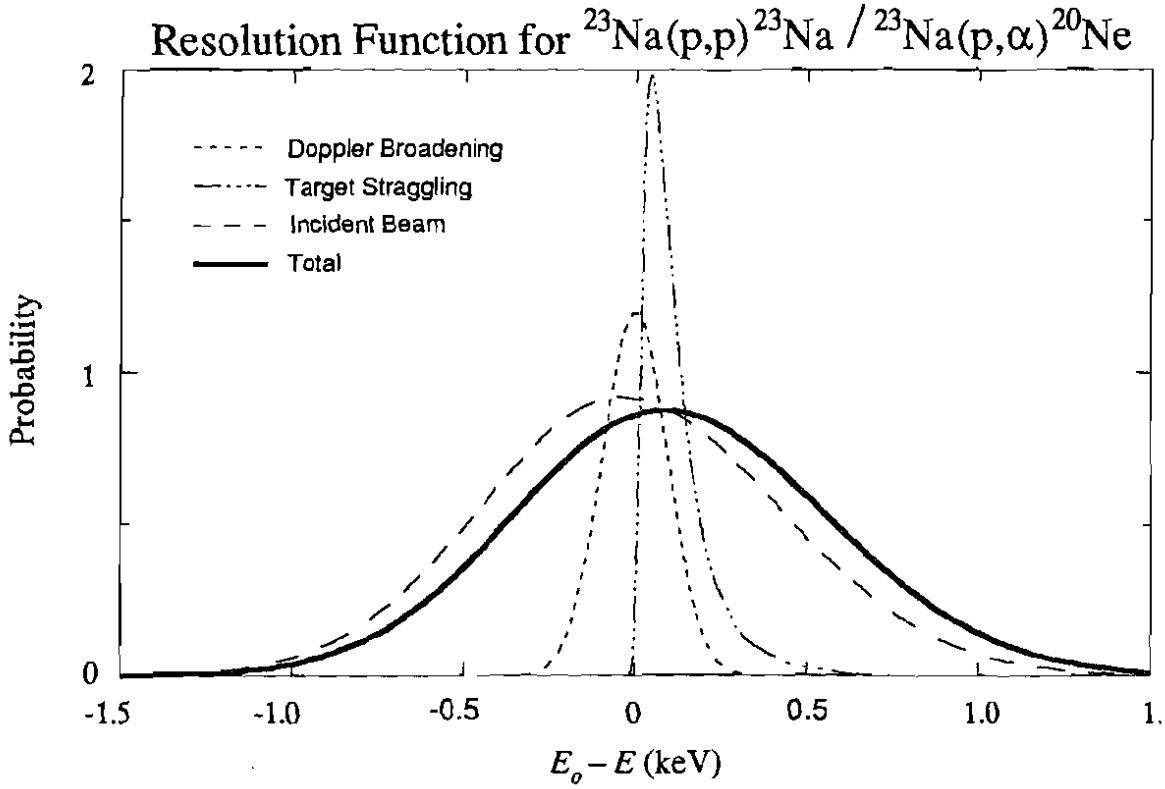
6.3.1.1 Target Straggling

The target resolution function is obtained by calculationally dividing the target into three sections and computing the straggling in each. There were no special problems associated with the sodium targets, and the programs and procedures used are as described in section 3.2. Since the target material is a compound, the thickness of the chlorine must also be taken into account for the straggling calculation.[13] Straggling in a compound is accurately approximated by considering each element separately as if in layers [Chu78]. The thicknesses and projectile energies used for the calculation are given in tabular form in table [6.1], column [b].

6.3.1.2 Intrinsic Beam Energy Spread for the 3.9 MeV Proton Beam

The beam energy profile just prior to entering the target has been measured using the ramp-ramp system outlined in section 3.1.1. Observed with the usual target ramping

[13] Only those contaminants which are mixed in with the sodium or are built up on the surface of the target need to be considered. In particular, the carbon foil backing is not important since it is not traversed by the particles before the scattering event.



	Incident Beam	Target Straggling	Doppler Broadening	Total Resolution Function
Description	Joined half-Gaussians	Sum of Vavilov Distributions	Gaussian	Convolution Result
Mean $[\mu_1']$ (keV)	0.00 *	0.119	0.00 *	0.119
Variance $[\sigma^2 = \mu_2]$ (keV^2)	0.189	0.00973	0.00912	0.208
Skewness $\left[\gamma_1 = \mu_3/\mu_2^{3/2}\right]$	0.0963	1.79	0.00	0.101
Kurtosis $\left[\gamma_2 = \left(\mu_4/\mu_2^2\right) - 3\right]$	–0.00456	3.96	0.00	0.00481

$\mu_n' \Rightarrow n^{th}$ algebraic moment $\mu_n \Rightarrow n^{th}$ central moment *set to this value

[For Normal Distribution, FWHM, $\Gamma = 2\sqrt{2\ln 2}\,\sigma = 2.355\sigma$]

Figure [6.8] : Resolution function and its constituents for the $E_p = 3.905$ MeV proton beam for the $^{23}Na(p,p)^{23}Na$ / $^{23}Na(p,\alpha)^{20}Ne$ experiment. (For an explanation of the distribution prameters and a description of the notation used, see [Ead71].)

system, a narrow resonance in $^{50}Cr(p,p)^{50}Cr$ at approximately E_p = 3.902 MeV is used as the energy marker (figure [6.9]). To obtain the beam energy spread function, the so-called trim ramp[14] samples the different particle energies in the incoming beam and causes this resonance structure to change positions in the excitation function spectra. A problem with the trim ramp controller and MBD interface prevented us from taking data at more than one angle (155°); however, the data obtained provide very precise values of the resonance energy. Each magnet setting (corresponding to one of the sixteen bits in the trim ramp) gives rise to a separate excitation function in which the $^{50}Cr(p,p)^{50}Cr$ resonance appears to have a slightly different resonance energy. This shifting in energy is depicted in figure [6.10] for several different ramp steps. Each of the sixteen different resonances is fitted with a Breit-Wigner resonance shape utilizing the same programs as used to fit the T = 2 level data.[15] The fits are generally very good, with chi-squared per degree of freedom between one and two. Also, the fits are very sensitive to the resonance energy (except the function corresponding to ramp bit 16 which has very poor statistics), with errors in the 20 to 100 eV range obtained from $\Delta\chi^2 = 1$. The sixteen values of resonance energy and errors so obtained serve as x-axis values for our experimental energy spread distribution.

---

[14] The trim ramp is a triangular waveform which modulates the current in the second 90° analyzing magnet via the small trimming power supply. This modulation produces a changing field which sweeps the beam from side to side. Since the beam is energy analyzed at the magnet exit, the effect is to sample particles of slightly different energies for each value of the ramp. Generally the ramp has sixteen steps (converted from digital bits) for each ramp cycle and a period of about a minute.

[15] Note that the resonance energy is used here only as an energy marker, and we are only interested in relative positions of the individual resonances. Thus, there can be many possible choices of fitting functions, and there need not be any nuclear physics involved. That is, a simple spline fit which reproduces the shape may give the necessary information. The choice to use the resonance fitting programs was based on convenience and the hope that the MINUIT search routines would provide some kind of reasonable error estimates for the resonance energy.

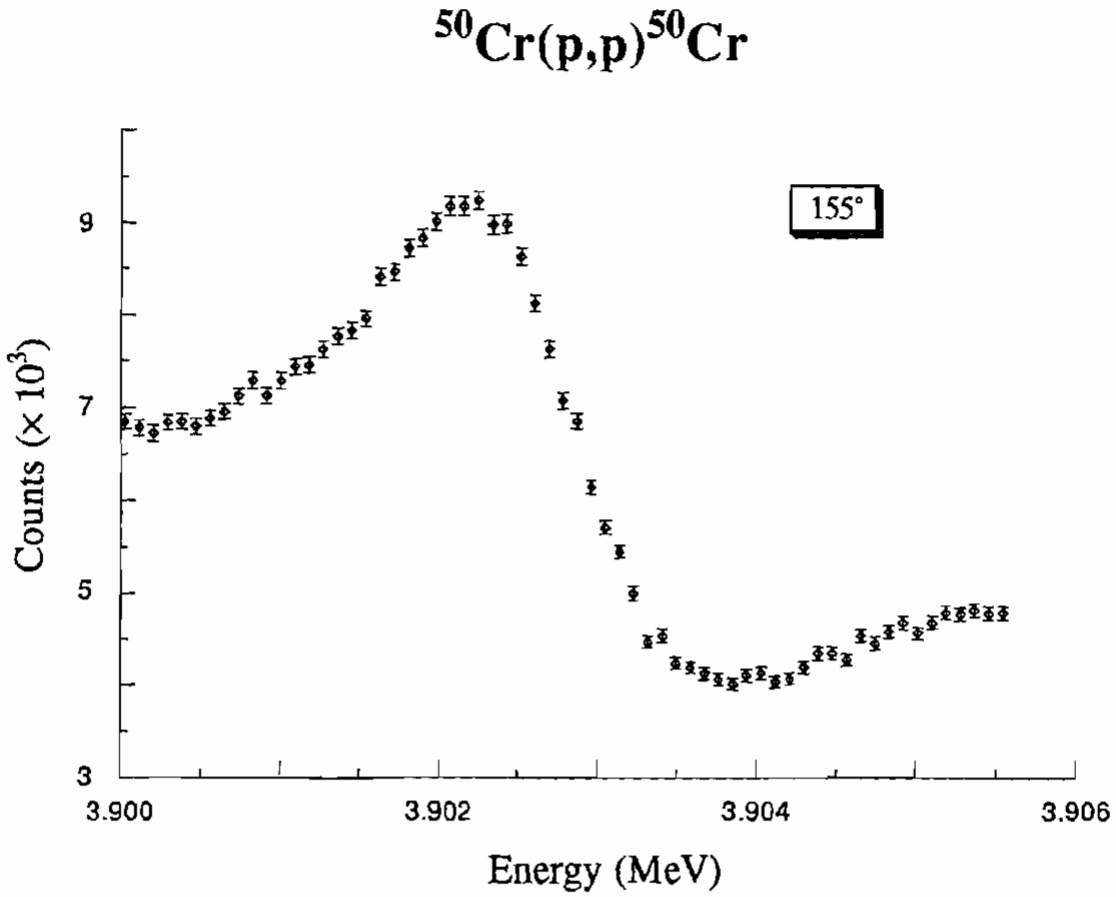


Figure [6.9]: $^{50}Cr(p,p)^{50}Cr$ resonance used as an "energy marker" for the resolution function measurement.

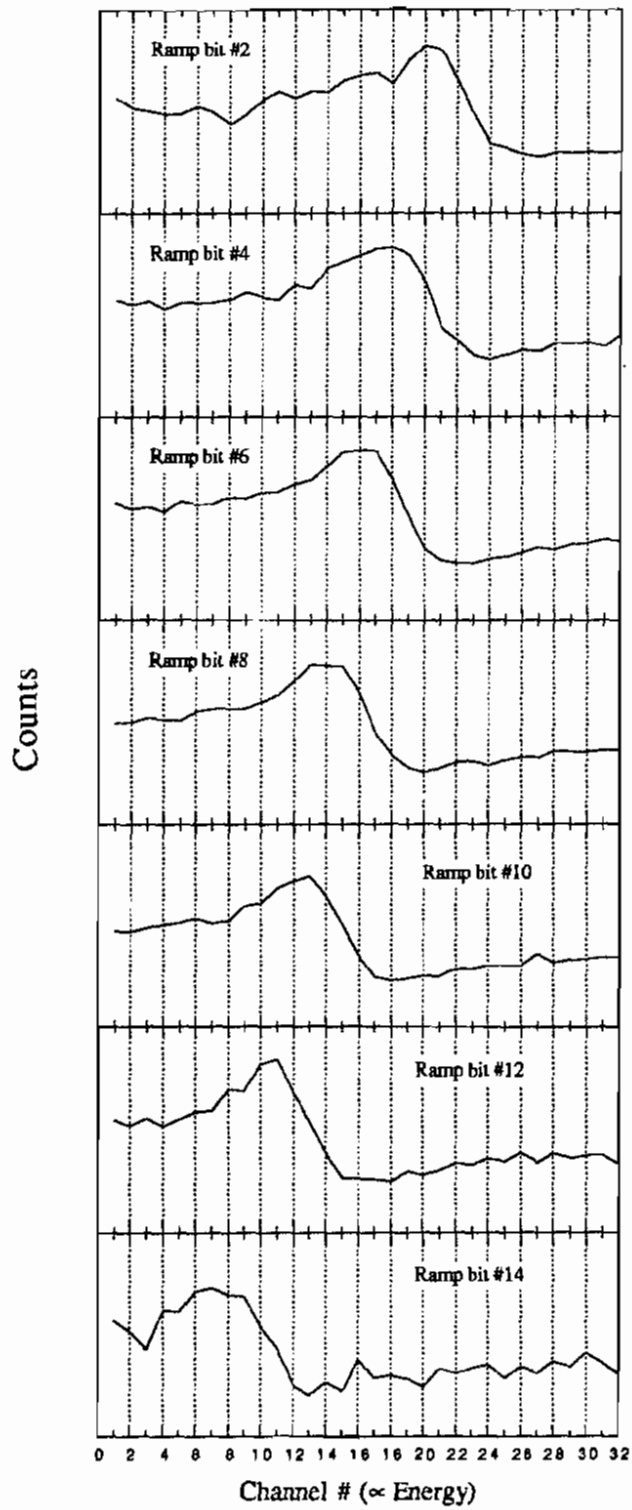


Figure [6.10]: Energy shift of $^{50}Cr(p,p)^{50}Cr$ resonance during the resolution function direct measurement. A shift of one channel corresponds to an energy change of 178 eV.

For the independent variable we need a measure of the relative number of particles incident on the target for each of the sixteen settings of the trim ramp. A tungsten contaminant provides the means to extract this value for relative counts. The $^{NAT}W(p,p)W$ reaction provides a high count rate to minimize statistical uncertainties, and we can verify, by looking at its excitation function, that it is non-resonant in this energy region. The resulting data distribution for relative counts versus resonance energy is shown in figure [6.11]. The error bars in the x direction indicate the errors on E_r given by MINUIT for $\Delta\chi^2 = 1$; errors in the y direction are square root errors and, as such, are generally smaller than the point size.

Although the data distribution itself may be directly convoluted with the other resolution distributions, fitting the data to a functional form insures smoothness and allows the function to be extrapolated beyond the range of the experimental x-values. We have chosen a fitting procedure that relies on reproducing the moments of the data distribution over the x-range of the data.[16] The first four moments of the data are considered, and they are given by:

$$\mu'_1 = 3902.8 \text{ keV}$$

$$\mu_2 = 0.181 \text{ keV}^2$$

$$\mu_3 = 0.00727 \text{ keV}^3$$

$$\mu_4 = 0.106 \text{ keV}^4$$

where μ'_n is the n^{th} algebraic moment and μ_n is the n^{th} central moment.

Since the data exhibit a slight skewness, indicated by a non-zero third central moment, we use a functional form capable of producing an asymmetric distribution. The

---

[16] For this particular experiment the data are probably good enough to fit point-for-point. However, the procedure of fitting to the moments had to be employed for the $^{24}Mg(\alpha,\alpha)^{24}Mg$ data because of the poor quality of the extracted experimental beam resolution function. Thus, mostly for convenience, the programs and fitting procedures are the same for both experiments.

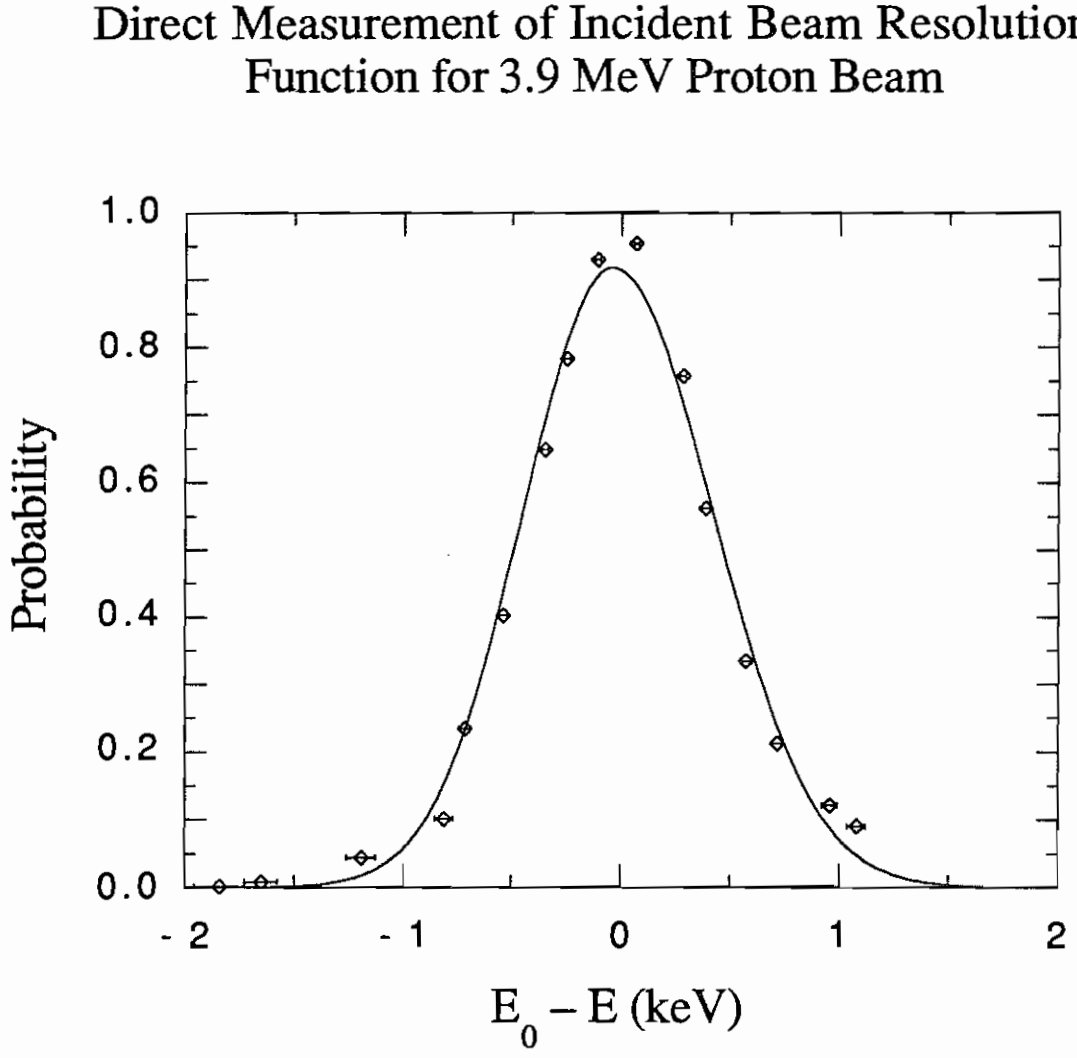


Figure [6.11]: Resolution function data and fit for the 3.9 MeV proton beam used in the $^{23}Na(p,p)^{23}Na$ / $^{23}Na(p,\alpha)^{20}Ne$ experiment. Here E_0 is the mean energy.

chosen distribution consists of two joined half Gaussians; that is,

$$\begin{aligned} & e^{-\alpha(x-x_0)^2} \qquad x \le x_0 \\ & e^{-\beta(x-x_0)^2} \qquad x > x_0 \quad . \end{aligned} \tag{6.1}$$

We acknowledge the fact that the difference in our function and a Gaussian is almost completely negligible in this case, but the programs are prepared for use on the more complicated case to follow. As such, the programs to fit counts versus energy to a functional form will be described more completely in section 6.4.2.2 for the resolution function in the $^{24}Mg(\alpha,\alpha)^{24}Mg$ experiment. The fit shown in figure [6.11] is the joined half-Gaussians incident beam resolution function, and it is shown along with the other resolution distributions in figure [6.8]. The near-normal fit to the data has a FWHM of about 1 keV, somewhat larger than expected from previous measurements and predictions and probably indicates poor operation of the fast-feedback system. We must keep in mind, however, that this value, strictly speaking, represents the beam distribution at a point between the two 90° magnets where the system resolution (after the first analyzing magnet) yields a beam energy width of about 1.88 keV. Thus, a notable reduction in the beam energy spread, resulting from the slit feedback system, is indicated. Moreover, since the total energy resolution of the 90-90 system is about 940 eV, we should not expect the resolution to improve significantly from the center 90-90 slits to the target.

6.3.1.3 Doppler Broadening for p +^{23}Na

The target temperature for our average NaCl target can be estimated in the usual manner, and the result is, as expected, about 10% higher than room temperature. However, although the Debye temperature for NaCl is tabulated, as mentioned in chapter III, our method of estimating the width of the Doppler broadening distribution is not valid for targets which exist as compounds. In particular, the Debye and Einstein

approximations for the mean quantal energy break down because of the introduction of the so-called optical modes of vibration, present whenever there is more than one atom per unit cell. Because of the availability of these additional vibrational frequencies, one might expect that approximations based purely on the acoustical modes (Einstein and Debye) would be too small and that the Doppler broadening for a compound would be greater than that of any single component.[17] However, the action of the molecular binding and the complicated sharing of energies possible between the atoms makes it difficult to speculate about the energy of a single constituent.

Nevertheless, lacking detailed knowledge of the phonon spectra, and cognizant of the many other errors involved in determining the resolution function, we proceed with the calculation as outlined in section 3.3, using the Debye temperature of NaCl. In any case, since the shape of the distribution is normal, and since an adjustable resolution function width is allowed in the analysis, we accept the inaccuracy of the calculation and view the solution as an approximate starting value. The relevant parameters and results of the calculation are given in the second column of table [6.2], and the distribution is shown as part of figure [6.8].

6.4 Lowest $T = 2$ State in ^{28}Si

The energy position of the resonance corresponding to the lowest $T = 2$ level in ^{28}Si has been established by the ^{24}Mg$(\alpha,\gamma)^{28}$Si reaction as $E_\alpha = 6.115 \pm 0.003$ MeV [Sno69]. A later experiment by Ikossi, *et al.* [Iko79] used elastic alpha scattering on ^{24}Mg to study this ^{28}Si $T = 2$ state. This experiment located the resonance at $E_\alpha = 6.114 \pm 0.005$ MeV and utilized good experimental energy resolution to obtain $\Gamma_\alpha = 222 \pm 22$ eV

[17] For example, it has been shown that the Doppler broadening distribution for $Li^{15}NO_3$ is about twice as wide as that for atomic nitrogen (^{15}N) at 100 K [Mor76].

for the resonance partial width (using the value for the branching ratio from [Fre79]). Because of this existing precise measurement and the simplicity of the analysis (all constituent particles and states have $J^{\pi} = 0^{+}$), our $^{24}Mg(\alpha,\alpha)^{24}Mg$ experiment figured to be one of verification (with somewhat improved experimental resolution), both of the previous experimental results and of our analysis methods. However, in many ways, the study of this $T = 2$ state in ^{28}Si was the most difficult and most interesting of the three experiments related in this work.

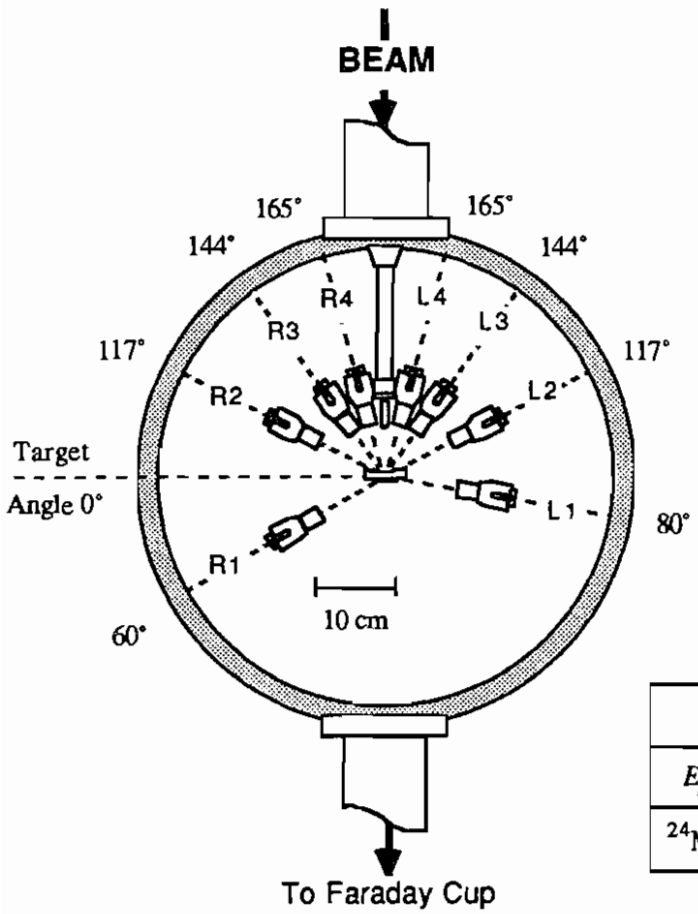
Experimentally, much of the difficulty lay in conjuring up a uniformly thin ^{24}Mg target. Magnesium target making woes are well chronicled in chapter IV, and we will not repeat them here except to note that poor quality targets are the main reason that this experiment was the first attempted and, after several tries over several years, the last completed.[18] Other problems with very low count rates and low beam currents on target contributed to the several less-than-successful experiments. Also, much time was involved in revamping and experimenting with different modes of operation of the alpha source. Generally, however, each measurement was better than the preceding one, and the information built up during these preliminary runs proved very helpful in setting up and performing the final experiment.

With a beam of doubly charged alpha particles[19], data were taken at five angles from 60 to 165 degrees in the energy range $E_{\alpha} = 6.100$ to 6.123 MeV. The experimental

---

[18] Actually, it is impossible to tell for sure if the poor resolution in the early experiments is the result of the target quality alone. However, since no abnormalities were observed in the experimental setup and high resolution system, and since the difficulties in making magnesium targets are well documented, the target uniformity is almost certainly to blame for the resolution inadequacy.

[19] The beam consisted of α^{++} since we had determined, from comparing "target-in" and "target-out" beam currents, that singly charged alphas are stripped of their remaining electrons as a result of passing through the target. Thus, a beam made up of α^{+} would yield erroneous values for the integrated beam current.



$^{24}Mg(\alpha,\alpha)^{24}Mg$

Detector	Lab Angle	dΩ (msr)	Angular Acceptance
R1	60°	5.0	2.9°
L1	80°	5.0	2.9°
R2 + L2	117°	5.0 + 5.0	2.9°
R3 + L3	144°	7.8 + 7.8	3.6°
R4 + L4	165°	7.8 + 7.8	3.6°

P_ℓ Zeros (lab angles)			
E_p = 6.11 MeV	P_{odd}	P_2	P_4
$^{24}Mg(\alpha,\alpha)^{24}Mg$	80.5°	116.7°	143.6°

Energy Range (MeV)	6.100 - 6.123	Energy Increment (eV)	180
Target Ramp Voltage (kV)	± 5.72	^{24}Mg Target Thickness (μg/cm^2)	1.35
Scanning range for α^{++} (keV)	± 11.44	Ta Contamination (μg/cm^2)	0.48
Number of bits in ramp	128	Average Beam Intensity (μA)	0.3

Figure [6.12]: Experimental setup for the study of the lowest $T = 2$ state in ^{28}Si.

setup and parameters are summarized in figure [6.12]. Only one target was used for the final data, and its thickness was deduced by comparison to targets of known thickness, determined from Rutherford scattering of alpha particles on magnesium at 60° at 5 MeV. Monitoring of the carbon buildup showed an increase of ≤ 5% over the course of the experiment. The target composition and a representative exit energy spectrum is given in figure [6.13].

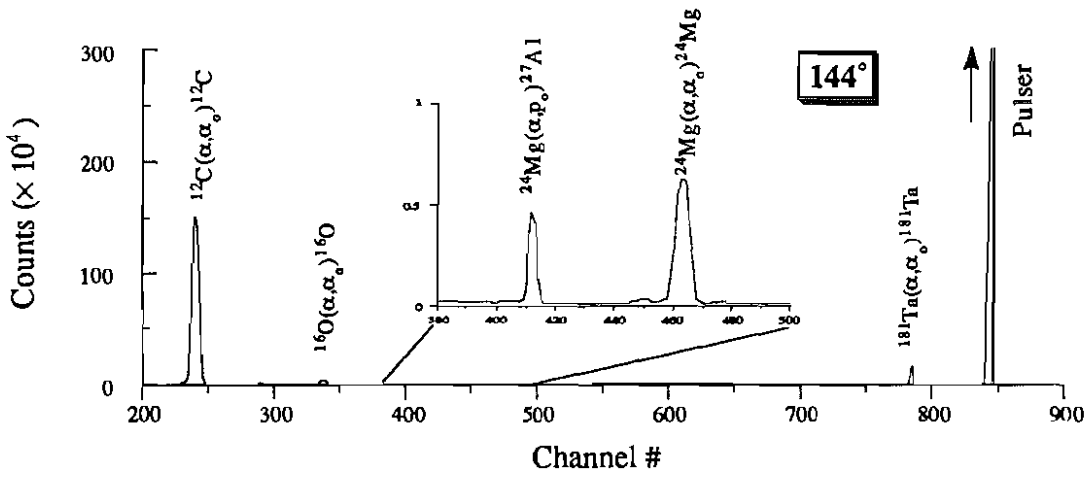
From our previous experience, we had learned to expect a rather large (relatively) background plateau in the 165° detector spectra beginning at almost the same energy as the $^{24}Mg(\alpha,\alpha)^{24}Mg$ peak. This background is attributed to two elastic scattering events from carbon, which occur simultaneously and are not resolved by the electronics. That is, the energies of two alphas from $^{12}C(\alpha,\alpha)^{12}C$ are added together to yield the same energy as one event from $^{24}Mg(\alpha,\alpha)^{24}Mg$. The very small cross section (< 1 mb/sr) for the alpha scattering on magnesium at 165° accounts for the relative importance of this intruder.[20] As a remedy we installed 150 $\mu g/cm^2$ carbon foils[21] on the detector collimators (at 165°) to degrade the energy of the scattered alpha particles.[22] The alphas scattered from carbon, by virtue of their lower overall energy, will lose more energy proportionately than the alphas from magnesium. As a result the background ridge will move away from the peak of interest. This is illustrated in figure [6.14]. Without this correction, the data for

---

[20] It is interesting to note that when studying a resonance in $^{24}Mg(\alpha,\alpha,)^{24}Mg$ nearby the $T = 2$ state ($E_r = 5992 \pm 5$ keV), Ikossi et al. [Iko79] record "an unidentified background which was present at the 165° spectra and was not present at other energies or angles".

[21] We began with 300 $\mu g/cm^2$ degrading foils but found that the kinematic shifts put the $^{24}Mg(\alpha,\alpha)^{24}Mg$ peak on top of the $^{24}Mg(\alpha,p)^{27}Al$ peak at 165°.

[22] Since the alphas pass through the foil after they are scattered, this process affects only the detector resolution, not the resolution of the incoming beam, which is important to the resonance measurement. The effect on the width of the exit energy peaks (already many tens of keV) is negligible.



Target Composition	
[only one target (W40) was used for the final data]	
Active Layer:	1.35 μg/cm^2 of ^{24}Mg
	0.48 μg/cm^2 of Tantalum
Backing Layer:	4.0 μg/cm^2 of Carbon

Figure [6.13]: Typical exit energy spectrum and target composition for the $^{24}Mg(\alpha,\alpha)^{24}Mg$ experiment. The incident projectile bombarding energy is 6.1 MeV and the spectrum energy calibration is 7.2 keV/channel.

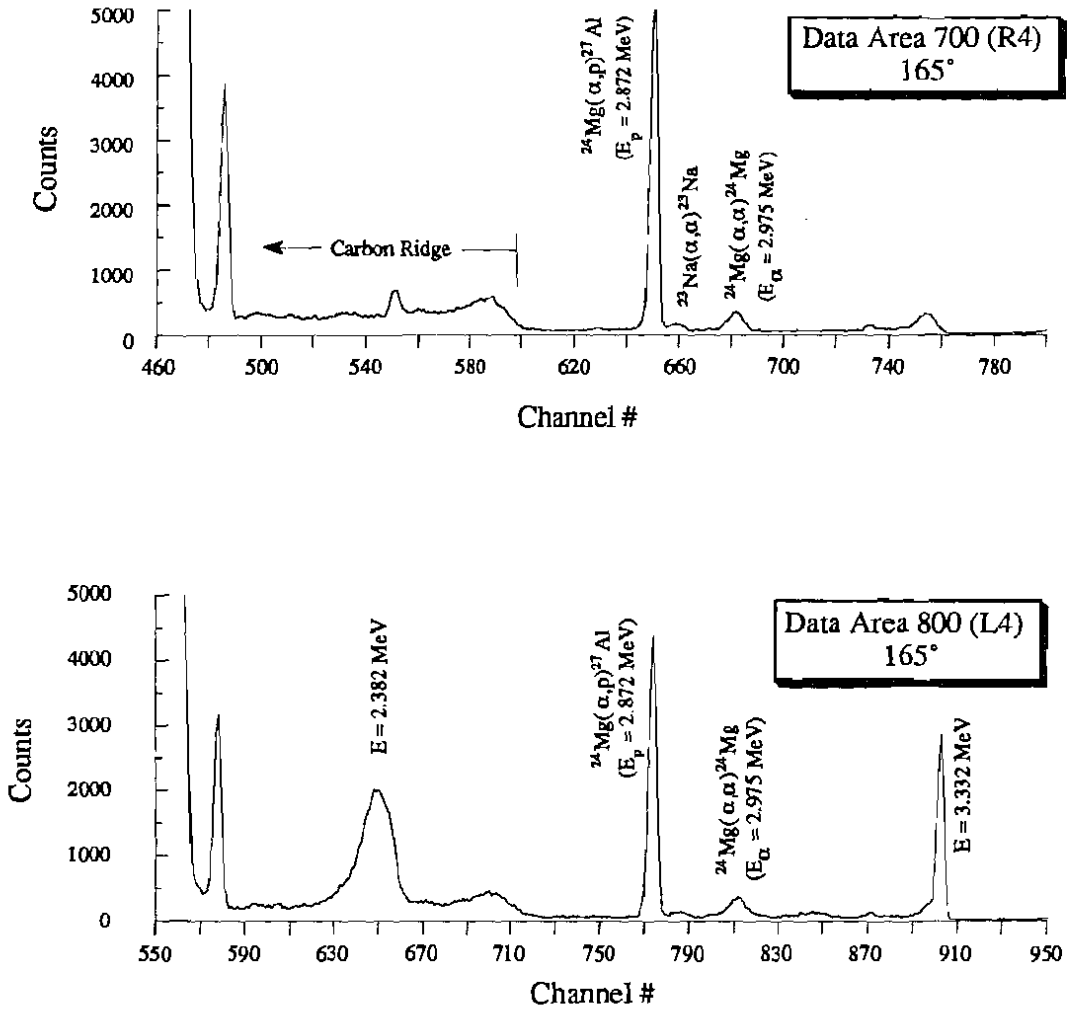


Figure [6.14]: Exit energy spectra for the right and left detectors at 165° showing the background "carbon ridge" attributed to two particle scattering from carbon. Also shown are the unidentified contaminant peaks occurring mainly in the left detector. Both detectors are equipped with 150 $\mu g/cm^2$ carbon degrading foils, and the energies given are those after passage through this foil. The incident projectile bombarding energy is 6.1 MeV, and the energy calibration is approximately 4 keV/channel for both spectra.

165° would be almost worthless in determining resonance parameters.

It may also be noted in figure [6.14] that there are two unidentified peaks in the spectra which are interesting because they appear much larger in the left detector than in the right. Like the apparent double carbon pileup, these right-left asymmetric peaks have also been observed in previous experiments with different detectors, electronics, and targets. However, since these peaks do not interfere with our ability to analyze the elastic scattering data, they are considered only as a mild curiosity, and no serious attempts were made at identification.

6.4.1 Pygmies

In addition to the curious artifacts present in the exit energy spectrum, there exists an unrelated, but very interesting, manifestation lurking in the excitation functions themselves. Although discussions of the excitation functions are properly the domain of chapter VII, this aspect of the data was not considered in the final fits and as such will be discussed here. There seems to be a small secondary bump a few keV above the $T = 2$ resonance (figure [6.15]). This structure, coined "pygmy resonance," is most noticeable at the more backward angles, although some observers deny its existence entirely. An effect of this kind has been observed in several different runs over a period of several years and with completely different targets and experimental setups.

Assuming that **something** is going on in this energy region, we can speculate as to the possible causes of such an effect. The observation of this artifact in more than one experiment tends to decry the possibility of an experimental flaw since there were different detectors, targets, scattering angles, electronics, etc. Along these same lines, the target ramp would have to be a prime candidate in producing such a false peak. However, the diagnostic "ramp excitation functions" produced by the ramp and a pulser

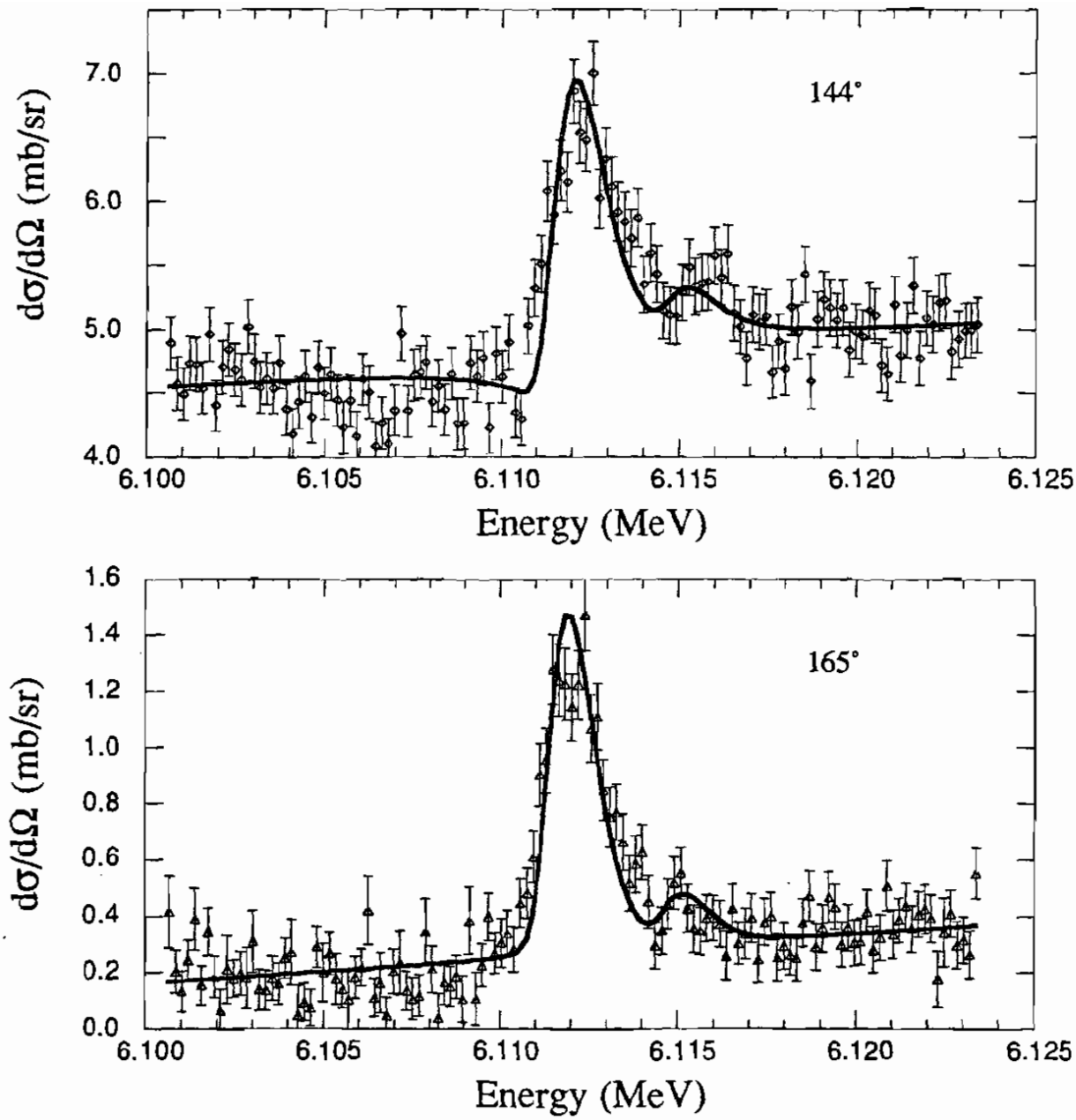


Figure [6.15]: Back angle scattering for $^{24}Mg(\alpha,\alpha)^{24}Mg$ showing the "pygmy" structure just above the main resonance corresoponding to the lowest T = 2 state in ^{28}Si. The fits are based on the simple model outlined in section 6.4.1 using Δ_k = 3.2 keV and P_k = 0.12.

are flat to within statistical errors.

Perhaps the bump has some physics origin like a contaminant in the target. The isotopic purity of the targets is >99.9%, and the chemical purity of the pre-evaporated material is also very high (99.9%). On the other hand, chemical contaminants can infiltrate the target during the evaporation or preparation of the substrate, so there are very likely to be some contaminants present in the target. However, kinematic calculations at 165° give the α-particle exit energy for $^{25}Mg(\alpha,\alpha)^{25}Mg$ to be 86 keV higher and that of $^{23}Na(\alpha,\alpha)^{23}Na$ to be 91 keV lower than the alpha energies from $^{24}Mg(\alpha,\alpha)^{24}Mg$ (Na is most likely nearby chemical contaminant). Since the energy calibration of the 165° spectrum is ~ 4 keV/channel and the observed exit energy peaks have a base width of ≤ 20 channels and a FWHM of about 10 channels, the contaminant peaks would be well resolved. In fact, we can see a small peak in the 165° spectrum corresponding to $^{23}Na(\alpha,\alpha)^{23}Na$. We conclude that there are no elastic alpha scattering peaks which could cause a problem, and inelastic scattering from a contaminant is unlikely to be a factor since the effect is seen at more than one angle.

Another possible cause of a secondary bump is another nuclear resonance in $^{24}Mg(\alpha,\alpha)^{24}Mg$ at this energy. We can quickly (although not quite certainly) dismiss this theory since the structures in question have narrow widths comparable to the width of the $T = 2$ state. Other nearby resonances would be expected to represent $T = 0$ states and as such have widths very much larger (several keV) than what is evidenced.

Finally, we can postulate about so-called atomic effects. Such effects as projectile interactions with atomic electrons have long been known to exist and have been noted experimentally (e. g. Lewis effect [Lew62]). Such effects would produce a secondary peak with the same width as the primary resonance but shifted up in energy corresponding to the energy lost by the projectile. We note for the record that the K-shell ionization energy for ^{24}Mg is approximately 1.8 keV ($= [Z - 1/2]^2 \times 13.6$ eV). From

inspection of the data this value seems too small to coincide with the pygmy bump. However, the K-shell ionization energy, E_k, for ^{28}Si is 2.5 keV, which is getting closer to the observed pygmy position. We could also consider such things as double K-shell ionization, but the probability for such occurrences would be expected to be very small.

Whatever the cause, we are free to try to construct a model which will reproduce the data. If we add the pygmy resonance as an incoherent term in the total cross section, we have

$$\sigma_{total} = \sigma_{nuc}(1 - P_k) + \sigma_{pygmy}P_k \qquad [6.2]$$

where P_k is the probability for the pygmy and σ_{pygmy} is calculated the same as σ_{nuc} except with $E \rightarrow E + \Delta_k$, the energy shift being Δ_k. For unconstrained fits of all five angles, the best fits occur for $\Delta_k \rightarrow 0$, so that P_k becomes irrelevant. This, in effect, is making the resonance peak a little wider. If the fits are constrained to 1 keV $< \Delta_k <$ 6 keV and $0 < P_k < 0.4$ then the fit converges to Δ_k = 3.5 keV and P_k = 0.053. The observed fit is not noticeably changed due to the small value of P_k. We conclude from this that, although there seems to be some structure just above the resonance, our data are not quite good enough (in statistics or in resolution) to extract any further quantitative information as to its origins. As a final note, the aforementioned secondary structure does seem to affect the $T = 2$ resonance analysis by causing the fits to include some of these points at the expense of expanding the resonance width. We will comment on errors of this type in chapter VII.

6.4.2 Resolution Function

The energy resolution function contains contributions from the intrinsic energy spread of the beam, Doppler broadening and straggling in the target, and possible atomic processes in the scattering. In order to reproduce correctly the observed cross sections,

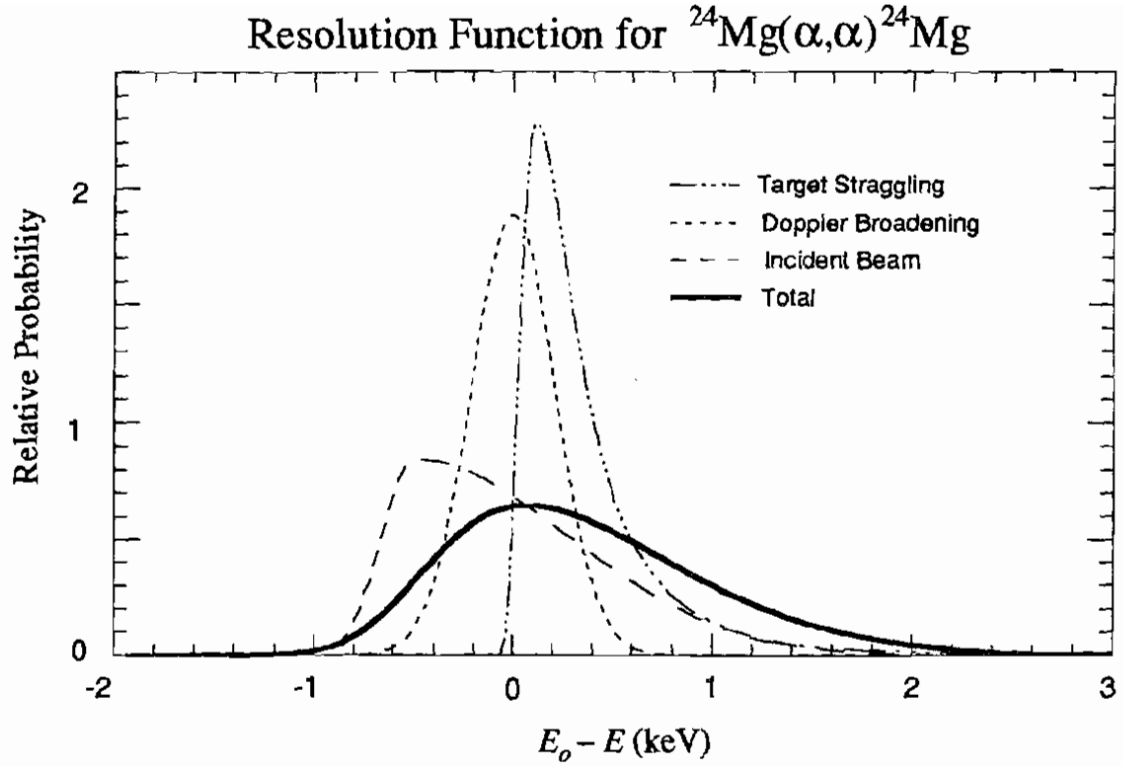
these effects must somehow be taken into account. From measured target thicknesses, straggling effects can be approximated with methods of section 3.2, and, assuming a Gaussian shape, the width of the Doppler broadening distribution can be estimated from knowledge of the solid state structure of the target material. Finally, the incident beam energy spread can be approximated through a measurement of the type described in section 3.1. The actual distributions and their convolution result are given as part of figure [6.16].

6.4.2.1 Target Straggling

The target straggling distribution is again calculated by the program TS, and the relevant parameters are given in table [6.1], column [c]. Note that for the α^{++} beam the straggling is an important contribution to the total resolution function, as can be seen qualitatively in figure [6.16]. We note that if the magnesium has oxidized, the actual target distribution should be even wider as a result of this extra contaminant. Since we have no way to accurately estimate the MgO content, we will use the distribution calculated and keep in mind that its present width may represent more of a lower limit.

6.4.2.2 Intrinsic Beam Resolution for the 6.1 MeV α -particle Beam

The ramp-ramp data for the $^{24}Mg(\alpha,\alpha)^{24}Mg$ experiment include alpha elastic and $^{24}Mg(\alpha,p)^{27}Al$ reaction data obtained at several different angles from 60 to 165 degrees at a lab energy of 5.943 MeV, only 170 keV away from the $T = 2$ state of interest. As before, each of the sixteen resonances at each of the angles was fitted with a Breit-Wigner resonance shape utilizing the same programs used to fit the $T = 2$ data. During the fitting procedure, some of the fitting parameters such as the background phase were found to be highly correlated to the resonance energy. These parameters were either fixed or tightly



	Incident Beam	Target Straggling	Doppler Broadening	Total Resolution Function
Description	Joined half-Gaussians	Sum of Vavilov Distributions	Gaussian	Convolution Result
Mean $[\mu_1']$ (keV)	0.00 *	0.362	0.00 *	0.362
Variance $[\sigma^2 = \mu_2]$ (keV^2)	0.266	0.119	0.0449	0.432
Skewness $\left[\gamma_1 = \mu_3/\mu_2^{3/2}\right]$	0.814	2.21	0.00	0.727
Kurtosis $\left[\gamma_2 = \left(\mu_4/\mu_2^2\right) - 3\right]$	0.422	6.45	0.00	0.687

$\mu_n' \Rightarrow n^{th}$ algebraic moment $\quad \mu_n \Rightarrow n^{th}$ central moment $\quad$ * set to this value

[For Normal Distribution, FWHM, $\Gamma = 2\sqrt{2\ln 2}\,\sigma = 2.355\sigma$]

Figure [6.16]: Resolution function and its constituents for the 6.112 MeV alpha beam of the $^{24}Mg(\alpha,\alpha)^{24}Mg$ experiment. The incident beam moments are calculated only over the energy range of the data values. (For an explanation of the distribution properties and a description of the notation used, see [Ead71].)

constrained to achieve consistent values and errors for the resonance energy. Once E_r is extracted for the independent variable, the vertical axis information is provided by counts from the non-resonant $^{12}C(\alpha,\alpha)^{12}C$ scattering (from the backing foil) because of its higher count rate. Thus, the errors in the y-direction, being purely statistical, are small enough to be considered negligible in this analysis. Representative results are shown in figure [6.17]. The errors in the x-direction represent confidence intervals for which our chi-squared function changed by unity. The relatively large errors may be attributed to poor statistics from the resonant scattering[23], resulting in a relatively shallow minimum for the chi-squared function.

To provide a more usable form for the resolution function which can be extrapolated to values outside the data range, the counts versus energy data must be translated into a functional form. In fitting the data to a functional form, the large error bars in the x-direction provide a problem since the usual chi-squared function assumes errors only in the dependent variable. However, a set of programs was developed to use MINUIT to fit the resolution function to a Gaussian distribution since it can be easily inverted ("x" written as a function of "y"). It was found that although some of the data could be fit very well with a Gaussian shape, other functions showed a marked asymmetry. In order better to reproduce the asymmetrical shape, a new approach was attempted.

Using distributions from five different data sets, the first four moments of the data distributions were calculated. Average statistical errors from the data points were computed for each data set and the values used as relative weights in the determination of the average values of the moments. Errors on the moments of the data were determined

[23] The analysis was plagued throughout by insufficient statistics which forced a less-than-ideal treatment of errors. In retrospect, we should have allowed more time for the measurement or used a somewhat thicker target or both.

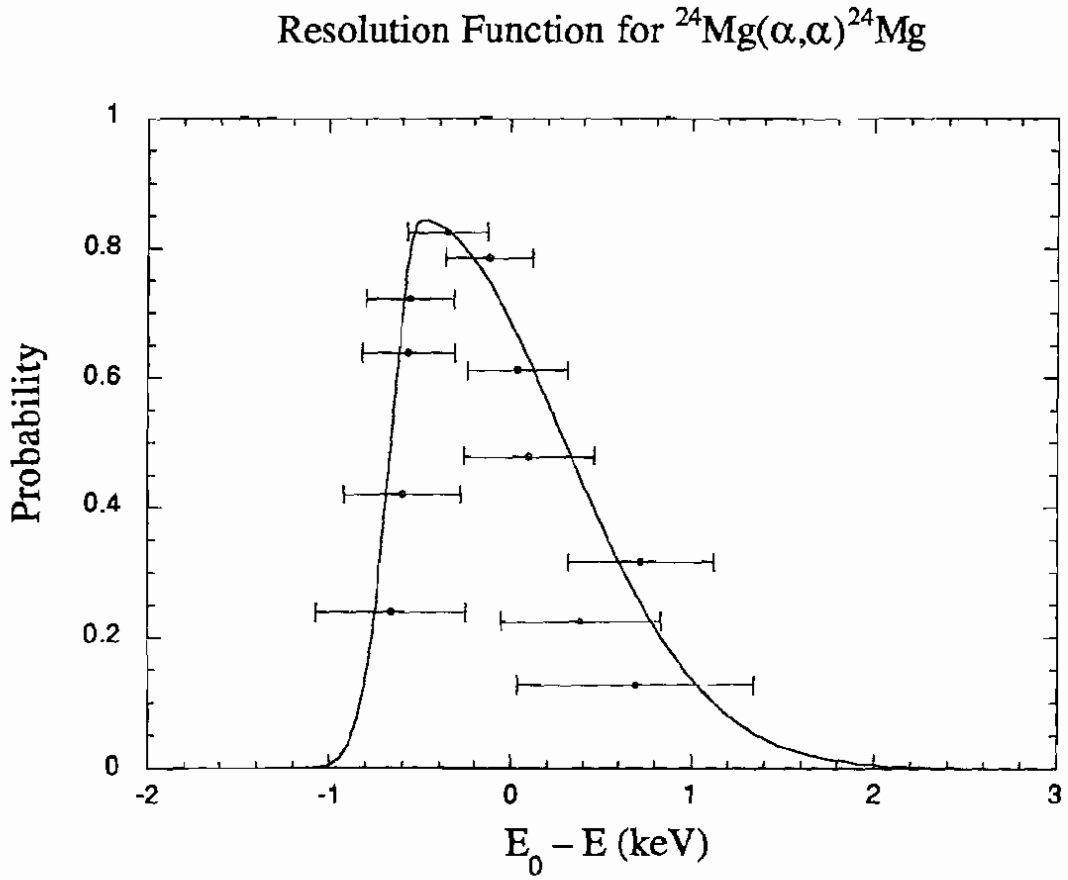


Figure [6.17]: Sample of the data and final fit for the resolution function of the 6.1 MeV α-particle beam. E_0 is the mean energy, so the low energy tail appears on the right.

based on the deviations of values between the five data sets and do not take into account the errors in specifying the energy positions of the points.

A summary of results for the data distributions is shown below:

first algebraic moment (mean)	$\mu'_1 = 6107.9 \pm 0.3$ keV
second central moment (variance = σ^2)	$\mu_2^2 = 0.245 \pm 0.096$ keV^2
third central moment	$\mu_3 = 0.177 \pm 0.236$ keV^3 *
fourth central moment	$\mu_4 = 0.449 \pm 0.484$ keV^4

* [Statistically, the so-called coefficient of skewness, $\gamma_1 = \mu_3/\mu_2^{3/2}$, is 1.46 ± 2.13]

The non-zero third central moment, although consistent with zero, suggests that the data do not follow a strictly Gaussian form. A purely "numerical" fit to the data could have been generated by convoluting a Gaussian function with another asymmetric function such as a Landau or Vavilov distribution (the output of the computer code STRAGL). However, we chose to try to fit the data to an analytical form, hoping to provide a more versatile solution, a better understanding of the errors on the fitting parameters, and a saving of computer time for the subsequent convolutions with the resonance shapes in the main search routine

The final form for the beam resolution function is that of two joined half Gaussians. That is, each half of the distribution is a Gaussian with a width independent of the other half. Furthermore, the low energy (right hand side when plotted versus resonance energy) half Gaussian may be added with a $\ln(z)/z^4$ function to increase the asymmetry:

$$\begin{cases} e^{-\alpha z^2} & z \le 0 \\ e^{-\beta z^2} + \gamma \dfrac{\ln(z+1)}{(z+1)^4} & z > 0 \end{cases} \qquad [6.3]$$

where $z = x - x_0$ and x_0, α, β, and γ are to be determined from the fitting procedure. Of course, the value of x_0 is really irrelevant since the x-values of the distribution will be shifted so that the mean is at zero for convolution with the other resolution effects. The factor $z+1$ in the denominator (instead of simply z) insures that there are no problems if z is very close to zero.

Fitting to the moments of the data distributions is complicated not only by the large error bars, but by the fact that the y-values do not equal zero on the wings of the data distribution. To compensate for this problem, the fit is calculated over the same x-range as the data values. The minimum values on the wings of the fit are less than one percent of the peak value, and the function is allowed to go to zero smoothly outside the x-range of the data. Results for the best fit are given in figure [6.17] and as part of figure [6.16]. By inspection of the figure, we see that the FWHM of the measured distribution[24] is about 1.1 keV, and we note that the upper limit on ΔE, from the known resolution of the 90-90 system slit settings, is 1.47 keV.

6.4.2.3 Doppler Broadening for $\alpha + {}^{24}Mg$

The width of the Doppler broadening distribution is again calculated from the methods of section 3.3, and the mean energy per atom is estimated by the Debye model. Although the energy loss for alpha particles in the target is substantial, the relatively low beam current limits the target heating so that the temperature remains about 10% above

---

[24] In the limiting case of a normal distribution, a variance of 0.266 keV^2 implies a FWHM of 1.2 keV.

that of the surroundings. The calculated width of the Doppler broadening (Gaussian) distribution is approximately 500 eV. The parameters used and intermediate results are given in the third column of table [6.2], and the distribution itself is shown in figure [6.16].

Since magnesium readily oxidizes in air, we acknowledge the fact that some (or all) of the Mg in the target may be in the form of MgO. In this case, the Doppler broadening calculations will be inadequate for the same reasons given in section 6.3.1.3 involving the NaCl compound target. On the other hand, if the target is elemental magnesium, we would expect a fairly good approximation for the Doppler broadening based on the Debye model.

VII. RESULTS AND CONCLUSIONS

This chapter will describe the analysis of the $T = 2$ resonance data en route to the extraction of relevant resonance parameters. This discourse includes program execution guidelines, general data fitting concerns, and presentation and discussion of the final parameter values and error estimates. Also, we will comment on the significance of the results and offer suggestions for improvement of these measurements and recommendations for future experiments of this type.

7.1 Fitting Guidelines -- Using RESMIN[1]

In dealing with a minimization problem with a relatively large number (>20) of parameters, one needs to be concerned with the interaction or correlations between the searched parameters since they can affect convergence to the χ^2 minimum as well as parameter error estimates. For our particular problem, some of the resonance parameters are often highly correlated and thus "compete" against one another. One obvious example is provided by the total width and resolution function width, since both parameters affect the observed width of the resonance fit. However, these parameters do influence the fit differently by virtue of their correlations with other parameters. We can get some idea of the possible correlations by looking at the relationship of the resonance parameters to the observed cross section. Adapted from equation [5.27], we have, for a

[1] We describe here the use of the resonance fitting program RESMIN in extracting the relevant resonance parameters for the $T = 2$ resonances in question. For a detailed description of the program itself along with its various functions and subroutines, we refer the reader to section 5.4 and program listings (appendix F).

particular scattering angle[2]:

$$\frac{d\sigma_{obs}(E)}{d\Omega} \propto (r^2 + c) + \left(r\cos\phi\sqrt{\Gamma_\alpha\Gamma_\beta}\right)\int_{-\infty}^{+\infty} \frac{(E'-E_R)\,R(E-E')}{(E'-E_R)^2 + \left(\frac{\Gamma}{2}\right)^2}\,dE'$$

$$+ \left(\Gamma_\alpha\Gamma_\beta - r\sin\phi\;\Gamma\sqrt{\Gamma_\alpha\Gamma_\beta}\right)\int_{-\infty}^{+\infty} \frac{R(E-E')}{(E'-E_R)^2 + \left(\frac{\Gamma}{2}\right)^2}\,dE' \qquad [7.1]$$

where r and ϕ represent sums of magnitudes and phases of interfering background amplitudes and c is a sum of terms which contributes to the background cross section.[3] Total and partial widths are given by the usual Γ's, E_R is the resonance energy, $R(E - E')$ is the total resolution function, and constant factors have been omitted for clarity.

We notice immediately that the partial widths and the background amplitudes occur as factors in the interference terms and can be "traded off" between each other. The correlation, however, is not perfect since there is a "pure" resonance term containing the product of partial widths without the presence of the background amplitudes. Thus, the extent to which the partial width and background amplitude correlations affect the search procedure may depend largely on the physics of the particular resonance reaction. That is, situations in which the pure resonant term plays a large part are apt to provide more consistent results and more reproducible values of the partial widths.

When provided with a fairly efficient program and relatively free computer time, one is tempted to perform many searches utilizing different schemes of parameter

---

[2] The values of the background amplitudes (r, ϕ, c) are functions of the scattering angle, as is the resulting value of the cross section.

[3] These background amplitudes are in general sums, but their specific form depends on the reaction and associated spin system. For example, the parameter c is a single term for our (p,α) reactions, and c is equal to zero for $^{24}Mg(\alpha,\alpha)^{24}Mg$.

interpretation and manipulation. However, after exploring many of the possibilities, it becomes evident that it is advantageous to set certain reasonable procedures for directing the searches and to follow these guidelines in all the final minimization runs. Some of the choices and procedures used are given below.

7.1.1 Parameter Choice

Although the parameter most desired is the alpha partial width, the aforementioned correlations with background parameters may make it an unwise choice for a search parameter. We chose to search on the total width, Γ, since it appears in the denominator of equation [7.1] and is less likely to be affected by off-resonance amplitudes. At the same time, we recognize that the total width is likely to be highly correlated with the resolution function width. However, the variable width parameter for the resolution function can be easily be fixed for any particular search.[4]

The branching ratios for the entrance (Γ_α/Γ) and exit (Γ_β/Γ) channels provide two other resonance search parameters. These are obvious choices since their search ranges can be constrained from published values [McG70] [Fre79]. Hence, the necessary partial width can be deduced from these first three parameters. The resonance energy, E_R, provides a fourth search parameter which seems to be very well determined by the searching procedures. The final resonance parameter, hinted at before, is the "expansion" or width factor, Δ, for the total resolution function. Note that this parameter is not itself the width of the function but rather a multiplicative factor which expands or contracts the x-axis (the energy axis) of the resolution distribution. This parameter is only necessary because of the relatively large errors in the determination of some of the pieces of the

[4] Also, unlike the several background amplitude at each angle, the width variable is represented by a single search parameter for all angles.

total resolution function (chapter VI).

The five parameters described above constitute the resonance parameters, which are the same for every angle. The remaining parameters, amplitudes from the non-resonant background (r, ϕ, c of equation [7.1]), are angle dependent and thus appear as different sets for each excitation function. That is, a typical situation would involve four scattering angles and two reactions ((p,p) and (p,α)) and thus would require eight sets (24 individual) of these background parameters. These background variables are more-or-less completely unknown and constrained only by the magnitude of the background cross sections and the fact that r and c must be greater than zero.[5]

7.1.2 General Fitting Procedures

The following standard procedures are used in obtaining the best-fit resonance parameters:

1) The final results represent simultaneous fits for several angles and, where applicable, include both elastic and inelastic data. For the final fits, all parameters are allowed to vary within their prescribed ranges as outlined above except the resolution function parameter, Δ, which in some cases is fixed. The particular value of Δ will be further discussed in the sections dealing with the results of the individual fits.

2) A χ^2 minimum is defined by MIGRAD convergence with the usual (default) MINUIT convergence criteria. Standard deviation errors are derived by MIGRAD (or MINOS when errors are suspected of being asymmetrical) and represent intervals in which our χ^2 changed by unity.

---

[5] The value of r represents the magnitude of a complex number and as such must be positive while c is, in general, a sum of squares and thus must also be greater than zero.

3) All fits include a linear energy dependence for the background cross section. The necessary slopes are calculated by a least squares fit of 10% to 30% of the wing data.[6]

4) Truncation of the wing data is utilized only if the wings behave poorly (noticeably non-linear) or are of poorer quality (large uncertainties) than the data in the resonance region. In any case, not more than 20% of the data is lost. Experience tells us that truncating the data changes the overall scale of χ^2 but has little effect on the position of the χ^2 minimum.

7.2 Caveats in Resonance Fitting

From a purely statistical point of view, fitting data with the "best" values for parameters of a particular model (point estimation) is fairly easy. One simply finds the set of parameters which minimizes the appropriately defined chi-squared function. Interval estimation is similarly straightforward by reporting the deviations in the parameter values which cause the χ^2 statistic to change by a suitable amount ($\Delta\chi^2 = 1$ for the usual one standard deviation errors or 68.3% confidence level). This procedure is made easier still by sophisticated searching and optimization computer programs such as MINUIT.

Of course, as often happens when mathematics meets reality, things are not as simple as they appear at first glance. There are several assumptions involved in the statistical approach which may lead to additional error when applied to less than perfect experimental data. Furthermore, it is hard to estimate the extent of these deviations from

[6] This same calculation yields a value for the *y*-intercept (cross section at the starting scan energy) which can be used in place of the background cross section search parameter. However, we found that it was better to have MINUIT control this overall cross section since it provides some constraints for the background parameters.

the ideal, and the effects are, in general, different for point and interval estimation. One underlying assumption of most standard statistical models is normality of the various parameter distributions and their associated errors.

Other, more specific assumptions, deal with the choice of the minimization function itself. The chi-squared statistic, as defined in equation [5.22] is sometimes denoted as a "weighted least squares," implying that the weights may not be the actual standard deviations of the measurements but perhaps only relative errors between different data points. In this case, an unknown overall scaling of the errors will affect interval estimation but not point estimation,[7] and the error estimates obtained will be proportional to the overall normalization. Although this kind of overall systematic error certainly may exist, it is impossible to predict, and we shall continue to refer to our minimization function as χ^2.

However, even if we accept the validity of the statistical framework and suitability of the χ^2 function, we must keep in mind that the estimation procedures consider the data values to be samples of the appropriate parent distribution which is represented by the theoretical or "fitting" function. That is, the model or function used to fit the data is assumed to be the correct one, and an infinite amount of data would exactly reproduce the model distribution. This is one place where the nuclear physics enters in and limits the accuracy of the point and interval estimation.

In our particular case, we know that our model of isolated resonances on energy-independent backgrounds cannot be exactly correct, if for no other reason than the requirement of an off-resonance linear energy dependence to reproduce the obvious slope of the data. As part of this "model" error, we add the uncertainties in the width and shape of the resolution function and the effects of other nearby resonances. It is not hard to see

[7] Although the value of χ^2 itself will be scaled up or down, the minimum value will "be in the same place"; that is, correspond to the same set of parameter values.

how, in some instances, this error may become very large and dominate the parameter error estimates in addition to adversely affecting the best fit parameter values.

Similarly, because of deficiencies in the model, we may have situations in which the minimum χ^2 parameter set does not represent the "best" fit. One example of this behavior is the $^{24}Mg(\alpha,\alpha)^{24}Mg$ data with the $T = 2$ resonance and nearby "pygmy" neighbor (see section 6.4.1). Whether or not this secondary bump is a real effect, it is clearly not accounted for by our simple model. Nevertheless, the fitting routine, treating all points equally, will try to generate a fit which is close to all the points. The result, in this case, is a broadening of the resonance to try to "split" the points of the secondary structure. Since we cannot expect the fitting routines to make decisions about the appropriateness of the model, we must sometimes interject our own biases in the form of fixed or tightly constrained parameters. In this manner we are supporting the "spirit" rather than the "letter" of the minimization procedure. Thus, we may often refer to "best fits" even when they do not correspond exactly to those with the absolute minimum value of chi-squared. Figure [7.1] demonstrates this scenario where the minimum χ^2 fit seems to misrepresents the data near the center of the resonance.

Whatever the sources of error, it is clear that the purely statistical error determinations are probably underestimates of the total error, and efforts to represent the additional uncertainties add an air of subjectivity to the statistically objective processes of point and interval estimation. In our quoted values for the resonance parameters, we try to give reasonable estimates for several of the aforementioned errors. Some of these estimates are based on the deviations of the values for varied conditions (different resolution functions, truncated wings of data, different background slopes, etc.) in the many searches performed and are thus probably more realistic, albeit somewhat arbitrary. Strictly speaking, these errors produced by sources such as model shortcomings are not directly interpretable as standard deviations of parameter values. Thus, no attempt was

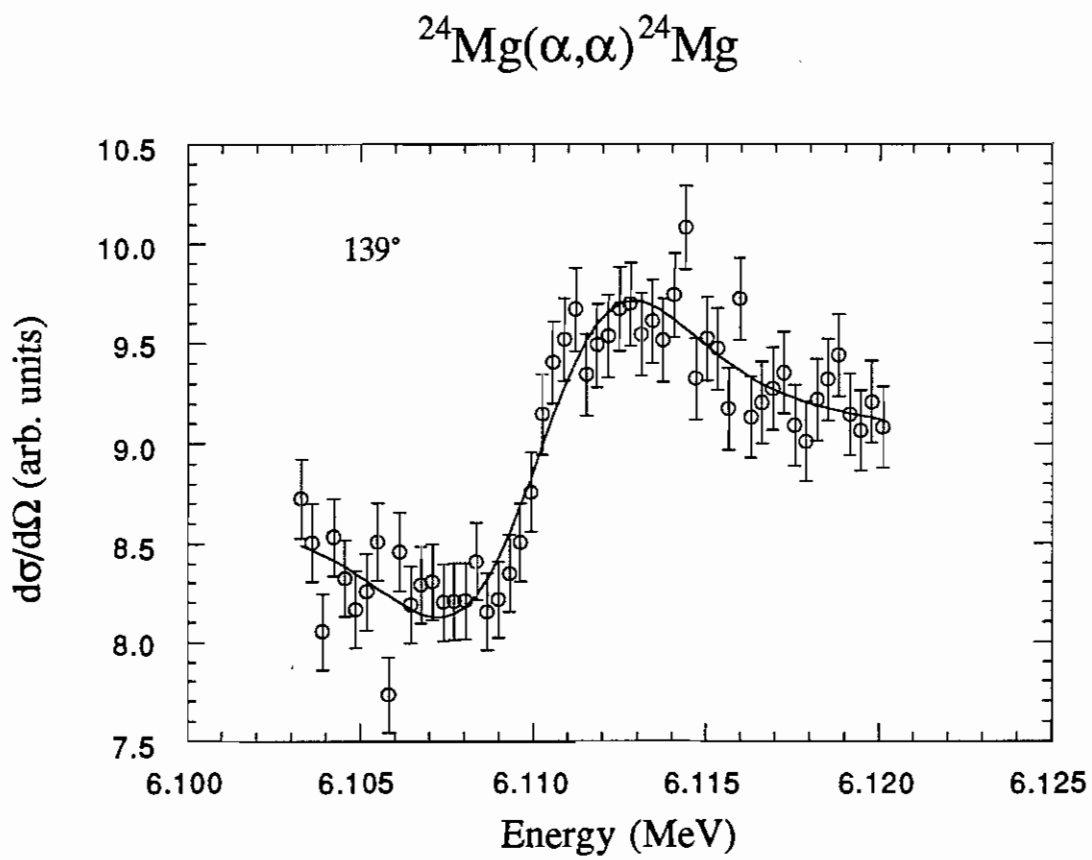


Figure [7.1]: Example of data and lowest χ^2 fit which seems to misrepresent the data in the resonance region.

made to combine the errors (e.g. adding variances) from these different sources, and the errors are quoted separately.[8]

7.3 Results from RESMIN

The final results for the resonance parameter values are given in table [7.1] for each of the three different $T = 2$ states examined. The quoted resonance energies in column [2] are projectile energies in the laboratory frame with error bars being those from the experimental energy calibration [Wil83a]. We note that standard deviation errors from the search routine (MIGRAD) were much smaller, generally only tens of eV, and, as such, are completely negligible. Furthermore, we recognize that all of the values for the incident particle resonance energies agree with published values to within allowed error ranges.

Column [3] of table [7.1] gives the value of the parameter, Δ, for each of the final fits. We recall that Δ is a factor which multiplies the x-axis (energy axis) of the total resolution function and should not be confused with the actual width of the resolution function distribution. The resolution functions, before multiplication by Δ, are described in detail in chapter VI and given in figures [6.4], [6.8], and [6.16]. Thus, the actual width of the resolution function for the final fits can be obtained by multiplying the widths of the initial functions (chapter VI) by the Δ parameter.

The chi-squared per degree of freedom (table [7.1], column [4]) is given for each of the fits from which the final resonance parameters are taken. This qualitative goodness-of-fit indicator suggests fairly good fits for all the data. Values of the resonance width

---

[8] We can think of the standard deviation or statistical error as representing the precision of the measurement (more data would lower these errors), whereas other errors might affect the overall accuracy and could only be improved by eliminating or reducing the sources of the errors (more careful measurements or using a better model).

[1] Reaction(s)	[2] Resonance Energy (MeV)	[3] Δ	[4] $\frac{\chi^2}{\nu}$	[5] Width Parameter	[6] Best Fit Value	[7] Allowed Range	[8] MIGRAD [or MINOS] Error	[9] Normalization Error (eV) [5%]	[10] Error in Slopes(eV) [5%]	[11] Other Error (eV)
$^{23}Na(p,p)^{23}Na$ $^{23}Na(p,\alpha)^{20}Ne$ [1^{st} $T=2$ state in ^{24}Mg]	3.905 ± 0.002	1.1	2.1	Γ	345 eV	> 0 eV	± 16 eV	± 17	± 17	± 125
				Γ_p/Γ	0.60	0.43 - 0.78 [a]	± 0.02			
				Γ_α/Γ	0.056	0.010 - 0.073 [a]	± 0.004			
				Γ_α	19 eV		(± 1) eV	(± 1)	(± 1)	± 4
$^{24}Mg(\alpha,\alpha)^{24}Mg$ [1^{st} $T=2$ state in ^{28}Si]	6.112 ± 0.002	1.2	1.1	Γ_α	105 eV	> 0 eV	$^{+5}_{-3}$ eV	± 5	± 5	$^{+6}_{-4}$
				Γ_α/Γ	0.80	0.63 - 0.80 [a]	- 0.08			
				Γ	132 eV		$^{+9}_{-4}$ eV	± 7	± 7	$^{+21}_{-9}$
$^{31}P(p,p)^{31}P$ $^{31}P(p,\alpha)^{28}Si$ [1^{st} $T=2$ state in ^{32}S]	3.288 ± 0.002	0.7	1.5	Γ	40 eV	> 0 eV	± 5 eV	± 2	± 2	± 30
				Γ_p/Γ	0.9	0.69 - 1.0 [a]	± 0.2			
				Γ_α/Γ	0.16	0.010 - 0.25 [a]	± 0.02			
				Γ_α	6.5 eV		(± 0.8) eV	(± 0.3)	(± 0.3)	± 2.5

[a] Combined ranges from [Fre79] and [McG70]

Table [7.1]: Results from our $T = 2$ state resonance fits. Resonance energies are those of the projectile in the laboratory frame. Widths are in the center-of-mass frame. Comparisons of energies and widths with other published values can be found in table [7.2]. For each reaction, errors in parentheses are derived from values above the dashed line. Columns [10] and [11] represent the so-called "model erors." See section 7.3 for a complete description of each column.

parameters (column [6]) represent those values extracted from the best fit to the data and are given with respect to the center-of-mass reference frame. The allowed ranges in column [7] are constraints imposed on the searched parameters. For the branching ratios, these constraints are a combination of the published values of the branching ratios and their errors [McG70] [Fre79].[9]

The statistical error (ideally, the standard deviation) given in column [8] of table [7.1] is a representative single-parameter error estimate from MINUIT based on $\Delta\chi^2 = 1$. These values are estimated by MIGRAD from the covariance matrix, or, where asymmetric errors are given, are derived from actual calculations of $\Delta\chi^2$ by MINOS. An error of about 5% is included for possible errors in the overall cross section normalization (column [9]). The most likely source of such an overall error is an incorrect determination of the target thickness. Since the cross section is proportional to the squares of the widths, a 5% error in the width would result from a 10% error in the cross section or target thickness.

Two other error estimates are given in table [7.1], columns [10] and [11]. These values are the so-called "model errors," meant to reflect the shortcomings of the fitting function which is used to model the data (see section 7.2). The first, 5%, error represents deviations in the widths produced for various calculations of the background slope. Note that this error involves different calculations of the slope for different ranges of background points but does not represent errors resulting from the inclusion of this energy dependence (non-zero versus zero slope). The second model error (column [11]) is a subjective value based on many factors directly or indirectly resulting from model deficiencies. Included in this estimate is the troublesome error from the uncertainty in the

[9] Note that these ranges are the union (not the intersection or overlap) of the ranges given by [McG70] and [Fre79]. This was necessary since some of the values from the two works do not agree to within the quoted errors.

resolution function width. Since this column is very much reaction specific, detailed discussions are left for the following sections involving the results of the individual experiments.

For each experiment the entries above the dashed horizontal line represent variables which were actually parameters in the MINUIT search. Parameters and values below the dashed line are derived from the ones above it. For the error estimates of the derived quantities, parentheses indicate errors which are simple arithmetic determinations based on searched parameter errors and values (e.g., errors multiplied by branching ratios). On the other hand, values which are not in parentheses indicate situations in which some other information is available as an aid to the estimation. Where applicable, these errors will be described along with the particular experimental results.

Although utilizing identical programs, the parameter searches behaved somewhat differently for different experiments. For instance, a 10% change in the resolution function width produced a large variation in the value of the total width for the $^{23}Na(p,p)^{23}Na$ / $^{23}Na(p,\alpha)^{20}Ne$ experiment while causing almost no change for the $^{31}P(p,p)^{31}P$ / $^{31}P(p,\alpha)^{28}Si$ data. Because of differences in the program behavior, the particulars of the searches and some of the error estimates for each experiment will be considered separately and discussed in detail in the following sections. In particular, we will try to point out those cases in which the different behavior may be caused by the underlying nuclear physics of the reaction.

7.3.1 Results for the Lowest T = 2 State in ^{24}Mg

Fits and data for the $^{23}Na(p,p)^{23}Na$, $^{23}Na(p,\alpha)^{20}Ne$ experiment are shown in figure [7.2]. The fits represent those parameter values given in table [7.1]. The slope of the background cross section was calculated using 35 points (~ 3 keV) on each end of the

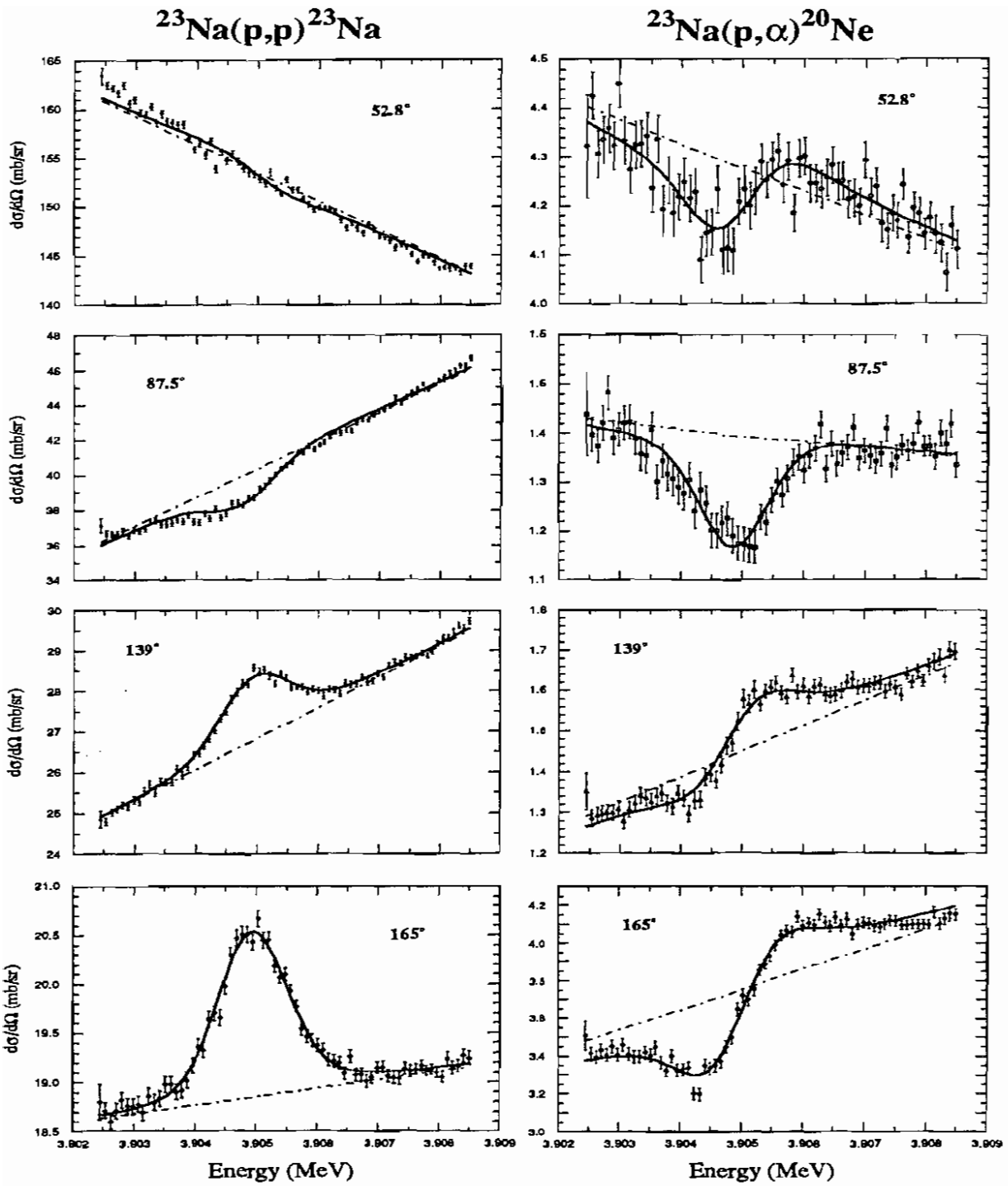


Figure [7.2]: Data and best fits for the lowest $T = 2$ state in ^{24}Mg . The dot-dashed line is a least squares fit to a linear background. For details and a list of the resonance parameter values, see the text and table [7.1].

full data set. The wings of the data set were then truncated, 27 points of the left and 25 on the right, since these points resulted from separate ramp scans and had much poorer statistics than the data on-resonance. All four angles for both (p,p) and (p,α), a total of eight data sets, were fit simultaneously to extract the resonance parameters.

The variation in the total width with changes in the resolution function parameter was a particular problem for these data. Only the approximate sum of the resonance width and the resolution function width seemed to affect the chi-squared value. That no particular combination of the two parameters is preferred is evidenced by figure [7.3]. Although the slightly preferred (lower) χ^2 minimum occurs for $\Delta = 1.1$, we can see from the figure that values of Δ between 1.0 and 1.2 serve almost as well if the value of the resonance total width is allowed to change. More discouragingly, a very wide range of values for the total width is possible for this 20% change in Δ. The best fit value of the total width in table [7.1] is that value for which $\Delta = 1.1$ and the rather large error in column [10] represents the range of Γ between the χ^2 minima for $\Delta = 1.0$ and 1.2.

Having peered into the abyss of ill-defined parameters, we note that the exit branching ratio becomes increasingly smaller as the value of the total width becomes larger. Furthermore, this trend is reproduced in many searches utilizing different values of the resolution function parameter. The approximate behavior of the branching ratio is shown in figure [7.4a] where the error band arises from different values of the resolution function width and errors involved in the starting points of the searches, the latter almost certainly because of correlations with the background amplitudes. This inverse proportionality of the best-fit values of the branching ratio and total width indicates an unsuspected sensitivity of the fits to the partial width.[10] By plotting the partial width versus the total width (figure [7.4b]), we find that, indeed, Γ_α is nearly constant

[10] We had expected the background parameter contributions to destroy the sensitivity of the fits to the partial widths (section 7.2).

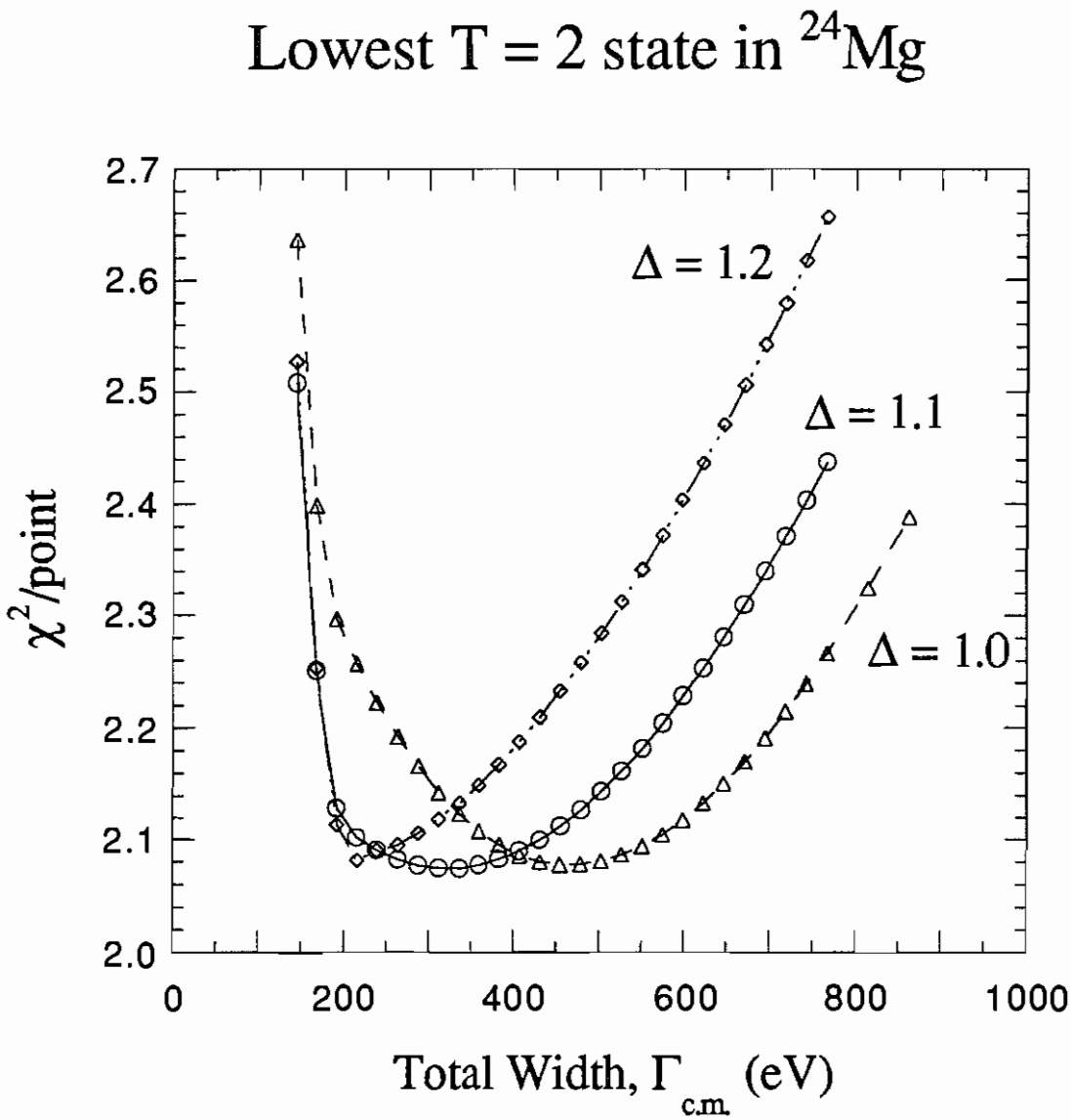


Figure [7.3]: Variation in χ^2 over values of the resonance total width for the lowest T = 2 state in ^{24}Mg. Δ is the resolution function multiplicative parameter.

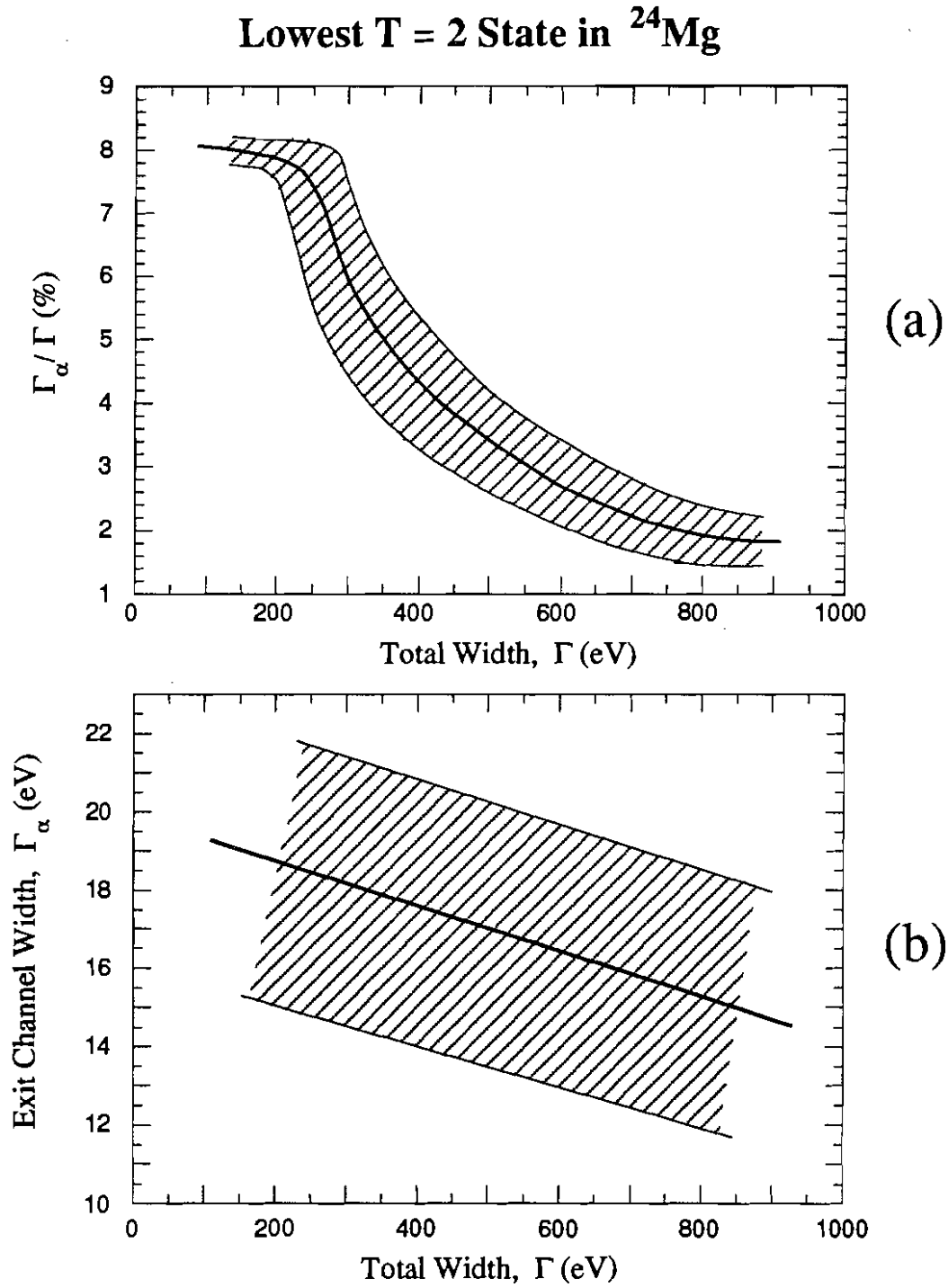


Figure [7.4]: Branching ratios (a) and partial widths (b) for a range of total width values. The error band (hatched area) represents possible parameter values for minimum chi-squared fits. Curves are guides, not fits.

throughout the entire range of Γ values, showing the aforementioned variations superimposed on a gently sloping curve.

The sensitivity of the search to the value of the partial width allows us to estimate more useful "other" errors, shown in column [11] of table [7.1], based on the variations mentioned above and a ± 10% range in Δ, the resolution function width parameter. Also, we can speculate on the underlying causes of this behavior of the minimization searches. As mentioned earlier (section 7.1), a strong contribution of the pure resonance term (not involving interference of the background amplitudes) at some angles would result in the partial widths becoming more important than the background parameters. It might be reasonable to conclude that resonance scans such as the $^{23}Na(p,\alpha)^{20}$ Ne ones, which show a relatively large excursion and a small overall cross section, might have these significant contributions from the pure resonant terms and thus be good indicators of the resonance partial width.

As expected, the resolution function derived for this experiment proved to be fairly accurate, with the minimum χ^2 occurring for Δ just less than 1.1, corresponding to <10% error in the width. Ironically, the closeness of the best fit and calculated distributions may be caused, in part, by the poorer-than-normal incident beam resolution. Because of its large width (> 1 keV FWHM), the incident beam energy distribution is by far the dominant contribution to the total resolution function. However, the scheme for measuring this incident beam function worked extremely well and provided fairly accurate results. Contrastingly, the calculations of the target straggling and certainly the Doppler broadening are subject to fairly large errors and may not be as well determined, but their contributions to the total function are relatively negligible. This leaves us with a total resolution function which is approximately equal to a relatively well determined incident beam energy distribution.

In addition to the primary data analyzed to extract the resonance parameters, data

with larger statistical uncertainties from a second set of angle settings were analyzed as a check. The fits incorporated the same basic resolution function (figure [6.8]) as did the fits for the primary data. For these data, minimum χ^2 fits with Δ fixed to 1.1 yield a value for Γ_α of about 20 eV. The data and fits are represented in figure [7.5].

7.3.2 Results for the Lowest T = 2 State in ^{28}Si

Fits for the $^{24}Mg(\alpha,\alpha)^{24}Mg$ data, utilizing the resonance parameters in table [7.1], are shown in figure [7.6]. The entire range of available data was employed in the fitting process, and the background slopes were calculated using 20 points (3.6 keV) at each end of the resonance scan. As usual, the branching ratio was permitted to vary, but the χ^2 minimum was achieved when Γ_α/Γ was at the upper limit of its allowed range. When the constraint was relaxed, the branching ratio after convergence was only slightly higher ($\Gamma_\alpha/\Gamma = 0.81$ instead of 0.80).

The simplified spin system requires that at each angle the background parameter c in equation [7.1] must be zero (section 5.1). Because of the strong constraint on the remaining, single complex background amplitude (its square must equal the off-resonant cross section), the search is very sensitive to the partial width, since it is largely responsible for the height or excursion of the resonance. However, the search works equally well (finds the same χ^2 minimum) when searching on the total width since Γ and Γ_α are directly related through the branching ratio. In practice, some searches were performed using the partial width and others using the total width, with almost identical results. The best-fit parameters are equal in both cases to well within the estimated standard deviations (< 1%). The error intervals are, however, somewhat different for the total and partial widths, since the latter is not as sensitive to changes in the resolution

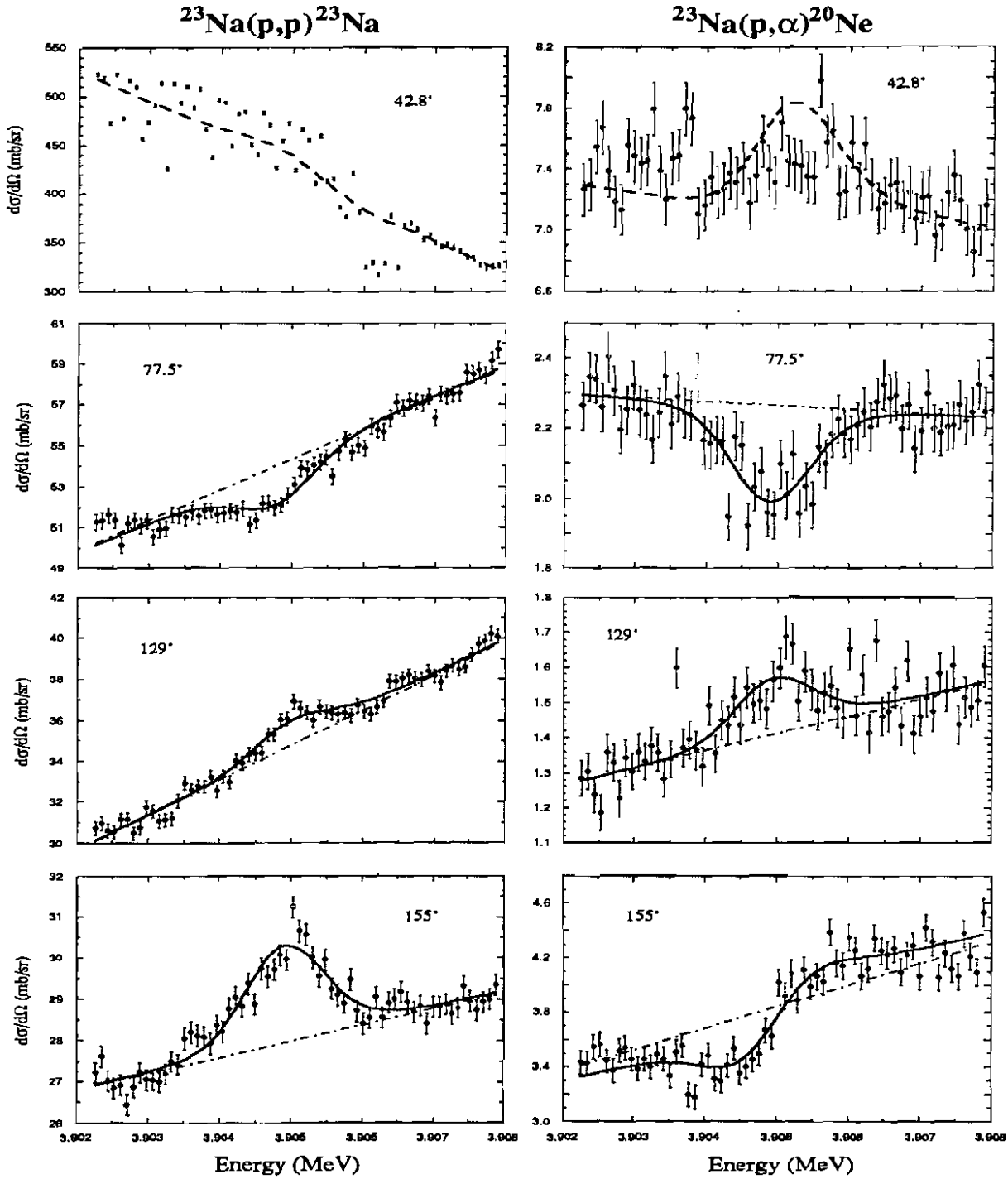


Figure [7.5]: Fits and data from angle set #2 for the lowest $T = 2$ state in ^{24}Mg . These angles are used as a check and are not included in the determination of the resonance parameters as given in table [7.1]. However, the best fit values are within about 5% of those determined from angle set #1. The dot-dashed line is a least squares fit to a linear background, and the dashed line for the 42.8° data indicates that it was not included in the fitting searches.

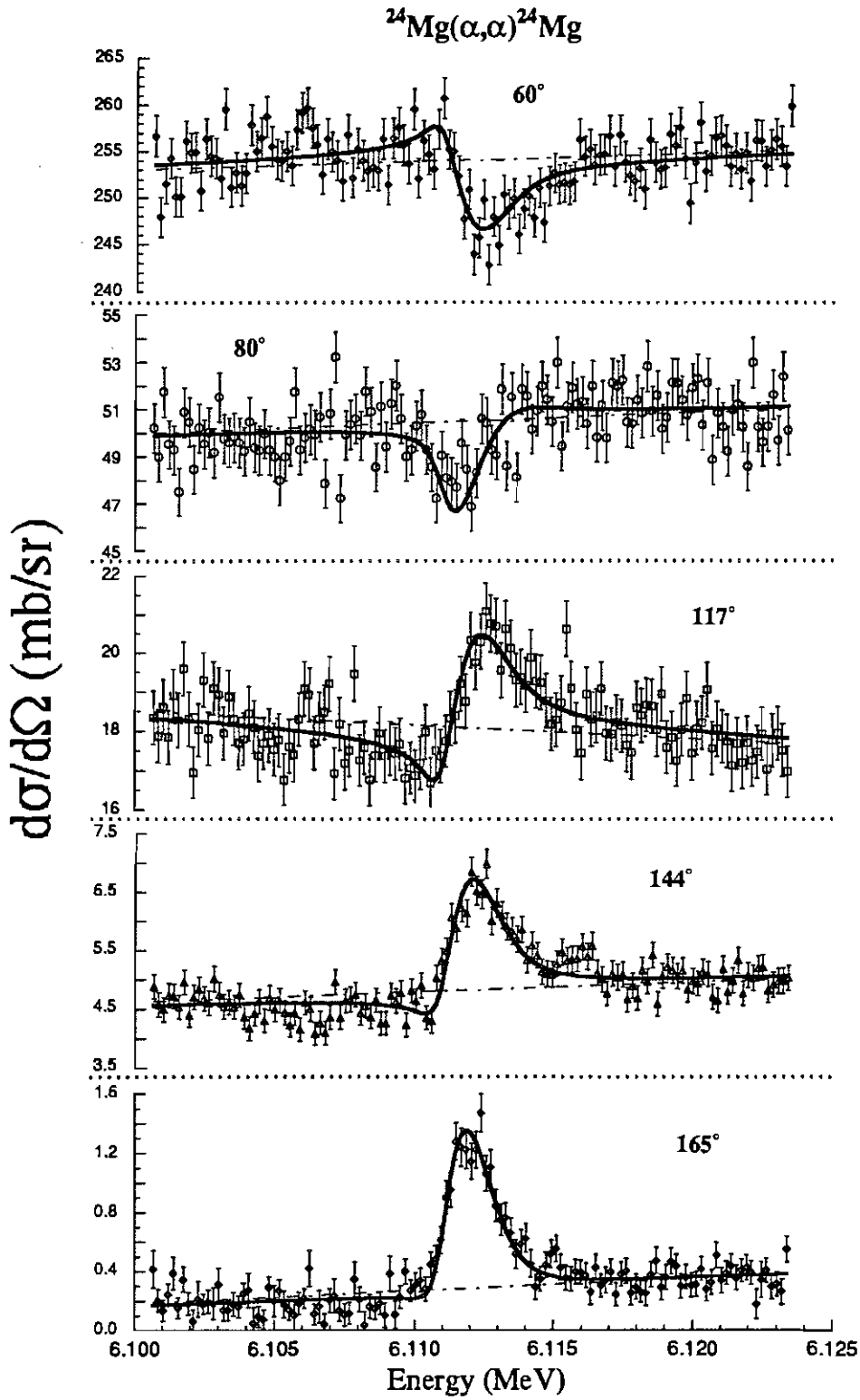


Figure [7.6]: Data and fits for the lowest $T = 2$ state in ^{28}Si. The dot-dashed line is a least squares fit to a linear background. For details and resonance parameter values, see the text and table [7.1].

function width.[11] The values and errors quoted in table [7.1] for the total width are, in fact, from searches using the total width as a search parameter and in every way on an equal footing with values quoted for the partial width.

Acceptable fits (by eye) were produced for values of Δ between 1.0 and 1.4. The values of table [7.1] are those for a fixed $\Delta = 1.2$, and the "other" model error (column [11] of table [7.1]) represents the spread of values between the χ^2 minimum fits for $\Delta =$ 1.0 and $\Delta = 1.4$. Figure [7.7] shows the behavior of χ^2 as a function of Γ_α near the minimum for $\Delta = 1.0$, 1.2, and 1.4. Note that the absolute minimum χ^2 fits are those with $\Delta \sim 1.4$, but, as we can see from the figure, the differences in the minimum values are slight. The fits with $\Delta = 1.4$ were judged not to represent the data as well as those with Δ = 1.2, mainly because of the secondary structure appearing just above (in energy) the resonance at several angles. As noted previously (section 6.4.1), this secondary structure serves to widen the resonance fit slightly to minimize the distance (actually the squares of the distance) to all the data points. The resolution function width is the parameter most available to produce this broadening since the total width is tied to the well-defined partial width through the branching ratio.

An error in the total resolution function width of + 20% is perfectly reasonable and could be produced by a number of causes. Since Mg target uniformity is almost always a problem, it is most likely that the straggling distribution is wider than estimated by our calculation based on a uniform layer of target material.[12] Also, our measurement of the incident beam resolution function was only marginally successful with large uncertainties in the width. Furthermore, as mentioned in chapter VI, the presence of any of the target material as MgO would result in a Doppler broadening distribution which

---

[11] A 50% change in Δ (from 1.0 to 1.5) produced only a 13% variation in Γ_α.

[12] Since straggling is proportional to the square root of the thickness, a 20% error in the straggling could be caused by an approximately 40% target non-uniformity.

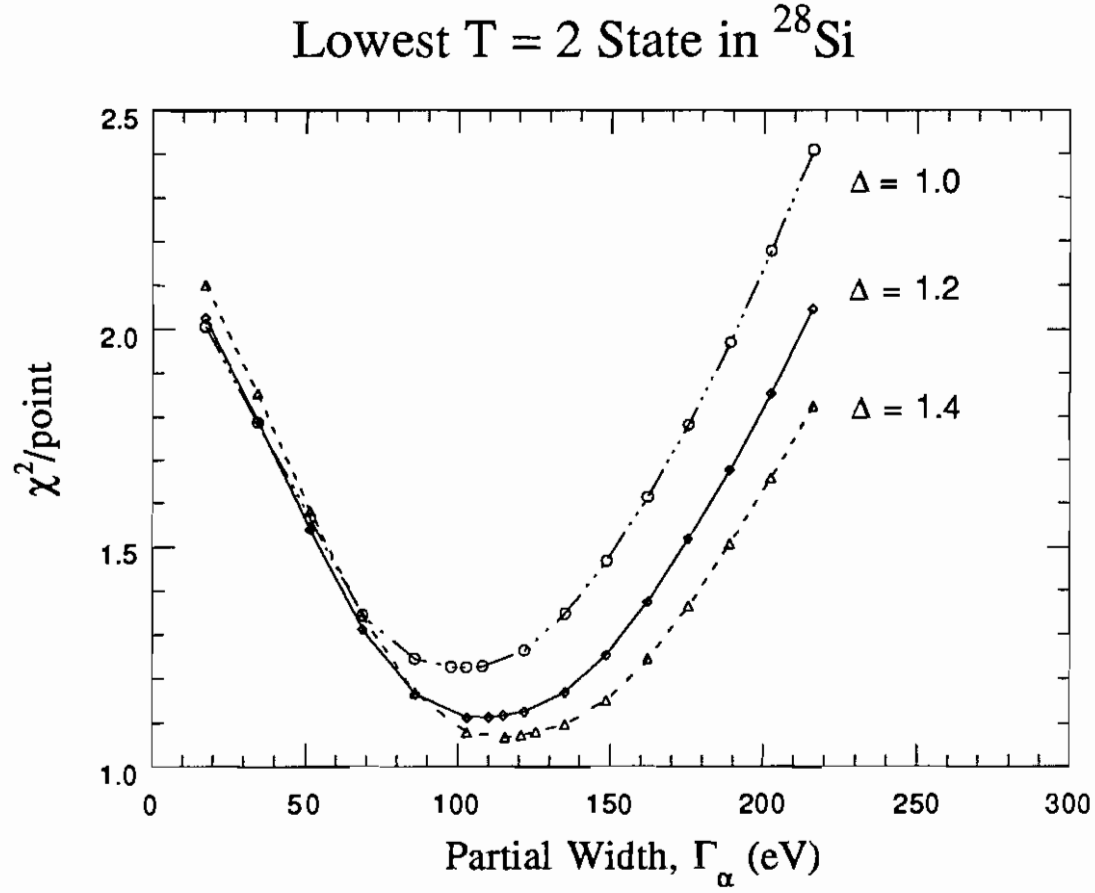


Figure [7.7]: Chi-squared as a function of the partial width for the ^{24}Mg(α,α)^{24}Mg data. The parameter Δ is the resolution function width multiplicative factor.

might be much different (and probably larger) than calculated by our simple model. Finally, there may always be unaccounted for experimental effects which can broaden ΔE so an error in the upward direction is understandable. With all this in mind, a resolution function width of 20% more than calculated is very acceptable and appropriate.[13]

Data from one of our previous experiments scanning this resonance with much poorer resolution were fitted using the same procedures as above but with an arbitrary width (> 4 keV) Gaussian resolution function. The best fits for these data occur with a total width of 144 ± 9 eV (MIGRAD error) and a branching ratio (again at the upper limit) of 0.80.

7.3.3 Results for the Lowest T = 2 State in ^{32}S

Data and fits for the $^{31}P(p,p)^{31}P$ and $^{31}P(p,\alpha)^{28}Si$ reactions are given in figure [7.8]. The full experimental range of data points was truncated by five points (785 eV) at each end, mainly because of the exponential behavior of the low energy wings of the elastic scattering data at 148° and 162°. Note also that some additional points for these angles were not factors in the fitting and are left out of the figures.[14] The data at 48° are left out because of experimental problems (section 6.2), and the 82° data are not included

---

[13] As a final note on the resolution function errors, two different incident beam resolution functions were obtained by fitting the measured values as described in section 6.4.2.2. These functions differ in the coefficients of the log term, γ, in equation [6.3] and the distributions differ mainly in the values of the third and fourth moments. The resolution function used for the best fits did not allow a log term ($\gamma = 0$) while the other function had a γ-coefficient of about 3. When used to fit the resonance data, the slightly different total resolution functions (again differing mainly in the values of their higher moments) made a less than 2% difference in the value of the partial width.

[14] In addition to the truncation, an additional nine points on the low energy ends of the (p,p) data at 148° and 162° were effectively removed from the fitting process. Although the points could not simply be deleted from only two data sets (without upsetting many of the program functions), they were assigned very large error bars to "weight" them out of consideration.

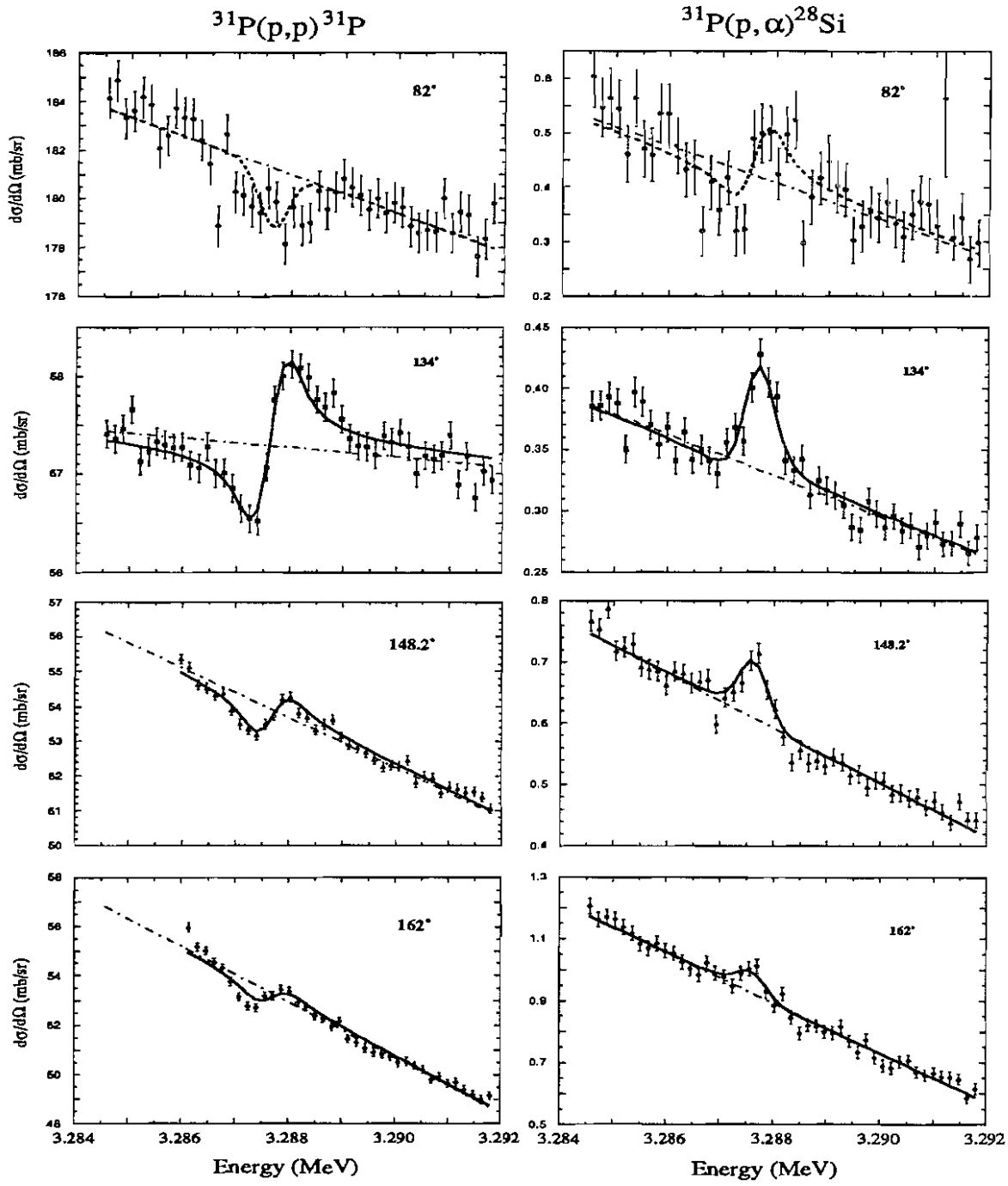


Figure [7.8]: Data and fits for the lowest $T = 2$ state in ^{32}S. The dot-dashed line is a least squares fit to a linear background and the dashed curve at 82° indicates that this data is not considered in determining the resonance parameters as given in table [7.1].

in the determination of the resonance parameters as signified by a dashed line for the best fit. Background slopes are calculated based on five points (785 eV) on each end of the data set.

The best fit data occurred with a resolution function parameter, Δ, of about 0.7 with minimum χ^2 values significantly higher for values of $\Delta \geq 0.8$. As we can see from figure [7.9] the minimum value of χ^2 occurs for a rather small value of the total width and produces a rather shallow minimum. Unfortunately, we are not as lucky as we were with the $^{23}Na(p,\alpha)^{20}Ne$ data, and the partial width is not constant over a wide range of total width values. In fact, the Γ_α behaves similarly to Γ, the result of a near constant branching ratio over much of the explored range of the total width.

The situation is worsened by the fact that the different searches seem to converge to nearly the same value of the total width, regardless of the value of the branching ratios. Thus, we have a situation in which the search is only slightly sensitive to the partial widths, implying that perhaps the background amplitudes are "picking up the slack" in determining the resonance height. This in turn implies that the interference terms (those containing background amplitudes and partial widths on essentially equal footing) are probably the most important terms in the cross section formula.

Still worse is the fact that the χ^2 minima are not always reproducible; that is, the parameter set for MIGRAD convergence seems to depend somewhat on the starting values of the involved parameters. However, virtually all the minima for χ^2 (for $0.6 \leq \Delta \leq 0.8$) occur in the range of the total width 10 eV $< \Gamma <$ 70 eV and for the partial width 4 eV $< \Gamma_\alpha <$ 9 eV. Note that the range of Γ_α is relatively smaller than that of Γ, since the searches do retain some sensitivity to the partial width. It is on this that we base our errors in column [11] of table [7.1] and acknowledge that the best fit values given are representative values and not from any one particular minimization search.

Given the manner in which we approximated the components of the total

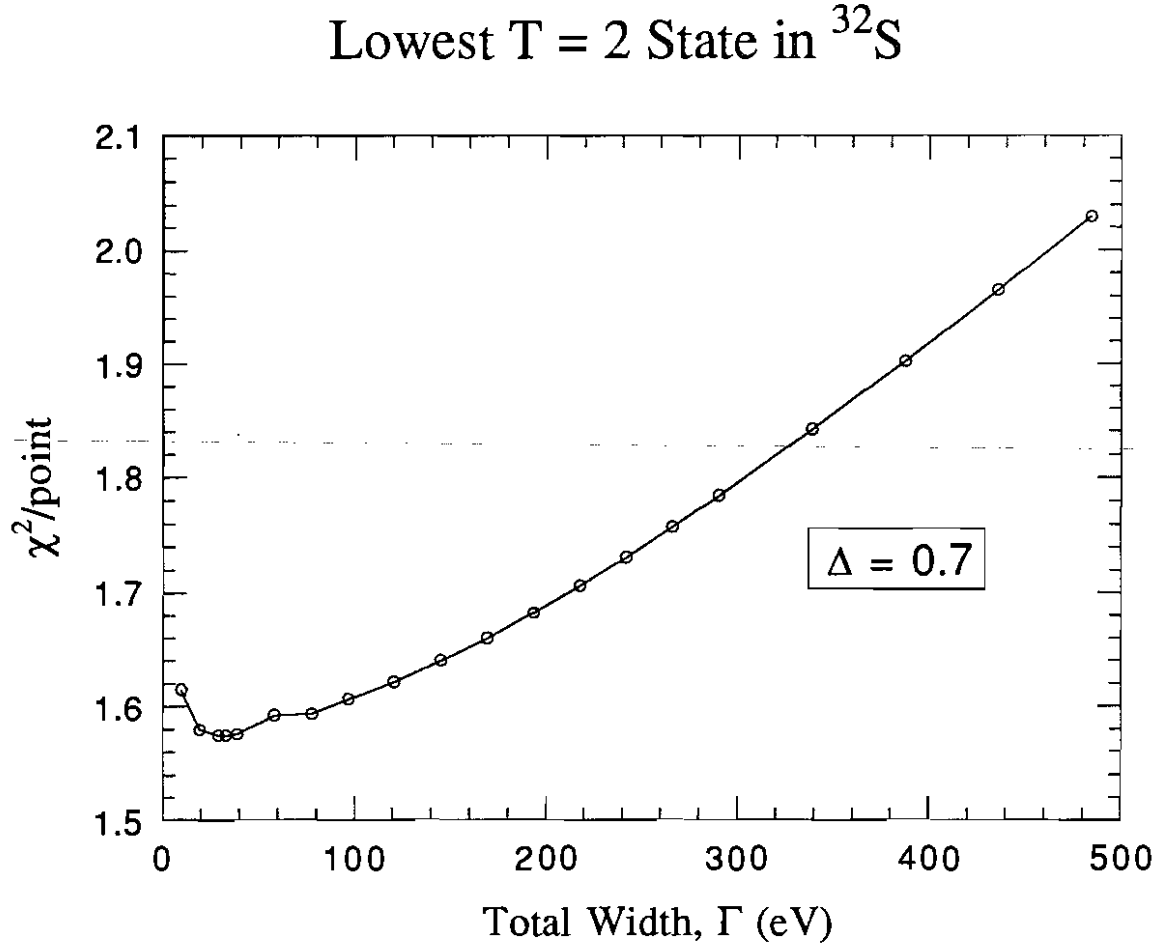


Figure [7.9]: Behavior of χ^2 as a function of the total width of the lowest T = 2 state in ^{32}S. The resolution function multiplicative parameter, Δ, is fixed but all other parameters are allowed to vary to determine the points on the curve.

resolution function, we cannot be dissatisfied with a 30% error in the width in either direction. Since we had no direct measurement of the incident beam resolution function[15] and since our method of estimating the width was akin to "grabbing at straws," any error value would not be too surprising. Given the fact that this component is probably the dominant one in the total resolution function, we can attribute to it most of the error. It is somewhat comforting to note that a 30% decrease in the width puts us closer to the expected and previously measured resolution [Wil82] of typical unpolarized proton beams at TUNL.

In addition to the data of figure [7.8], we have analyzed the data taken at other angles, identified in chapter VI as angle set #2. Using the measured resolution function (figure [6.4]), we find that these data are fit fairly well using the resonance parameters of table [7.1]. The minimum χ^2 fits with Δ fixed at 0.7 give $\Gamma = 30 \pm 8$ (MIGRAD errors) and $\Gamma_\alpha = 5.9$ (from multiplying Γ by the branching ratio). These fits and data are shown in figure [7.10].

7.4 After the Searching

After the resonance parameters are determined, the reduced alpha widths, γ_α^2, are calculated by dividing out angular momentum and coulomb barrier penetration probabilities (see section 1.4). These probabilities are given by the Coulomb penetrabilities which are computed using the TUNL program PENET utilizing a radius parameter of $1.4\,A^{1/3}$. Here A is the target mass (or residual nucleus mass in the case of inelastic scattering). Note that there is a fairly large error involved in the accuracy of the penetrability calculation, in large part due to the choice of the radius parameter.

---

[15] We recall that this experiment was performed before the incident beam measurement scheme was fully operational.

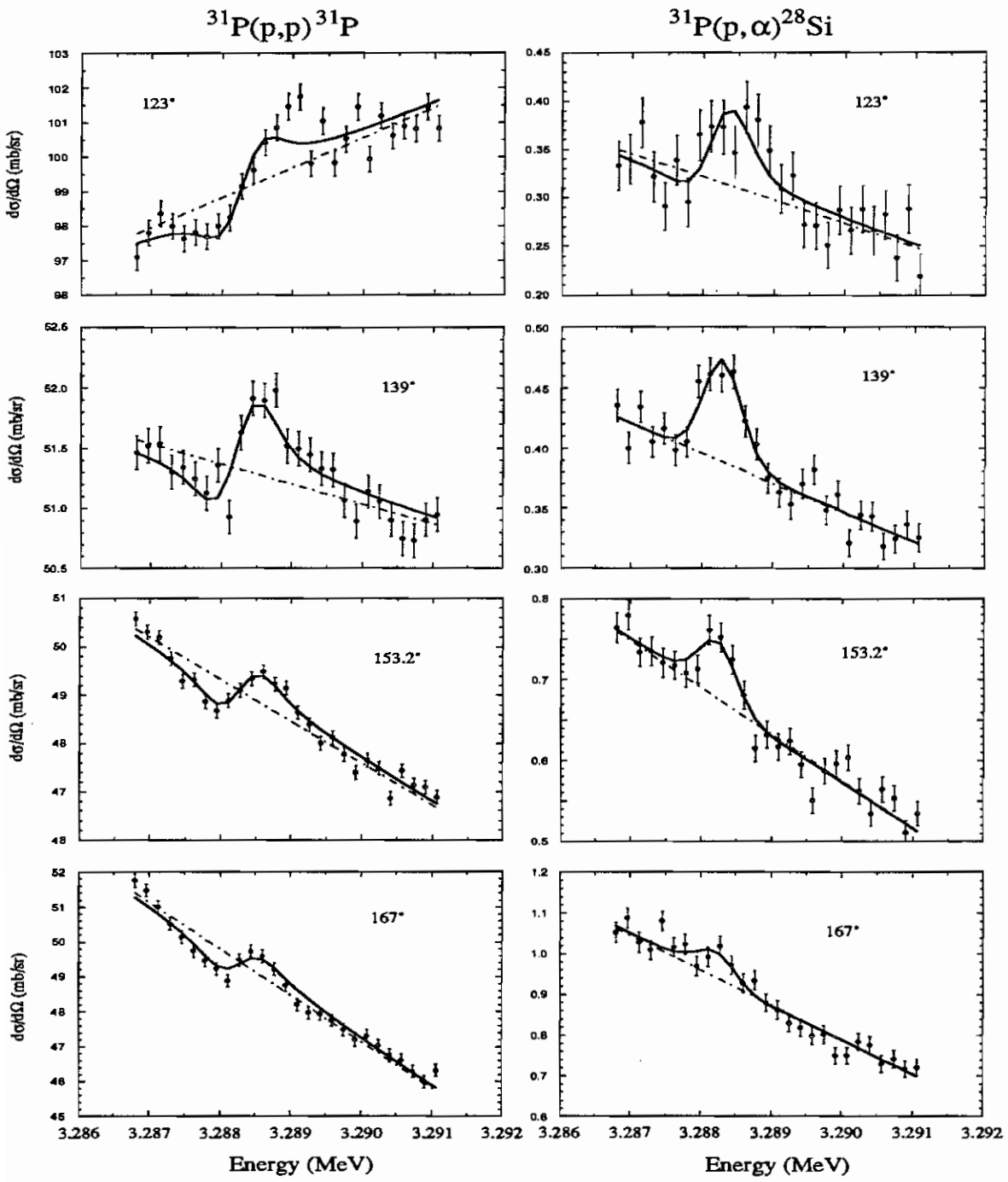


Figure [7.10]: Fits and data from the second angle set for the lowest $T = 2$ state in ^{32}S . These data were **not** involved in the determination of the resonance parameters, although the minimum chi-squared fits shown are almost identical to those produced from the parameter values in table [7.1]. The dot-dashed line is a least squares fit to a linear background.

Dimensionless single particle fractions,

$$\theta_i^2 = \frac{\gamma_i^2}{\gamma_{sp}^2}$$

are calculated by dividing the reduced widths by the single particle estimates, $\hbar^2/\mu R^2$, where μ is the system reduced mass, and we again use $R = 1.4\,A^{1/3}$ as the radius parameter. Again, as with the reduced width estimates, because of a large latitude allowed in the value of R and various definitions of the single particle widths, one should be cautious of comparisons with other published values of the single particle fraction. Finally ,we use the square root of this single particle fraction as our estimate of the isospin-mixing amplitude.[16]

7.5 Conclusions

Condensed results for the primary resonance search parameters for the $T = 2$ states, along with previously published values, are presented in table [7.2]. We note that the present determinations agree with other values except in the cases of the resonance energies in ^{24}Mg and the partial width of the $T = 2$ state in ^{28}Si based on ^{24}Mg(α,α)^{24}Mg.

Results of the calculations of the reduced widths and isospin-mixing amplitudes as outlined in section 7.4 are given in table [7.3]. Also, tabulated in table [7.3] are values of the isotensor-mixing amplitudes calculated from a schematic model based on previously available data and presented by [Wil83].[17] Note that the errors on the model predictions are only those from the compiled data and from fitting a $Z^{4/3}$ dependence to the

---

[16] Our definitions of radius parameter, single particle estimate, and mixing amplitude are compatible with those of [Wil83].

[17] The T-mixing amplitude calculations used the same assumptions about the values of the radius parameter and the form of the single particle reduced widths.

Lowest $T = 2$ state in	Present Work	Previous Work
E_x (MeV)	15.432 ± 0.002	15.4364 ± 0.0006 [Heg77]
^{24}Mg		
Γ_α (eV)	19 [± 4.4]	30 ± 14 [Fre79]
E_x (MeV)	15.224 ± 0.002	15.225 ± 0.003 [End78]
^{28}Si		
Γ_α (eV)	105 [± 10]	222 ± 22 [Iko79] (301 ± 190 [Fre79])
E_x (MeV)	12.050 ± 0.002	12.050 ± 0.004 [End78]
^{32}S		
Γ_α (eV)	6.5 [± 2.7]	< 37 [Ver73] [a]

[a] using $\Gamma_\alpha/\Gamma = 0.16$

Table [7.2]: Comparison of present and previous values for $T = 2$ state resonance parameters. E_x is the energy of the $T = 2$ state with respect to the ground state, and values in brackets are errors from table [7.1] (columns [8] - [11]) added in quadrature.

Reaction	γ_α^2 (eV)	$\lvert\theta_\alpha\rvert$ ($\times 10^{-3}$)	$\lvert\theta_\alpha\rvert$ ($\times 10^{-3}$) [Wil83]
^{23}Na(p,α)^{20}Ne	10 [± 2]	3.5 [± 0.4]	6.9 ± 0.9
^{24}Mg(α,α)^{24}Mg	218 [± 20]	17.1 [± 0.8]	8.8 ± 1.1
^{31}P(p,α)^{28}Si	36 [± 15]	7.4 [± 1.5]	10.8 ± 1.4

Table [7.3]: Calculated reduced widths and T-mixing amplitudes compared with the results of a simple schematic model. Brackets emphasize that our errors are not standard deviations, but rather errors derived from those of table [7.1] (columns [8] - [11]) added in quadrature.

amplitudes and do not include those from model assumptions. Furthermore, the errors on our experimental values are simply propagated from the errors given in table [7.2]. Thus, the "agreement" between the determined and predicted amplitudes is really fairly good. The values of amplitudes in table [7.3] are compared graphically in figure [7.11].

Although experimentally challenging, the investigations of the lowest $T = 2$ states in ^{24}Mg, ^{28}Si, and ^{32}S proved to be successful in many respects. Our experimental determinations of the single particle fractions are included, along with other available data in this mass region, in figure [7.12]. Our final values for the resonance parameters are, I believe, an improvement over previously published, experimentally determined values.

Another important aspect of this work is the refinement of the techniques used to obtain and analyze the high resolution resonance data. The data acquisition programs allow us to analyze selected parts of spectra for each setting of the target ramp voltage. Also on hand are fairly general programs to fit excitation function data with an isolated Breit-Wigner resonance. In their present form, the programs can be used to analyze resonance reactions which proceed through a $J^{\pi} = 0^{+}$ compound state.

Especially gratifying is the determination of the beam resolution function for the ^{23}Na(p,p)^{23}Na / ^{23}Na(p,α)^{20}Ne data. This determination proved that when proper procedures are followed, our direct measurement system provides good estimates of the shape and width of the beam resolution function. Further refinements of this measurement , along with projected improvements in the TUNL high resolution system hardware, may provide the possibility of even more precise determinations of isospin forbidden decay widths from resonance reactions.

The refinement of the resolution function determination is probably the most obvious and consequential improvement for high resolution work. However, one should not overlook the other accomplishments involved in this work: the specialized data taking

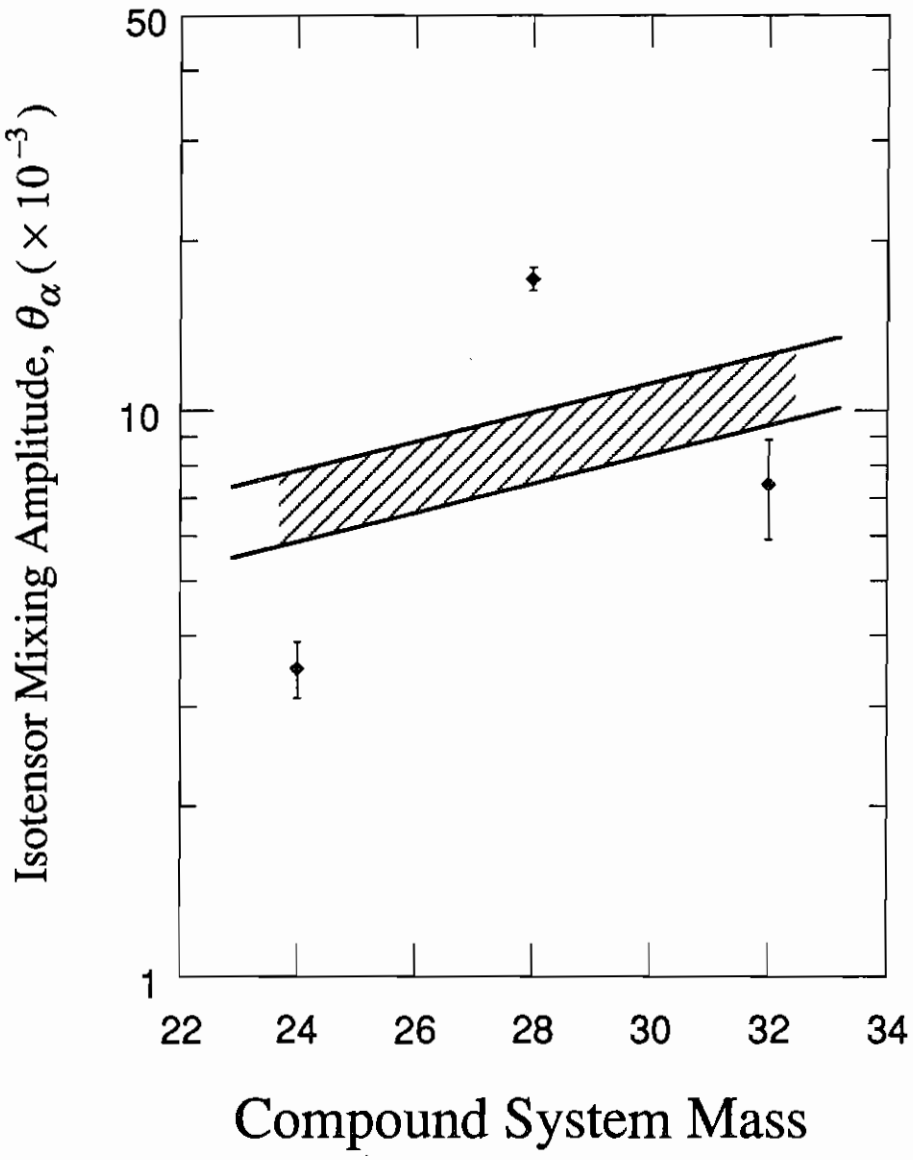


Figure [7.11]: T-mixing amplitudes derived from this work (points) compared with the values derived from the model of [Wii83] (hatched area).

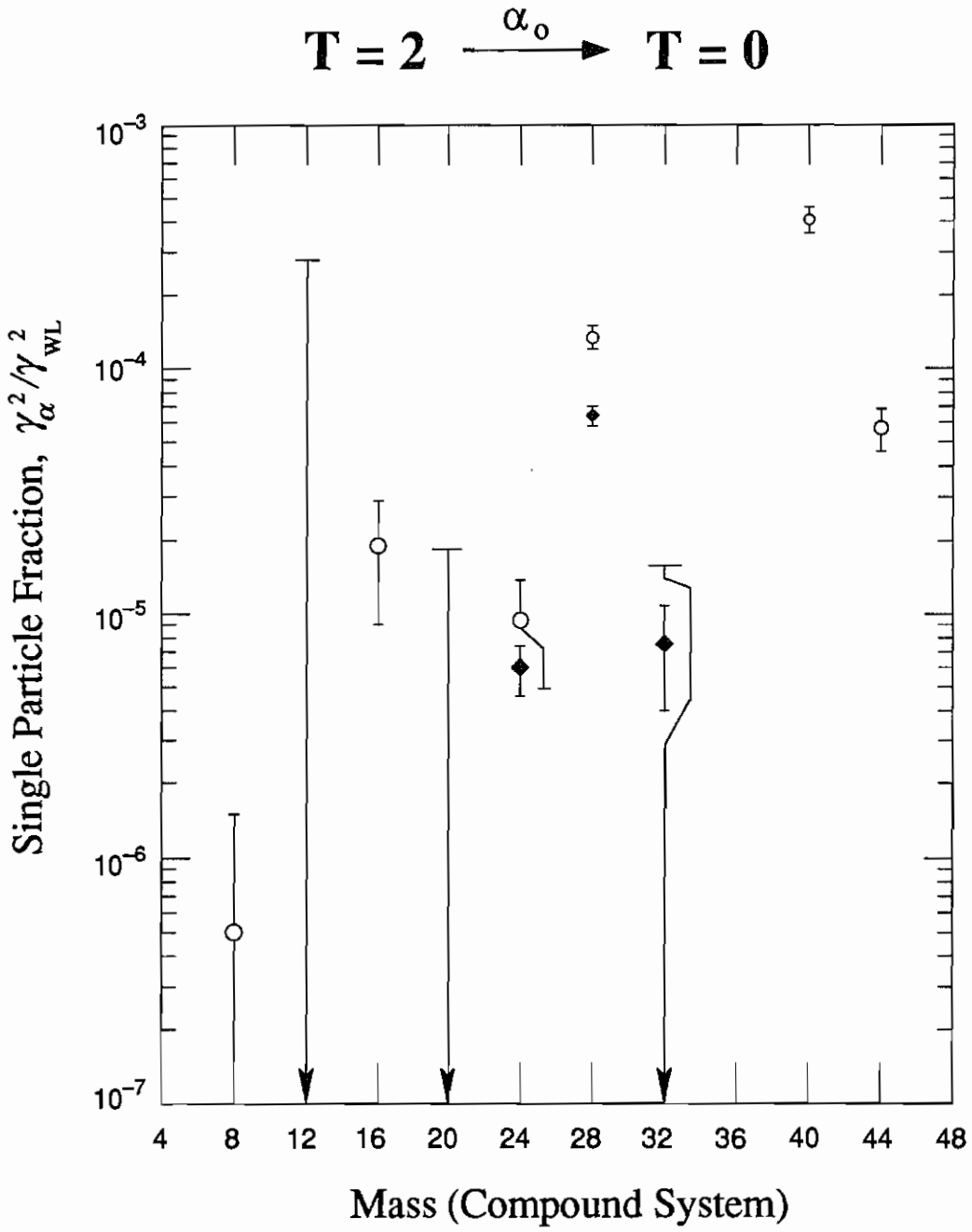


Figure [7.12]: Results from this work (solid diamond points) included with other available experimental data (circular points and upper limit error bars) [Fre79] [Hes87]. Wigner limit reduced widths are calculated according to [Fre79].

procedures, the effective storage and analysis of the raw data, and the overall generality, efficiency, and power of the resonance fitting routines.

7.6 Comments and Suggestions for Future Work

There have been great strides in the TUNL/UNC high resolution work in the last decade or more, but there is room for continued improvement. The following sections contain some suggestions and thoughts about the condition and future of high resolution data acquisition.

7.6.1 Ramp-Ramp System

Although much progress was made during the course of these experiments, the data acquisition and analysis procedures used for the direct measurement of the beam energy resolution should continue to evolve as users become more familiar with the system and the underlying principles. Hardware improvements to the ramp-ramp system might include thinner slits and NMR field measurements of both 90° magnets. These suggestions originate with and are elaborated on by Mooney [Moo89].

7.6.2 Targets

Since some of the resonance parameters may be affected by the overall normalization of the excitation functions, careful determinations of target thicknesses are essential. Since errors are propagated, one should be wary of determining thicknesses by comparison with other targets, and Rutherford scattering measurements should be used whenever possible. Thin target fabrication continues to be a problem, and helpful diagnostics are few. Many suggestions for improvements in target making are presented

in section 4.3, so they will not be repeated here.

7.6.3 TUNL High Resolution System

We have seen the importance of good energy resolution in studying narrow resonances. Continued emphasis should be placed on the proper operation of the accelerator feedback system, since it can be largely responsible for determining the incident beam's energy resolution. In the future, the planned tandem accelerator upgrade to a pelletron charging system should provide improved beam energy resolution, taking some of the burden off of the present slit feedback system.

Related to the question of beam energy spread is the operation of the high resolution beam transport system. In order to insure maximum beneficial effects on the beam energy resolution, careful attention should be given to proper settings, not only of the slits, but of the various focusing elements in the system. Careful beam optics calculations would be helpful in determining these settings. These calculations might also point the way to improvement of the beam transmission which generally has been fairly poor (< 30 %). To this end, I believe that a complete realignment of the system might be instrumental in raising the percentage of transmitted beam. In fact, the slits between the two 90° magnets have been observed to be offline by about 0.25 cm. Of course, realignment of these slits would necessitate a check and perhaps a recalibration of the overall energy accuracy of the system.

The beam transport from the exit of the 90-90 system to the target also provided some problems during several of the high resolution runs. Particularly annoying was the behavior of the beam-line slit feedback system. The system was designed to keep the beam centered on target using steerers driven by feedback signals from selected slits positioned either before or after the target. In practice, using this system injected noise

into the 90-90 magnet system, causing a much less stable beam on target. At times the noise was enough to break the 90-90 magnet NMR feedback loop, causing the beam to be lost altogether. Since we were never able to pin down the exact cause of this aggravation, I cannot offer any definite suggestions for the resolution of this problem. Perhaps it might be worthwhile to examine carefully the way in which the power supplies are wired to the steerers and the control panel.

7.6.4 Analysis

For the extraction of parameters describing isolated resonances, our programs worked very well. The decision to use MINUIT to perform the searches seems to have paid off by allowing larger parameter sets to be dealt with in a reasonable amount of time. The modularity inherent in the C programming language and the generality of most of the program functions should allow application of much of the program to other isolated resonance analyses. Although the Ridge computer, on which the programs ran, is being phased out at UNC, the functions should be reasonably portable to other environments, especially those running UNIX. Since the slowest part of the program involves the convolution of the cross section and resolution functions, it has been suggested that perhaps some type of Fourier transform method might be useful. Descriptions of algorithms for the so-called fast Fourier transform (FFT) and their use in evaluating convolutions are well represented in the literature (e. g., [Bur85]). Although some data may not be appropriate for this treatment (too "noisy"), it may be worth exploring further.

7.6.5 Future Experiments

It is tempting to suggest redoing some of these experiments with better experimental conditions; e.g. better energy resolution, more accurate measurement of the

resolution, and/or better statistics. When analyzing the data, I found myself wishing for better data, and I suppose that one is never completely satisfied. In particular, the $^{31}P(p,p)^{31}P$ / $^{31}P(p,\alpha)^{28}Si$ seemed to suffer from poor statistics and a somewhat larger-than-ideal energy step size. However, given the status of the similar published data and the errors inherent in the analysis, the data are sufficiently precise to provide results which are some of the best available. The pygmy resonance outlined in section 6.4.1 is perhaps the most compelling reason to continue the study of any of the present experiments with better statistics and resolution. Although it is not clear what factors contribute to the observation of this effect, a more careful study of this system might prove to be interesting.

7.6.6 A Final Note

Many of the problems cited above are minor ones, but ones which are nagging in the sense that they represent an overall degradation of the capabilities of the high resolution system. TUNL has long been known as a leader in low energy, high resolution nuclear physics. The philosophy of "if it ain't broke, don't fix it" works well in some instances, but it also may be detrimental to the invocation of new ideas and improvements necessary to remain at the forefront of scientific endeavors. The development of our beam energy resolution measurement scheme is certainly a step in the right direction, and I look for continued improvements of this system. The planned accelerator upgrade is also encouraging and should provide the impetus for future high resolution studies. With the unique experimental tools available, perhaps what is really needed is not simply a set of minor improvements but a continued commitment to the kind of high resolution work which has been performed so successfully at TUNL over the past few decades.

APPENDIX A: DATA HANDLING PROGRAMS

Following is a brief description of the programs used for data acquisition and for the calculation of cross sections from the raw data.[1] The files are grouped into three categories: managing the data acquisition, "online" calculations (HSYS), and "offline" calculations (OSYS). All of the programs described here run on the TUNL VAX 11/780, and the listings may be found in notebooks at UNC (currently in Phillips Hall, room 175).

A.1 Programs for High Resolution Data Acquisition

Note that the names and descriptions are general ones, and files used during any particular run may be different from those listed. Also, not all files are needed for every run.

----- Initialization files -----

hrlogin.com	clear directory, get our files, call iden, set up xsys
initsym.com	run by hrlogin, initializes global symbols

--- DAP and EVAL codes for XSYS subprocesses -- in archive HRDAPEVL.AR ---

d1d2.dap	detectors one and two .dap file
d3d4.dap	detectors three and four .dap file
d5d6.dap	detectors five and six .dap file
d7d8.dap	detectors seven and eight .dap file
dndm.evl	from this template .evl files are written by evalwrt.com
hrsc.dap	.dap file for scaler subprocess

[1] Descriptions and listings of the resonance fitting programs are in appendix F.

parm.dap	parameter .dap, reads csd, nmr, and ramp
parm.evl	parameter .evl (maintains csd, nmr, and ramp data areas)

----- Scaler display and other routines -- in archive MISC.AR -----

boxusr.for	manage the scaler display
files.txt	contains this text
hrfiles.set	dummy file to signal presence of hires files
subrem.for	subroutine called to estimate time remaining in run
wrapup.sh	tools shell script to clean up and save non-data files after a run

----- Files to start/stop data acquisition -- archived in RUNCOM.AR -----

beginrun.com	begin a run
endrun.com	end a run, call for calculations if desired
restart.com	clear appropriate data areas, reset flags, start new

----- Setup routines, generally run once at beginning of an experiment to define data acquisition environment, use of adc's, etc. - in archive SETUP.AR -----

boxsu.com	box setup
hralloc.com	allocate data areas
hrscal.com	scaler setup
hrsetsub.com	setup necessary subprocesses (runs boxsu and hrscal)
iden.com	global symbols and logical names for hires
runiden.com	global symbols unique to particular run
symsave.com	save selected global symbols in file symrest.com

----- Useful, general routines that are independent of the data acquisition environment established, e.g., by setup.ar -- in UTIL.AR -----

boxchange.com	move box display to new terminal
bstep.com	preferable to estep when making small changes
debug.com	called on (error .and. g_debug) by some routines, allows retry
ener.com	read nmr, write field and beam energy implied by field
error.com	called on error detected by program or DCL
estep.com	step magnet using "nmr estep" command of logstuff
estepman.com	standin for estep when automatic stepping can't be done
evalwrt.com	make .evl files from template
listsu.com	list setup results (lmem, scal list, gate list, type symrest)
rwsym.com	read output of command into global symbol
setstep.com	sets EVAL variable 'step' for each active MBD channel
suppress.com	check that electron suppressor is on
widegate.com	set gates for storing portion of spectrum for each target ramp value
wrtdat.com	write selected data areas (for step excitation only)
xsymrest.com	define XSYS commands (can be run inside a subprocess)

A.2 HSYS -- Programs for Online Cross Section Calculation

***** Main analysis programs *****

calc.for	Main file -- calls all other necessary files
hinfup.for	Updates the info data area from the text file infcom.txt
hmastup.for	Updates the master data area from the text file mastcom.txt
hmastdmp.for	Writes contents of master data area to the terminal
hinfdmp.for	Writes contents of info data area to the terminal

hscaldmp.for	Writes contents of scaler data area to the terminal
hcalcdmp.for	Writes contents of calc data area to the terminal
solang.for	Calcs. solid angles, if needed, and writes to master DA

***** User defined text files to be used by calculation programs *****

infcom.txt	Text file with information to be written into info data area
mastcom.txt	Text file with information for inclusion in master data area

***** Include files for use by all programs *****

hparam.for	Include file containing parameter definitions
hinfcom.for	Include file defining infcom common block and declaring equivalences of info variables (variables whose names begin with i_)
hmastcom.for	Include file defining mastcom common block and establishing equivalences of master variables (those beginning m_)
hscalcom.for	Include file defining scalcom common block and establishing equivalences of scaler variables (those beginning s_)
hcalccom.for	Include file defining calccom common block and establishing equivalences of calc variables

***** Programs for compiling and linking *****

halign.for	Object, used by linker, to force common block page alignment
hlink.com	Command file used to link hrsystem files wth XSYS
hlink.opt	Contains command line used by XSYS linker
hmake.com	Command file to produce executable version of file in hprog
hmakelib.com	Command file to produce library from hlib.ar programs

***** Subroutines in object library to be called by the main programs *****

bg.for	Calculates background and stores in background data area
calcnorm.for	Calculates normalization for a particular detector and gate
calcsig.for	Calculates cross section for a particular detector and gate
detcalc.for	Calls calcsig for each detector, gate, and step
detnorm.for	Calls calcnorm for each detector, gate, and step
detsum.for	Calculates background subtracted sums for each detector
findix.for	Finds index in X_DIR for given data area number
getsym.for	Updates the master data area with selected global symbols
gtran.for	Transfers gates from one data area to another
herror.for	Types error message and calls EXIT()
hmap.for	Maps hsys data areas to respective common blocks
scaldtc.for	Save scalers in scaler data area and calculate dtc's
wrtdat.for	Write data to plot files and to terminal

A.3 OSYS -- Programs for "Offline" Analysis of Ramp Data

Note that the offline analysis programs (OSYS) utilize and are, in fact, built around, the HSYS routines. Also, our notion of a "standard file" refers to the form of the cross section versus energy files which are written by HSYS.

addem	– generates repeated calls to hadd to sum different runs
addem2	– addem-like file specifically for the run of 10/87
addres	– addem-like file specifically for the resolution function of 10/87
buildup	– calculates counts per BCI in gate #1 to monitor carbon buildup on targets
comb	– combine standard files by averaging to the nearest points

comblin – combine standard files by linear interpolation between data points

combnp – combine standard files by averaging to the nearest point. Average over total energy range or energies common to all files (enhanced version of comb).

combspl2 – combine standard files of different step sizes using a cubic spline interpolation (output only over the common energy range)

dtccalc – dead time correction calculation for offline analysis (calls HSYS scaldtc)

enershft – adds arbitrary number to energies of standard files

error – filter to take data area output, add square root errors, and output in two column form

hadd – subroutine to add or subtract any types of data areas, with offset if desired (more general than XSYS ADD)

madd – main driver routine for subroutine hadd

mpulscorr – subroutine to correct for errors from time jitter in the ramp and normalize to an overall BCI

mult – for standard files, multiply cross section and errors by arbitrary number

mvgat – subroutine to shift a gate a specified number of channels

norm – normalize one standard file to another by equating their averages over their common energy range.

ocalc – main program to manage cross section calculations for offline analysis

overcorr – correct for overflow of I*2 data areas by adding multiples of 2^{16}

pulscorr – subroutine which uses the pulser excitation function to calculate correction for the time error in the target ramp voltage

resbci – convert XSYS data area dump file to relative BCI for resolution function measurement

reserr – produce square root errors for counts versus channel files

resfile – converts standard files for form suitable as input to RESMIN

resfile2 – convert counts versus channel files for input into RESMIN

scomb – combine standard files by averaging same array entries (does not consider actual energy associated with each point)

smush – average adjacent points of standard files

standard – puts summed data areas back into the standard format after adding runs

standard2 – streamlined version of standard for 11/87 run

standres – specialized version of "standard" for resolution function

subem – subtract runs from the cumulative total (calls hadd)

targnorm – when adding runs together, calculates factor to correct cross section for different target thicknesses from run to run

thicnorm – calculates and writes target thickness into the master data area

trangats – subroutine to transfer gates from one data area to another

wrtnz – write data areas to fortran files omitting channels with zero counts

wrtout – writes data area into fortran file with three column output (channel #, counts, and squre root of counts)

APPENDIX B: ESTIMATING TARGET ENERGY FOR DOPPLER BROADENING

In order to calculate the Doppler broadening in section 3.3, we need the total energy of vibration for the molecules of a lattice [Oma75] (quantal mean energy):

$$\bar{E} = \int_0^{\omega_m} \bar{\varepsilon}(\omega)\, g(\omega)\, d\omega \qquad [B.1]$$

where the integral is over the allowed frequencies, from zero to some maximum, ω_m, and $g(\omega)$ is the density of states function or phonon density spectrum. The average energy, $\bar{\varepsilon}$, of an oscillator in a solid at thermal equilibrium is given by:

$$\bar{\varepsilon} = \frac{\sum_{n=0}^{\infty} \varepsilon_n e^{-\beta\varepsilon_n}}{\sum_{n=0}^{\infty} e^{-\beta\varepsilon_n}} \quad \text{where } \beta = \frac{1}{kT}, \qquad [B.2]$$

k is the Boltzmann constant (= 8.63 x 10^{-5} eV/K/molecule), and T is the absolute temperature. We consider each quantum oscillator[1] equivalent to a mode of oscillation of the lattice, so for each oscillator (or mode) the allowed energies, ε_n, are

$$\varepsilon_n = \left(n + \frac{1}{2}\right)\hbar\omega \,. \qquad [B.3]$$

Substituting and simplifying[2] we get

$$\bar{\varepsilon} = \frac{\hbar\omega}{2} \coth\left(\frac{\hbar\omega}{2kT}\right) \qquad [B.4]$$

and putting all this together yields:

---

[1] Classically, $\varepsilon_n \to \varepsilon = kT$, and we get the result $\bar{\varepsilon} = kT$.

[2] Use

$$\sum_{n=0}^{\infty} e^{-xn} = \frac{1}{1-e^{-x}} \quad \text{and} \quad \varepsilon_n e^{-\beta\varepsilon_n} = \frac{\partial}{\partial\beta} e^{-\beta\varepsilon_n} \,.$$

$$\overline{E} = \int_0^{\omega_m} g(\omega) \frac{\hbar\omega}{2} \coth\left(\frac{\hbar\omega}{2kT}\right) d\omega \quad . \qquad [B.5]$$

B.1 Einstein Model

The Einstein model for estimating the mean quantal energy treats the oscillators as isolated and independent with a common frequency, ω_E. Thus, for the density of states[3]:

$$g(\omega) \rightarrow g_E(\omega) = \delta(\omega_E - \omega) \qquad [B.6]$$

so that,

$$\overline{E} = \frac{\hbar\omega_E}{2} \coth\left(\frac{\hbar\omega_E}{2kT}\right) \qquad [B.7]$$

where the *Einstein frequency*, ω_E, is related to the *Debye temperature*, Θ_D (in section B.2), by:

$$\hbar\omega_E = \frac{3}{4} k \Theta_D \quad . \qquad [B.8]$$

This leads to an energy per molecule (or atom) of

$$\overline{E} = \left[\frac{3\Theta_D}{8T} \coth\left(\frac{3\Theta_D}{8T}\right)\right] kT \quad . \qquad [B.9]$$

Also, we can see from the series expansion of $\coth(x)$ that $x\coth(x) \rightarrow 1$ as $x \rightarrow 0$ so we get the correct high temperature (classical) limit of kT. Furthermore, when $T = \Theta_D$, we get an energy of 1.05 kT. Thus, whenever $T \geq \Theta_D$, the Einstein model yields energies within a few percent of the classical value.

---

[3] Note that we are interested in the energy per degree of freedom (average energy per lattice site) and thus the normalization condition for $g(\omega)$ is

$$\int_0^{\omega_m} g(\omega) d\omega = 1$$

rather than the usual

$$\int_0^{\omega_m} g(\omega) d\omega = 3N_A$$

for calculating the total energy of the lattice.

B.2 Debye Model

The Debye model postulates independent modes (not necessarily independent atoms) in the lattice with frequencies given by a linear dispersion relation, $w \propto q$ (where q is the wave number = $2\pi/\lambda$). The density of states for this linear dispersion relation is given by:

$$g(\omega) = 3\frac{\omega^2}{\omega_m^3} . \qquad [B.10]$$

The maximum allowed frequency, ω_m, also known as the Debye frequency, ω_D, defines the Debye temperature, Θ_D, by:

$$\hbar\omega_D = k\Theta_D . \qquad [B.11]$$

Within the approximations of the Debye model, the expression for the mean quantal energy reduces to:

$$\overline{E} = \frac{3}{8}\hbar\omega_D + \frac{3\hbar}{\omega_D^3}\int_0^{\omega_D} \frac{\omega^3}{e^{\beta\hbar\omega}-1} d\omega . \qquad [B.12]$$

Changing variables to $t = \beta\hbar\omega$, we have:

$$\overline{E} = \frac{3}{8}k\Theta_D + 3kT\left(\frac{T}{\Theta_D}\right)^3 \int_0^{\frac{\Theta_D}{T}} \frac{t^3}{e^t-1} dt$$

$$= \left[\frac{3\Theta_D}{8T} + \mathbf{I}\right] kT \qquad [B.13]$$

where **I** represents the Debye integral:

$$\mathbf{I} = \frac{3}{x^3}\int_0^x \frac{t^3}{e^t-1} dt \qquad [B.14]$$

whose values are tabulated in [Abr72]. Note, in the high temperature limit, $\mathbf{I} \rightarrow 1$, and we

get the classical limit value, kT. Also, at $T = \Theta_D$ the energy is 1.05 kT, again providing $\overline{E}$ within a few percent of the classical value for temperatures $T \geq \Theta_D$.

As an example, for the $^{24}Mg(\alpha,\alpha)^{24}Mg$ experiment we have $\Theta_D = 318$ K and $T =$ 330 K so that the Einstein model yields:

$$\overline{E} = 1.04\,kT \cong 30\ \text{meV}.$$

For the Debye model, we have, $x = .96$ and $\mathbf{I} \sim 0.7$ so that:

$$\overline{E} = 1.04\,kT \cong 30\ \text{meV}.$$

As expected, both models provide approximately the same results, and the value is close to the classical value, kT.

APPENDIX C: CONVOLUTION

Since we are dealing with data which is of a discrete nature, the convolution integral required in section 5.2 can be replaced by a sum, and the resulting *discrete convolution* is given by:[1]

$$y(E) = f * g = \sum_{k=-\infty}^{+\infty} g(E_k) f(E - E_k) \qquad [C.1]$$

or

$$y_n = (f * g)_n = \sum_{k=-\infty}^{+\infty} g_k f_{n-k} \qquad [C.2]$$

If either of the functions $g(E)$ or $f(E)$ go to zero at the wings then the convolution will be zero outside of some interval, and the sum can be reduced to a finite interval thus making numerical calculation possible. Therefore, for the purposes of the computer evaluation, it is necessary to assume that the functions are zero outside the range of the data arrays. One consequence of this assumption is that, in order to yield reasonable answers for functions which do not go to zero (e.g. the excitation functions themselves), one must calculate "extra" wing values of the function to be included in the convolution sum. These extra long wings insure that the convolution result is accurate within the original range of the function.

Intuitively, the operation of convolution is sometimes confusing and is usually dealt with in terms of Fourier transforms and the so-called convolution theorem.

---

[1] The energy interval, ΔE (dE), need not be included explicitly in the sum if the ΔE are all equal for both functions and if the functions themselves are normalized consistently within this convention. In this sense the sum may be thought of not as an approximation to the integral but rather as a redefinition of the convolution for discrete functions.

However, more helpful discussions and insights into evaluating discrete convolutions over finite ranges can be found in textbooks on digital signal analysis [Bea79] [McC79] [Pap77]. There are several steps in determining the convolution:

1) Fold f_k; that is, form the array f_{-k};

2) Shift f_k for pertinent values of n, resulting in f_{n-k};

3) Multiply g_k and f_{n-k} term by term for each particular n;

4) Sum product over all values of k.

The total resolution function is produced from a convolution of the incident beam resolution, target straggling, and Doppler broadening distributions by a program written corresponding to the above prescription. Since this convolution is done only once, outside the resonance fitting program, speed is not important, and the program does not need to compromise the clarity of the algorithm. This accuracy of this version of the routine was verified on functions whose convolutions are known analytically (Gaussians, Lorentzians, and delta functions).

On the other hand, the convolution subroutine used to fold the resolution function with the theoretical cross section may be called hundreds of times, and some shortcuts were implemented. The basic algorithm was preserved, but the output function must be such that the energy axis (x-axis) corresponds with that of the data. Thus, there are some points, those outside the data range, which do not need to be calculated. This streamlined routine produced the same results, over the data energy range, as did the more time-consuming stand-alone version. For more specific information about the convolution routines, see the documented program listings in appendix F.

APPENDIX D: CHI-SQUARED

The most general form of a chi-squared statistic[1] is given by [Jam80]:

$$\chi^2 = \sum_{ij} [\, d_i - f_i(a)] \, \mathbf{V_{ij}} [\, d_j - f_j(a)] \qquad [D.1]$$

where d_i are the observed values (data), f_i are the fitting values as functions of the parameters a, and $\mathbf{V_{ij}}$ is the inverse of the data covariance matrix. If the observations are statistically independent, then $\mathbf{V_{ij}}$ is diagonal so that

$$\chi^2 = \sum_i \frac{(d_i - f_i(a))^2}{e_i^2} \qquad [D.2]$$

where e_i^2 are the variances associated with the data points, $1/e_i^2$ being the diagonal elements of $\mathbf{V}$. Also, we note that, for counting statistics, $e_i^2 = d_i$, so that the chi-squared may also be written

$$\chi^2 = \sum_i \frac{(d_i - f_i(a))^2}{d_i} \quad , \qquad [D.3]$$

sometimes referred to as Neyman's chi-squared.

More generally, the chi-squared statistic defined in equation [D.2] can also be referred to as a *weighted least squares*, where $1/e_i^2$ are the weights. This terminology emphasizes the fact that the weights may not always be the variances and perhaps only the relative weights are known. In this case there is an unknown overall normalization, and the parameter errors determined by examining changes in χ^2 will be proportional to this overall factor.

---

[1] Here a chi-square statistic is any function of the data and fit that is asymptotically distributed in the classical chi-square form [Bak84].

Also, one encounters a chi-squared of the form

$$\chi^2 = \sum_i \frac{(d_i - f_i(a))^2}{f_i} \qquad [D.4]$$

which may be labelled a "true," "statisticians," or Pearson's chi-squared, and other similar statistics (e.g., equation [D.3]) are referred to as "chi-squared-like" or "modified chi-squared." However, within our framework and definitions, it is proper to continue to refer to equation [D.2] as chi-squared. In general, the terms "chi-squared," or "chi-squared statistic," or "weighted least squares" will be used interchangeably. Also, we will regard the weights, $1/e_i^2$, as reflecting the true statistical errors in the data points. However, as mentioned above, we should keep in mind that should there be some unknown overall or systematic error, then the parameter errors derived from the change in χ^2 may be proportional to some overall normalization.

APPENDIX E: APPROXIMATING DEBYE TEMPERATURES FROM ELASTIC CONSTANTS

The Debye temperature, Θ_D, is given by:

$$k\Theta_D = h\nu_m \qquad [E.1]$$

where ν_m is the highest allowed vibrational frequency. Roughly following the treatment and notation of [Sei40], we set the total number of modes with frequencies less than ν_m equal to the total number of degrees of freedom:

$$\frac{4\pi V}{c^3}\nu_m^3 = 3N \quad , \qquad [E.2]$$

and, solving for the Debye temperature, we get

$$\Theta_D = \left(\frac{3N}{4\pi V}\right)^{\frac{1}{3}}\frac{h}{k}c \qquad [E.3]$$

where N is the total number of atoms, V is the volume of the sample, and c is the average speed of vibration of the medium given in terms of the transverse and longitudinal velocities by:

$$\frac{3}{c^3} = \frac{1}{v_l^3} + \frac{2}{v_t^3} \quad . \qquad [E.4]$$

For an isotropic solid, the compressibility, κ, can be related to v_t and v_l by

$$\frac{1}{\kappa} = \rho v_l^2 - \frac{4}{3}\rho v_t^2 \qquad [E.5]$$

where ρ is the density. If we assume that the transverse velocity of vibration is half the longitudinal velocity, then we can relate c to the compressibility of the solid, by:

$$c = \left(\frac{3}{17}\right)^{\frac{1}{3}}\sqrt{\frac{3}{2\rho\kappa}} \qquad [E.6]$$

Substituting this value for the mean velocity and rewriting the number of atoms per unit

volume in terms of density and atomic weight gives us:

$$\Theta_D = \left(\frac{9\rho N_0}{68\pi A}\right)^{\frac{1}{3}} \frac{h}{k}\sqrt{\frac{3}{2\rho\kappa}} \qquad [E.7]$$

where A is the atomic weight and N_0 is Avogadro's number.

For phosphorus, $\rho = 2.2$ g/cm^2, $\kappa = 5.56 \times 10^{-12}$ cm^2/dyne, and $A = 31$ g/mole. These values yield $\Theta_D = 204$ K, corresponding to a mean quantal energy of 1.02 kT or 28.9 meV from the Debye model at a target temperature of 328 K.

APPENDIX F: RESONANCE FITTING PROGRAM LISTINGS

The following functions, along with MINUIT, comprise the majority of the resonance fitting program RESMIN. The routines which are part of the MINUIT optimization package are not listed. These programs were written in C and ran on a Ridge 32 workstation at UNC. Some of the functions are adapted from those used by [Moo89] in the analysis of data from polarized proton scattering. As such, there are some references to analyzing power fits. However, many of the program features ignored the possibility of polarized projectiles and the program, as a whole, should be expected to work properly only for cross section excitation functions.

```
/*************************************************************************
*   backsl.c - Calculate background slopes and intercepts for group of
*              data sets from least squares fit to a straight line.
*              Stores these values in the XFCN structure for that
*              particular group. Can be run at stratup or as an
*              option in FCN. FCN also includes provision for making
*              any particular slope identically zero. Note that the
*              magnitudes of the cross section are given as MINUIT
*              parameters and these may be set to the values calculated
*              from the least squares. Since the energy points are
*              equally spaced, the slope may be calculated in mb/sr/channel
*              instead of (mb/sr)/eV.
*               [Formulas from Bevington chapter 6]
*/

#include "resfit.h"

void backsl(pinf,pdat)
struct INFO *pinf;
struct XFCN pdat[];
{
  int N, i, j, lrp, hrp, calcint, calcslp;
  double s1, s2, s3, s4, s5, s6, yval, yesq, delta, lincor;
  char ans[2];

  if (debug>1) fprintf(fpdeb,"\n\t\t========== backsl begin ==========\n");
  if (debug) fprintf(fpdeb,"backsl:\n");

  printf("Calculate slopes? "); scanf("%s",ans);
  calcslp = tolower(ans[0]);
  printf("Calculate y-intercepts? "); scanf("%s",ans);
  calcint = tolower(ans[0]);

  if ((calcint == 'y') || (calcslp == 'y')) {
    N = pinf->pts;
    printf("\n Total number of points in excitation function is %d \n", N);
    lrp = 0; hrp = 0;
    while ((lrp < 1) || (hrp > N) || (lrp > hrp)) {
      printf("\n What range of points contains the resonance? ");
      scanf("%d %d", &lrp, &hrp);
    }

printf("\n Points between %d and %d will be ", lrp, hrp);
printf("left out of the least squares calculation \n\n");

  /* for mb/sr/chan can use j instead of x-values */
for (i = 0; i < pinf->sets; i++) {
  printf("For detector %d  - ", i);
  printf("background type? ([l]inear, [e]xponential, or [n]o calc) >");
  scanf("%s", ans);
  if (tolower(ans[0]) != 'n') {
    pdat[i].btype = tolower(ans[0]);
    for (j=0,s1=0.0,s2=0.0,s3=0.0,s4=0.0,s5=0.0,s6=0.0; j < lrp; j++) {
      if (pdat[i].btype == 'e') {
        if (debug > 1) printf("exponential background\n");
        yval = log(pdat[i].xsc[j]);
        yesq = log(pdat[i].var[j]);
      }
      else {      /* default value is linear fit */
        yval = pdat[i].xsc[j];
        yesq = pdat[i].var[j];
      }
      s1 += 1.0/yesq;
      s2 += j*j/yesq;
      s3 += j/yesq;
      s4 += j*yval/yesq;
      s5 += yval/yesq;
      s6 += yval*yval/yesq;
    }
    for (j = hrp; j < N; j++) {
      if (pdat[i].btype == 'e') {
        yval = log(pdat[i].xsc[j]);
        yesq = log(pdat[i].var[j]);
      }
      else {      /* default value is linear fit */
        yval = pdat[i].xsc[j];
        yesq = pdat[i].var[j];
      }
      s1 += 1.0/yesq;
      s2 += j*j/yesq;
      s3 += j/yesq;
      s4 += j*yval/yesq;
      s5 += yval/yesq;
      s6 += yval*yval/yesq;
```

```
        }
        delta = s1*s2 - s3*s3;
        if (calcslp == 'y') pdat[i].bslp = (s1*s4 - s3*s5)/delta;
        if (calcint == 'y') pdat[i].bint = (s2*s5 - s3*s4)/delta;
        lincor = (s1*s4 - s3*s5)/sqrt(delta*(s1*s6 - s5*s5));

        if (pdat[i].btype == 'e') {
          printf("For det %d w/ c*exp(ax) fit: a = %lf\n",i+1,pdat[i].bslp);
          printf("                             c = %lf\n",exp(pdat[i].bint));
        }
        else {
          printf("For det %d :  slope = %lf\n", i+1, pdat[i].bslp);
          printf("              intercept = %lf\n", pdat[i].bint);
          printf("              linear corr. coeff. = %lf\n", lincor);
        }
      }
    }
  }
  if (debug>1) fprintf(fpdeb,"\n\t\t========== backsl end ==========\n");
}
```

```
/*****************************************************************
*  calc.c  -  This is the kernel of resfit.  Accepts parameters directing
*             the construction of artificial excitation functions, and
*             calls routines to compare the functions with data, and to
*             return the overall chi-square.
*/

#include "resfit.h"
int nres, conidx;
REAL xint[ALEN], yint[ALEN], xcon[ALEN];
REAL wr[MAXPTS], wi[MAXPTS], wsqr[MAXPTS];
REAL wrRaw[RAWPTS], wiRaw[RAWPTS];
REAL pyr[MAXPTS], pyi[MAXPTS], pysqr[MAXPTS];
REAL pyrRaw[RAWPTS], pyiRaw[RAWPTS];
struct XFCN elfit[MAXDETS], infit[MAXDETS];
extern int ninterp, inflag, new[], widflag;
extern struct XFCN eldat[], indat[];
extern struct INFO data, in, resf, bw;
extern double gfac;

REAL calc(p)
REAL *p;                        /* array of parameter values */
{
  void breit(), lorentz(), tranparm(), xsex(), norm();
  int interp(), convol();
  REAL expres(), chisqr(), moment();
  register int i, j;
  int do_breit, do_breit2, do_resf, do_pyg, pygidx, endelas;
  REAL ar, ai, mag, breal;      /* background amplitudes & cross section */
  double brfac, dx, ext, gamma, egiXgo, igiXgo;

  if (debug>1) fprintf(fpdeb,"\n\t\t========== calc begin ==========\n");
  if (debug) fprintf(fpdeb," calc: \n");

  switch (widflag) {

    case 'p':
      if (!inflag) {            /* elastic scattering only */
        gamma = p[WID]/p[BRI];
      }
      else {
        if (p[BRO] != 0.0) {
```

```
        gamma = p[WID]/p[BRO];
      }
      else {
        printf("** Exit channel branching ratio is zero - ");
        printf("cannot determine total width in this scheme!\n");
        printf("\t.....For elastic scattering, set bro = bri.\n");
        gamma = p[WID]/p[BRI];
      }
    }
    egiXgo = p[BRI]*p[BRI]*gamma*gamma; /* for elastic (GI = GO) */
    igiXgo = p[BRO]*p[BRI]*gamma*gamma; /* for inelastic */
    break;

  case 't':
    gamma = p[WID];
    egiXgo = p[BRI]*p[BRI]*p[WID]*p[WID]; /* for elastic (GI = GO) */
    igiXgo = p[BRO]*p[BRI]*p[WID]*p[WID]; /* for inelastic */
    /* igiXgo = p[GIGO]*egiXgo; */
    break;

  default:
    printf("**** Don't know how to interpret parameter one! \n");
    break;
}

/* Recalculate only data structures whose controlling parameters change */

do_breit = new[WID] || new[BRI] || new[BRO] || new[ER];
do_breit2 = new[WID] || new[BRI] || new[BRO] || new[ER] || new[DK];
do_resf = new[RESP];
do_pyg = (p[PK] != 0.0);        /* if Pk = 0 do not need to calculate */
if (debug>1) {
    fprintf(fpdeb,"do_breit = %d : do_resf = %d :", do_breit, do_resf);
    fprintf(fpdeb,"do_breit2 = %d : do_pyg = %d \n", do_breit2, do_pyg);
    fprintf(fpdeb,"p[RESP] = %lf \n",p[RESP]);
}

  /* p[RESP] should be > 0.0 and < 2 (unless recalc range of B-W) */
if ((do_resf) && (p[RESP] != 0.0)) {
   if (debug>1) fprintf(fpdeb,"p[RESP] = %lf \n",p[RESP]);
   dx = expres(p[RESP]);              /* everything in eV */
   nres = interp(resf.x,resf.y,resf.pts,xint,yint,bw.dx);
   norm(yint,yint,nres);
   if (debug>2) {
     fprintf(fpdeb,"Current resolution function for convolution \n");
     ext = (nres-1) * bw.dx;      /* x-range of resol. func. in eV */
     fprintf(fpdeb," Range is %lf eV -- \n", ext);
     fprintf(fpdeb,"\t from %lf to %lf \n", xint[0], xint[nres-1]);
     fprintf(fpdeb,"\t using %d points in %lf eV steps \n", nres, bw.dx);
     fprintf(fpdeb,"\t & mean %lf\n",moment(1,0.0,nres,xint,yint,NULL));
     for (i=0; i < nres; i++) {
       fprintf(fpdeb,"\t %lf \t %lf \n", xint[i], yint[i]);
     }
   }
}
if (do_breit) {
  /* if (p[BRO] == 0.0) p[BRO] = PDSMAL; */
  lorentz(gamma, p[ER], wrRaw, wiRaw);  /* same for elas & inel */
}
if (do_breit2 && do_pyg) lorentz(gamma, p[ER]+p[DK], pyrRaw, pyiRaw);

if (do_breit || do_breit2 || do_resf) {
  if (p[RESP] != 0.0) {
    if (do_breit || do_resf) {
      conidx = convol(xint, yint, nres, wrRaw, wiRaw, xcon, wr, wi);
    }
    if (do_pyg) {
      if (debug>1) fprintf(fpdeb,"\tcalculating for pygmy...");
      pygidx = convol(xint, yint, nres, pyrRaw, pyiRaw, xcon, pyr, pyi);
      if (pygidx != conidx)
          printf("\n ***** Results of convol not consistent\n");
    }
  }
  else {     /* width of res. func. is zero (e.g. Lorentzian approx.) */
       if (debug>1) fprintf(fpdeb,"zero width res. func. - no convolution\n");
    conidx = 0;
       for (i=0; i < data.pts; i++) {
         wr[i] = wrRaw[i*ninterp];
         wi[i] = wiRaw[i*ninterp];
         wsqr[i] = (wr[i]*wr[i]) + (wi[i]*wi[i]);
         pyr[i] = pyrRaw[i*ninterp];
         pyi[i] = pyiRaw[i*ninterp];
         pysqr[i] = (pyr[i]*pyr[i]) + (pyi[i]*pyi[i]);
       }
```

```
  }
}


switch (data.sets) {
  case 6:
    if (eldat[5].cflag) {
     tranparm(&p[DET5], &ar, &ai, &mag, &breal);
     xsex(ar, ai, mag, breal, &eldat[5], &data, gamma,
       egiXgo, wr, wi, pyr, pyi, p[PK], elfit[5].xsc);
    }
  case 5:
    if (eldat[4].cflag) {
     tranparm(&p[DET4], &ar, &ai, &mag, &breal);
     xsex(ar, ai, mag, breal, &eldat[4], &data, gamma,
       egiXgo, wr, wi, pyr, pyi, p[PK], elfit[4].xsc);
    }
  case 4:
    if (eldat[3].cflag) {
     tranparm(&p[DET3], &ar, &ai, &mag, &breal);
     xsex(ar, ai, mag, breal, &eldat[3], &data, gamma,
       egiXgo, wr, wi, pyr, pyi, p[PK], elfit[3].xsc);
    }
  case 3:
    if (eldat[2].cflag) {
     tranparm(&p[DET2], &ar, &ai, &mag, &breal);
     xsex(ar, ai, mag, breal, &eldat[2], &data, gamma,
       egiXgo, wr, wi, pyr, pyi, p[PK], elfit[2].xsc);
    }
  case 2:
    if (eldat[1].cflag) {
     tranparm(&p[DET1], &ar, &ai, &mag, &breal);
     xsex(ar, ai, mag, breal, &eldat[1], &data, gamma,
       egiXgo, wr, wi, pyr, pyi, p[PK], elfit[1].xsc);
    }
  case 1:
    if (eldat[0].cflag) {
     tranparm(&p[DET0], &ar, &ai, &mag, &breal);
     if (debug>1) {
       fprintf(fpdeb,"Calling xsex with ar, ai, mag, breal ");
       fprintf(fpdeb,"of %lf, %lf, %lf, %lf\n", ar, ai, mag, breal);
     }
       xsex(ar, ai, mag, breal, &eldat[0], &data, gamma,
         egiXgo, wr, wi, pyr, pyi, p[PK], elfit[0].xsc);
      }
      break;
    default:
      printf("calc: unexpected value of data.sets = %d",data.sets);
      break;
  }
/*          this section works okay, does same as next switch
  endelas = DET0 + (3 * data.sets);
  if (inflag) {
   if ((in.sets > 0) && (in.sets < 7)) {
    for (i=0; i < in.sets; i++) {
     if (indat[i].cflag) {
      tranparm(&p[endelas+3*i], &ar, &ai, &mag, &breal);
      xsex(ar, ai, mag, breal, &indat[i], &in, gamma,
          igiXgo, wr, wi, pyr, pyi, p[PK], infit[i].xsc);
     }
    }
   }
   else {
    printf("calc: unexpected value of in.sets = %d", in.sets);
   }
  }
*/

  if (inflag) {
   switch (in.sets) {
    case 6:
     if (indat[5].cflag) {
      tranparm(&p[DET5+(3*data.sets)], &ar, &ai, &mag, &breal);
      xsex(ar, ai, mag, breal, &indat[5], &in, gamma,
          igiXgo, wr, wi, pyr, pyi, p[PK], infit[5].xsc);
     }
    case 5:
     if (indat[4].cflag) {
      tranparm(&p[DET4+(3*data.sets)], &ar, &ai, &mag, &breal);
      xsex(ar, ai, mag, breal, &indat[4], &in, gamma,
          igiXgo, wr, wi, pyr, pyi, p[PK], infit[4].xsc);
     }
    case 4:
     if (indat[3].cflag) {
```

```
      tranparm(&p[DET3+(3*data.sets)], &ar, &ai, &mag, &breal);
      xsex(ar, ai, mag, breal, &indat[3], &in, gamma,
          igiXgo, wr, wi, pyr, pyi, p[PK], infit[3].xsc);
    }
  case 3:
    if (indat[2].cflag) {
      tranparm(&p[DET2+(3*data.sets)], &ar, &ai, &mag, &breal);
      xsex(ar, ai, mag, breal, &indat[2], &in, gamma,
          igiXgo, wr, wi, pyr, pyi, p[PK], infit[2].xsc);
    }
  case 2:
    if (indat[1].cflag) {
      tranparm(&p[DET1+(3*data.sets)], &ar, &ai, &mag, &breal);
      xsex(ar, ai, mag, breal, &indat[1], &in, gamma,
          igiXgo, wr, wi, pyr, pyi, p[PK], infit[1].xsc);
    }
  case 1:
    if (indat[0].cflag) {
      tranparm(&p[DET0+(3*data.sets)], &ar, &ai, &mag, &breal);
      if (debug>1) {
        fprintf(fpdeb,"Calling xsex with ar, ai, mag, breal ");
        fprintf(fpdeb,"of %lf, %lf, %lf, %lf \n", ar, ai, mag, breal);
      }
      xsex(ar, ai, mag, breal, &indat[0], &in, gamma,
          igiXgo, wr, wi, pyr, pyi, p[PK], infit[0].xsc);
    }
    break;
  default:
      printf("calc: unexpected value of in.sets = %d", in.sets);
      break;
  }
 }

 if (debug>1) fprintf(fpdeb,"\n\t\t========== calc end ==========\n");

 return (chisqr());
}
```

```
/*********************************************************************
*    cconv.c -- Convolute distributions (x1,y1) and (x2,y2) to produce
*           (x,y). Required: distributions are equally spaced in x.
*           Input distributions are assumed to be normalized.
*           Result will not be. Also, y1 and y2 are considered to be
*           zero outside the ranges of x1 and x2 respectively.
*           This version assumes y2 is actually two distributions
*           with a common x-axis, as might be the case for a complex
*           function. Doing both parts saves having to call twice.
*/
#include "/usr/ref/inc/stdinc.h"

int cconv(x1,y1,n1,x2,y2r,y2i,n2,x,yr,yi)    /* return # pnts in result */
int n1, n2;                  /* # pnts in distributions 1, 2 */
double *x1, *x2, *x;     /* independent variables for 1, 2, and result */
double *y1, *y2i, *y2r, *yr, *yi; /* dependent variables for 1, 2, and result */
{
    int i, j, n;
    double dx, dx1, dx2, *px;

    if (debug) fprintf(fpdeb," cconv: \n");

    n = n1 + n2 - 1;                  /* # pnts in result */
    if (debug>1) fprintf(fpdeb," cconv:\t # pnts, n = %d \n",n);
    x[0] = x1[0] + x2[0];             /* starting x-value */
    x[n-1] = x1[n1-1] + x2[n2-1];         /* max x-value */
    dx = (x[n-1]-x[0])/(n - 1);           /* step value */

    dx1 = (x1[n1-1] - x1[0])/(n1 - 1);   /* should have dx=dx1=dx2 */
    dx2 = (x2[n2-1] - x2[0])/(n2 - 1);
    if (((dx1-dx2) > PDSMAL) || ((dx1-dx) > PDSMAL))
        printf("******delta x's not all equal \n");
    if (debug>1)
      fprintf(fpdeb," cconv:\t dx=%lf dx1=%lf dx2=%lf\n",dx,dx1,dx2);

    for (i=0, px=x; i < n ; i++, px++) {
      *(yr+i) = 0.0;          /* initialize result */
      *(yi+i) = 0.0;          /* initialize result */
      *px = *x + i*dx;        /* calc x-values for result*/
      if (debug > 4) fprintf(fpdeb,"\t x[%d]=%lf \n",i,x[i]);
    }
```

```
        /* do convolution to get y-values */
    for (i=0 ; i < n1 ; i++) {
      for (j=0 ; j < n2 ; j++) {
        yr[i+j] += y1[i]*y2r[j];
        yi[i+j] += y1[i]*y2i[j];
        if (debug > 5) fprintf(fpdeb,"yr[%d]=%lf \n",i+j,yr[i+j]);
      }
    }

    if (debug > 4) {
      fprintf(fpdeb,"cconv:  convolution distribution \n");
      for (i=0 ; i<n ; i++)
        fprintf(fpdeb,"\t%lf \t %lf \t %lf \n", x[i], yr[i], yi[i]);
    }
    return(n);
}
```

```
/*****************************************************************
*  cg.c  -  function to return Clebsch-Gordan coefficient
*        - uses algebraic formula from Brink and Satchler (2.34).
*        - passed arguments are two times actual values;
*          e.g. passed arguments of 1,1,2,1,-1,0 produce output
*          of <1/2 1/2 1 | 1/2 -1/2 0> (for <j1 j2 j3 | m1 m2 m3>).
*        - passed arguments must be integers.
*/

#include "/usr/ref/inc/stdinc.h"

double cg(j1,j2,j3,m1,m2,m3)
int j1, j2, j3, m1, m2, m3;    /*two times actual values to insure integers*/
{
      int imin, imax, i;
      int low(), ipow();
      double fact(), coef1, coef2, s, clebsch;

  if (debug) printf(" cg: \n");
  if (debug>1) fprintf(fpdeb," cg: <%d/2,%d/2,%d/2,%d/2 | %d/2,%d/2> \n",
                                        j1,j2,m1,m2,j3,m3);
              /* put in constraints which make coefficients vanish */
      if (m3 != (m1+m2)) return(0.0);
      if ((j1<0) || (j2<0) || (j3<0)) return(0.0);
      if ((ABS(m1)>j1) || (ABS(m2)>j2) || (ABS(m3)>j3)) return(0.0);

      coef1 =(fact((j2+j3-j1)/2)/fact((j1+j2+j3+2)/2))*
            fact((j1+j2-j3)/2)*fact((j1+j3-j2)/2)*(double)(j3+1)*
            fact((j1+m1)/2)*fact((j1-m1)/2)*fact((j2+m2)/2)*
            fact((j2-m2)/2)*fact((j3+m3)/2)*fact((j3-m3)/2) ;

      coef2 = sqrt(coef1) ;

      imin = -(low(0,(j3-j2+m1),(j3-j1-m2)))/2 ;    /* determine limits*/
      imax = (low((j1-m1),(j2+m2),(j1+j2-j3)))/2 ;  /* for sum       */

      s = 0;
      for (i=imin; i<=imax; i++)
            s = s + ipow(-1,i)*(1.0/(fact((j1-m1)/2 - i)*
               fact((j3-j2+m1)/2 + i)*fact((j2+m2)/2 - i)*
               fact((j3-j1-m2)/2 + i)*fact((j1+j2-j3)/2 - i)*fact(i)));
```

```
    clebsch = coef2 * s ;
    if (debug>1) fprintf(fpdeb," cg: returning %lf \n",clebsch);

    return(clebsch);
}
```

```
/*********************************************************************
*  chidet.c  - Calculate chi-square for fit to data. The global variable
*          "normoff" determines whether normalization is allowed to
*          change to improve the fit.
*/

#include "resfit.h"
extern struct INFO data;

REAL chidet(norm,fit,dat,esqr)
REAL norm, *fit, *dat, *esqr;     /* normalization, generated data,
                        observed data, and square of error */
{
  REAL tmp, sum;
  register int i;

  if (debug>1) fprintf(fpdeb,"\n\t\t========== chidet begin ==========\n");
  if (debug) fprintf(fpdeb," chidet: \n");
  if (debug>1) fprintf(fpdeb,"       norm %lf\n",norm);

            /* Calculate chi-square with (normalized?) fit */
  for (i=0, sum=0.; i< data.pts ; i++) {
    tmp = norm*fit[i] - dat[i];
    sum += tmp*tmp/esqr[i];
   }

  if (debug>1) fprintf(fpdeb,"chidet returning %lf \n",sum);
  if (debug>1) fprintf(fpdeb,"\n\t\t========== chidet end ==========\n");

  return (sum);
}
```

```
/*************************************************************************
*  chisqr.c  -  Calculate total chi-square for the fit(s) to data.
*               The global variable "fittype" determines whether cross
*               section data, analyzing power data, or both are fitted.
*/

#include "resfit.h"
extern struct XFCN eldat[], indat[], elfit[], infit[];
extern struct INFO data, in;
extern int normoff, inflag;
extern char fittype;

REAL chisqr()
{
  register int n;
  REAL normfit(), chidet(), tmp;

  if (debug>1) fprintf(fpdeb,"\n\t\t========== chisqr begin ==========\n");
  if (debug) fprintf(fpdeb," chisqr: \n");

                    /* compute average chi-squared [with norm] */
  if (fittype != 'a') {
    for (n=0, tmp = 0.; n < data.sets; n++) {
      if (eldat[n].cflag) {
        if (!normoff) {
          eldat[n].normsig = normfit(elfit[n].xsc,eldat[n].xsc,eldat[n].var);
        }
        tmp += chidet(eldat[n].normsig,elfit[n].xsc,eldat[n].xsc,eldat[n].var);
      }
    }
    if (inflag) {
      for (n=0; n < in.sets; n++) {
       if (indat[n].cflag) {
         if (!normoff) {
          indat[n].normsig = normfit(infit[n].xsc,indat[n].xsc,indat[n].var);
         }
         tmp +=chidet(indat[n].normsig,infit[n].xsc,indat[n].xsc,indat[n].var);
       }
      }
    }
/*     if (fittype != 's') {
            if (!normoff)
        eldat[n].normay = normfit(elfit[n].ay,eldat[n].ay,eldat[n].ayesqr);
       tmp += chidet(eldat[n].normay,elfit[n].ay,eldat[n].ay,eldat[n].ayesqr);
     }
*/
  }

/*  if (fittype == 'b') tmp /= 2.; */

  if (debug>1) fprintf(fpdeb,"chisqr returning %lf\n",tmp);
  if (debug>1) fprintf(fpdeb,"\n\t\t========== chisqr end ==========\n");

  return (tmp);
}
```

```
/*************************************************************
*  cm2lab.c - To convert center of mass to laboratory angle
*/

#include "/usr/ref/inc/stdinc.h"

double cm2lab(thcm,mpi,mti,mpf,mtf,elab,q)
double thcm, mpi, mti, mpf, mtf, elab, q ;
{
  double ecm, x, thlab ;

  if (debug) fprintf(fpdeb," cm2lab: \n");
  ecm = elab * mti/(mpi+mti) ;
  x = sqrt((mpi*mpf/(mti*mtf)) * ecm/(ecm+q)) ;

  thlab = atan(sin(thcm)/(x+cos(thcm))) ;

  if (debug>1) {
    fprintf(fpdeb," cm2lab: ecm is %lf ; x is %lf \n", ecm, x);
    fprintf(fpdeb," cm2lab: thcm is %lf ; thlab = %lf \n",thcm,thlab);
  }

  return(thlab) ;
}
```

```
/*************************************************************************
*    convol.c - Convolutes the real and imaginary parts of a Breit-
*               Wigner function with a resolution function (x1,y1).
*               Note this is a "specialized" convolution routine,
*               designed to work with these specific functions. Hence,
*               it does NOT calculate the full x range of the convolution
*               and it provides the result in step sizes and x values
*               to be comparable directly (chi-squared per point) to
*               the data. (A more general program is cconv.c)
*/

#include "resfit.h"
extern int ninterp;
extern struct INFO data, resf, bw;

int convol(x1,y1,n1,y2r,y2i,x,yr,yi)
int n1;
double *x1, *y1, *y2r, *y2i, *x, *yr, *yi;
{
  register int i, j, k;
  int idx;

  if (debug>1) fprintf(fpdeb,"\n\t\t========== convol begin ==========\n");
  if (debug) fprintf(fpdeb," convol: \n");

   /* calc starting x value - note assumes x1[0] is negative */
   /* starting x value is in general x1[0] + x2[0]       */
   /* and # of points would be n1 + n2 -1                */
  x[0] = x1[0] + bw.x[resf.pts - 1];  /* resf.pts refers to original resol.
                function upon which the wings of the B-W depend */
  if (debug>1) {
    fprintf(fpdeb,"x1[0] = %lf ;", x1[0]);
    fprintf(fpdeb,"bw.x[resf.pts-1] = %lf \n", bw.x[resf.pts-1]);
    fprintf(fpdeb,"bw.pts = %d \n", bw.pts);
    fprintf(fpdeb,"resf.pts = %d \n", resf.pts);
    fprintf(fpdeb,"ninterp = %d \n", ninterp);
    fprintf(fpdeb,"k limit = %lf \n", (data.ext+ resf.ext)/data.dx);
  }

  if (debug > 2) fprintf(fpdeb,"x-values for convolution result -- \n");
  for (k = 0; k < NINT((bw.ext - resf.ext)/data.dx) + 1; k++) {
    x[k] = x[0] + (k * data.dx);
```

```
    if (debug > 2) fprintf(fpdeb,"\t x[%d] = %lf \n", k, x[k]);
    yr[k] = 0.0;
    yi[k] = 0.0;
    for (i=0, j = resf.pts-1 + (k*ninterp); ((j>=0) && (i<n1)); i++, j--) {
  /* the index i should always be less than the number of points */
  /* but for j, 0 is a valid index and it should be allowed */
      if (debug > 3) fprintf(fpdeb,"indexes - (k,i,j) = (%d,%d,%d)\n",k,i,j);
      yr[k] += y1[i] * y2r[j];
      yi[k] += y1[i] * y2i[j];
    }
  }
  for (i = 0; (ABS(x[i] - data.xmin) > PDSMAL) && (i < data.pts); i++)
    if (debug>2)
      fprintf(fpdeb,"point %d ; difference is %le \n", i, x[i]-data.xmin);
  idx = i;
  if (debug>1) fprintf(fpdeb,"index into convol array is %d \n", idx);

  if (idx == data.pts) printf("**** Convol: x array does not match data!!!\n");

  if (debug>1) fprintf(fpdeb,"\n\t\t========== convol end ==========\n");
  return(idx);
}
```

```
/*********************************************************************
*   csl2cm - Converts lab cross section to center of mass
*            Pass masses, com energy, q-value, com angle in radians,
*            and returns multiplicative factor.
*/

#include "/usr/ref/inc/stdinc.h"

double csl2cm(ma,mA,mb,mB,Ecm,Q,tcr)
double ma, mA, mb, mB, Ecm, Q, tcr;
{
  double x, num, denom, factor;

  if (debug) fprintf(fpdeb," csl2cm: \n");

  x = sqrt((ma*mb)/(mA*mB) * Ecm/(Ecm+Q));

  num = 1 + (x*cos(tcr));
  denom = 1 + (x*x) + (2*x*cos(tcr));
  denom = denom * sqrt(denom);
  factor = num/denom;

  if (debug>1) {
    fprintf(fpdeb," csl2cm: x is %lf \n", x);
    fprintf(fpdeb," csl2cm: numerator = %lf : denominator = %lf\n",num,denom);
    fprintf(fpdeb," csl2cm: conversion for %lf degrees is %lf \n",tcr,factor);
  }

  return(factor);
}
```

```
/*****************************************************************
*    expres.c  -  Routine to expand (or contract) the resolution function according
*                 to the value of a MINUIT parameter without altering the value of
*                 the mean (or 1st moment). Always operates on the original
*                 distribution so that results  will be consistent. Since it only changes
*                 the shape, it uses only the x-values. The calling routine should
*                 normalize the result before passing it to the convolution routine.
*                 Also, input number of points equals output number of points but
*                energy ranges are not necessarily preserved.
*/

#include "resfit.h"
extern struct INFO resf;

double expres(factor)
double factor;          /* expansion factor */
{
  register int i;
  double moment();
  double dx, mom1, shift;

  if (debug>1) fprintf(fpdeb,"\n\t\t========== expres begin ==========\n");
  if (debug) fprintf(fpdeb," expres: \n");

  dx = factor * resf.dx;                /* new value of x interval     */
  if (debug>1) fprintf(fpdeb," new value for x interval is %lf \n",dx);

  for (i = 0; i < resf.pts; i++) {
    resf.x[i] = resf.xmin + (i * dx);    /* new x array starting at same */
  }                                      /* place but not over same range */

  mom1 = moment(1,0.0,resf.pts,resf.x,resf.y,NULL);
  if (debug>1) fprintf(fpdeb," shifted value for mean is %lf \n",mom1);

  shift = resf.mean - mom1;             /* shift x values for equal means  */

  for (i = 0; i < resf.pts; i++) {
    resf.x[i] += shift;
  }
  if (debug>1) fprintf(fpdeb,"\n\t\t========== expres end ==========\n");
  return(dx);
}
```

```
/*****************************************************************
*   fact.c - Function to return factorial of an integer
*/

#include "/usr/ref/inc/stdinc.h"

double fact(x)
int x;
{
     double y = 1.0 ;              /* returns 1 for zero factorial */
     if (x < 0) {
       printf("fact: negative argument");
       return(0.0);
     }
     while (x) y *= x-- ;
     return(y);
}
```

```
/*****************************************************************
*  fcn.c  - This is the single point of contact between the functions
*           which make up resfit and the calling program minuit.  The
*           variables passed as arguments are the only knowledge resfit
*           has of minuit's workings.  Minuit is written in FORTRAN which
*           passes everything by reference.  Thus scalar data types, which
*           the c compiler assumes have been passed by value, are
*           interpreted as pointers to scalar data types.  Also, since
*           called by a FORTRAN routine, assembly code name should include
*           a trailing underscore.  This is taken care of in the compiling
*           stage rather than naming the function fcn_().
*/

#include "resfit.h"
int debug, defile, debcount, normoff, new[MAXPARM], WrapTo = TOFILE;
int xcalc, useint;
REAL minnorm, maxnorm;
extern int nparm, nres, trunlhs, inflag, parmform, widflag;
extern REAL oldp[],smin[],smax[],imin[],imax[],amin[],amax[],xint[],yint[];
extern REAL nsavgsig, rmin, rmax;
extern char fittype;
extern struct XFCN eldat[], indat[], elfit[], infit[];
extern struct INFO data, in, bw;
FILE *fpdeb;

void fcn(nvarparm, gradient, retval, parm, flag)
INTEGER *nvarparm, *flag;
REAL gradient[], parm[], *retval;
{
  void initdat(), initplot(), wrap(), plot(), plotres(), getres(), getbw();
  void backsl(), mininfo(), wrtasdat();
  int trun();
  REAL calc(), shifter();
  int i, j, ians, selparm, nsets, detnum, parmnum;
  static int Calls = 0, CallsPerPlot = 0, ForcePlot = 0, firsttime = TRUE;
  double dfree, muldb;
  char ans, answer[2];
  FILE *fp, *fopen();

  switch (*flag) {

    case 1:
      if (!firsttime) {
       printf("\n\t ***** illegal call to FCN with flag 1 *****\n");
       break;
      }
      printf("fcn: debug level? (0 = off) > "); scanf("%d",&debug);

      if (debug < 0) {
        fpdeb = fopen("debug.out","w");
        defile = 1;
      }
      else {
        fpdeb = stderr;
        defile = 0;
      }
      debug = ABS(debug);
      debcount = 0;

      normoff = TRUE;
      printf("\n ========= normalization is ");
      printf( normoff ? "FIXED\n\n" : "VARIABLE\n\n");
      printf("Wrap output to file (resmin.out)\n");
      initdat(parm);
      initplot();
         printf("\n\tinitial number variable parameters is %d\n",*nvarparm);
      xcalc = 1;
      useint = 'n';
      firsttime = FALSE;
      break;

    case 3:
      *retval = calc(parm);
      wrap(nparm,*nvarparm,parm,*retval);
      break;

    case 4:
      for (i=0; i<nparm; i++)
        if (oldp[i] == D_NOVALUE)     /* only satisfied for 1st call */
          new[i] = TRUE;
        else
          new[i] = (ABS(parm[i]-oldp[i]) >= 1.0e-9);
      *retval = calc(parm);
      if (debug) {
```

```
    fprintf(fpdeb,"fcn: *retval = %lf\n", *retval);
    debcount++;
    if ((debcount > 5) && (!defile)) {
      printf("\n Cancel debug? "); scanf("%s",answer);
      if (tolower(answer[0]) == 'y') debug = 0;
      debcount = 0;
    }
  }
  for (i=0; i<nparm; i++) oldp[i] = parm[i];
  break;

case 6:
  normoff = !normoff;
  printf("======= normalization is ");
  printf( normoff ? "FIXED\n" : "VARIABLE\n");
  printf("for elastic data ..");
  printf(" cross section, analyzing power\n");
  for (i=0; i< data.sets; i++)
     printf("%f, %f\n", eldat[i].normsig, eldat[i].normay);
  if (inflag) {
    printf("for inelastic data ..");
    printf(" cross section, analyzing power\n");
    for (i=0; i < in.sets; i++)
      printf("\n%f, %f\n", indat[i].normsig, indat[i].normay);
  }
  if (normoff) {
    printf("set norms to 1.0 ?"); while (!(ans=getchar()));
    if (tolower(ans) == 'y') {
      for (i=0; i<data.sets; i++) eldat[i].normsig=(eldat[i].normay=1.0);
      if (inflag)
        for (i=0; i<in.sets; i++) indat[i].normsig=(indat[i].normay=1.0);
    }
  }
  else {
    printf("max percentage normalization [0 .. 100]");
    scanf("%lf", &maxnorm);
    minnorm = max(0., 1-maxnorm/100.);
    maxnorm = min(2., 1+maxnorm/100.);
  }
  break;

case 7:
  printf("change value of what parm ? ");
  scanf("%d",&selparm);
  printf("parm %2d = %f -- change to ?\n", selparm, parm[selparm-1]);
  printf("       ");
  scanf("%lf",&parm[selparm-1]);
  break;

case 8:
  printf("Wrap output to ");
  if (WrapTo == TOFILE)
    {WrapTo = TOSTDOUT; printf("stdout\n");}
  else
    {WrapTo = TOFILE; printf("file\n");}
  break;

case 9:                              /* plot data + fit */
  ForcePlot = 1;
  break;

case 10:                    /* schedule plotting of data + fit */
  printf("currently plotting after %d calls -\n",CallsPerPlot);
  printf("plot after ? calls [ 0 to cancel plotting ] : ");
  scanf("%d", &CallsPerPlot);
  Calls = 0;
  break;

case 11:                           /*turn on or off debugging */
  if (defile) fclose(fpdeb);
  printf("debug level is %d ; input new value: ",debug);
  scanf("%d", &debug);
  if (debug < 0) {
    fpdeb = fopen("debug.out","w");
    defile = 1;
  }
  else {
    fpdeb = stderr;
    defile = 0;
  }
  debug = ABS(debug);
  break;

case 12:              /* shift energy to minimize chi squared */
```

```
        printf("present value of resonance energy is %lf\n",parm[ER]);
        parm[ER] -= shifter();
        printf("new value of res energy = %lf \n",parm[ER]);
        new[ER] = TRUE;
    *retval = calc(parm);
        break;

case 13:            /* new resolution function and bw range */
    getres(bw.dx);
    getbw();
    new[RESP] = TRUE; new[ER] = TRUE;
    *retval = calc(parm);
    break;

case 14:        /* print value for chi-squared per degree of freedom */
    *retval = calc(parm);
    dfree = ((data.sets * data.pts) - *nvarparm - 1);
    if (inflag) dfree += in.sets * in.pts;
    printf("\n\tChi-squared per degree of freedom: %lf\n", *retval/dfree);
    break;

case 15:                /* clear output file resmin.out */
    printf("\n\t Clearing output file resmin.out.... \n");
    fp = fopen("resmin.out", "w");
    fclose(fp);
    break;

case 16:                /* write parameter file */
    wrtparm(parm);
    break;

case 17:                    /* truncate wings of data */
      /* note that this truncation is a ONE WAY process     */
      /* the wing data is not saved and cannot be retrieved */
    trunlhs += trun(&data,eldat);   /* ret val used to correct slopes */
    getbw();                /* recalc bw (mainly for speed) */
    new[ER] = TRUE;         /* force to do full calculation */
    *retval = calc(parm);
    break;

case 18:        /* write fit to file in 'fake' data format */
    wrtasdat(&data,eldat,elfit);
    break;

case 19:        /* test access to MINUIT arrays */
    mininfo();
    break;

case 20:    /* calc background slope & intercept by least squares */
    printf("\nFor elastic data...\n");
    backsl(&data,eldat);
/*    printf("Set MINUIT parameters to intercept values? ");
    scanf("%s",answer);
    if (tolower(answer[0]) == 'y') {
      detnum = 1;
      while (detnum != 0) {
        printf("\n which detector (0 to quit)? >"); scanf("%d",&detnum);
        if (detnum) {
          printf("parameter # of background cross section? > ");
          scanf("%d",&parmnum);
          if (eldat[detnum-1].btype == 'e') {
            parm[parmnum-1] = exp(eldat[detnum-1].bint);
          }
          else {
            parm[parmnum-1] = eldat[detnum-1].bint;
          }
        }
      }
    }
*/
    if (inflag) {
      printf("\n\nFor inelastic data...\n");
      backsl(&in,indat);
/*      printf("Set MINUIT parameters to intercept values? ");
      scanf("%s",answer);
      if (tolower(answer[0]) == 'y') {
        detnum = 1;
        while (detnum != 0) {
          printf("\n which detector (0 to quit)? >");
          scanf("%d",&detnum);
          if (detnum) {
            printf("parameter # of background cross section? > ");
            scanf("%d",&parmnum);
            if (indat[detnum-1].btype == 'e') {
```

```
            parm[parmnum-1] = exp(indat[detnum-1].bint);
          }
          else {
            parm[parmnum-1] = indat[detnum-1].bint;
          }
        }
      }
    }
*/
  }

case 21:
  printf("\n Use value of intercept for background cross section? ");
  scanf("%s",answer);
  useint = tolower(answer[0]);
  break;

case 22:        /* change value of background slopes */
  printf("\n\t\t for elastic (or only) data \n");
  for (i = 0; i < data.sets; i++) {
    printf("\t detector %d slope = %lf \n", i+1, eldat[i].bslp);
  }
  printf("change which detector? (0 to quit) "); scanf("%d", &ians);
  if ((ians > 0) && (ians <= data.sets)) {
   printf("current value %lf ; change to > ", eldat[ians-1].bslp);
   scanf("%lf", &eldat[ians-1].bslp);
  }
  if (inflag) {
    printf("\n\t\t for inelastic data \n");
    for (i = 0; i < in.sets; i++) {
      printf("\t detector %d slope = %lf \n", i+1, indat[i].bslp);
    }
    printf("change which detector? (0 to quit) "); scanf("%d", &ians);
    if ((ians > 0) && (ians <= in.sets)) {
     printf("current value %lf ; change to > ", indat[ians-1].bslp);
     scanf("%lf", &indat[ians-1].bslp);
    }
  }
  break;

case 23:        /* change value of background y-intercepts */
  printf("\n\t\t for elastic (or only) data \n");
  for (i = 0; i < data.sets; i++) {
    printf("\t detector %d y-intercept = %lf \n", i+1, eldat[i].bint);
  }
  printf("change which detector? (0 to quit) "); scanf("%d", &ians);
  if ((ians > 0) && (ians <= data.sets)) {
   printf("current value %lf ; change to > ", eldat[ians-1].bint);
   scanf("%lf", &eldat[ians-1].bint);
  }
  if (inflag) {
    printf("\n\t\t for inelastic data \n");
    for (i = 0; i < in.sets; i++) {
      printf("\t detector %d y-intercept = %lf\n",i+1,indat[i].bint);
    }
    printf("change which detector? (0 to quit) "); scanf("%d", &ians);
    if ((ians > 0) && (ians <= in.sets)) {
     printf("current value %lf ; change to > ", indat[ians-1].bint);
     scanf("%lf", &indat[ians-1].bint);
    }
  }
  break;

case 24:        /* turn off (or back on) calculations for detector */
  printf("\n\t\t for elastic (or only) data \n");
  for (i = 0; i < data.sets; i++) {
    printf("\t detector %d calculation is ", i+1);
    printf(eldat[i].cflag ? "ON\n" : "OFF\n");
  }
  printf("change which detector? (0 to quit) "); scanf("%d", &ians);
  if ((ians > 0) && (ians <= data.sets)) {
    eldat[ians-1].cflag = !eldat[ians-1].cflag;
    printf("detector %d calculation has been turned ", ians);
    printf(eldat[ians-1].cflag ? "ON\n" : "OFF\n");
    if (!eldat[ians-1].cflag) {
      printf("\n Associated MINUIT parameters should be FIXed \n");
    }
  }
  if (inflag) {
   printf("\n\t\t for inelastic data \n");
   for (i = 0; i < in.sets; i++) {
     printf("\t detector %d calculation is ", i+1);
     printf(indat[i].cflag ? "ON\n" : "OFF\n");
   }
```

```
    printf("change which detector? (0 to quit) "); scanf("%d", &ians);
    if ((ians > 0) && (ians <= in.sets)) {
      indat[ians-1].cflag = !indat[ians-1].cflag;
      printf("detector %d calculation has been turned ", ians);
      printf(indat[ians-1].cflag ? "ON\n" : "OFF\n");
      if (!indat[ians-1].cflag) {
        printf("\n Associated MINUIT parameters should be FIXed \n");
      }
    }
  }
  break;

case 25:
  printf("currently using calculation mode %d\n", xcalc);
  printf("Switch to? >"); scanf("%d", &xcalc);
  break;

case 26:        /* select/change parameter interpretation */
  answer[0] = '0';
  while ((tolower(answer[0])!='p') && (tolower(answer[0])!='c')) {
    printf("Background amplitudes - [p]olar or [c]artesian? ");
    scanf("%s",answer);
  }
  parmform = tolower(answer[0]);

  answer[0] = '0';
  while ((tolower(answer[0])!='p') && (tolower(answer[0])!='t')) {
    printf("\nParameter one - [p]artial or [t]otal width? ");
    scanf("%s",answer);
  }
  widflag = tolower(answer[0]);
  break;

case 27:        /* multiply data by arbitrary number */
  printf("number to multiply data by? > ");
  scanf("%lf", &muldb);
  printf("All data being multiplied by %lf\n",muldb);
  for (i = 0; i < data.sets; i++) {
    for (j = 0; j < data.pts; j++) {
      eldat[i].xsc[j] *= muldb;
      eldat[i].std[j] *= muldb;
      eldat[i].var[j] = eldat[i].std[j] * eldat[i].std[j];
      eldat[i].xov[j] = eldat[i].xsc[j]/eldat[i].var[j];
    }
  }
  if (inflag) {
    for (i = 0; i < in.sets; i++) {
      for (j = 0; j < in.pts; j++) {
        indat[i].xsc[j] *= muldb;
        indat[i].std[j] *= muldb;
        indat[i].var[j] = indat[i].std[j] * indat[i].std[j];
        indat[i].xov[j] = indat[i].xsc[j]/indat[i].var[j];
      }
    }
  }
  break;

case 28:    /* rescale plots (e.g. after truncation) */
  initplot();
  break;

default:
  printf("fcn: unexpected flag value = %d\n",*flag);
  printf("flags: 3  -- wrap\n");
  printf("       6  -- toggle floating normalization\n");
  printf("       7  -- change parameter value\n");
  printf("       8  -- toggle wrap destination\n");
  printf("       9  -- plot data + fit\n");
  printf("      10 -- schedule plotting\n");
  printf("      11 -- change debug level\n");
  printf("      12 -- calculate and shift resonance energy\n");
  printf("      13 -- input new resolution function \n");
  printf("      14 -- print chi-squared per degree of freedom \n");
  printf("      15 -- clear output file \n");
  printf("      16 -- write parameter file \n");
  printf("      17 -- truncate wings of data \n");
  printf("      18 -- write 'fake' data \n");
  printf("      19 -- test access to MINUIT arrays \n");
  printf("      20 -- calculate background slopes \n");
  printf("      21 -- select type of non-resonant background \n");
  printf("      22 -- list and/or alter background slopes \n");
  printf("      23 -- list and/or alter background intercepts \n");
  printf("      24 -- turn calculations on and off \n");
  printf("      25 -- change calculation mode \n");
```

```
    printf("     26 -- select/change parameter interpretation \n");
    printf("     27 -- multiply data by arbitrary number \n");
    printf("     28 -- rescale plots \n");
    break;
}

            /* Plot data and fits */

if ((CallsPerPlot) || (ForcePlot)){
  Calls++;
  if ((Calls > CallsPerPlot) || (ForcePlot == 1)) {
    Calls = 0;
    putchar('\037'); putchar('\033'); putchar('\014'); /* clear screen */
    if (fittype != 'a') {
      switch (data.sets) {
      case 6:
        plot(&eldat[5].xsc[0], &eldat[5].std[0], &elfit[5].xsc[0],
          255, 511, 259, 519, smin[5], smax[5], eldat[5].normsig);
      case 5:
        plot(&eldat[4].xsc[0], &eldat[4].std[0], &elfit[4].xsc[0],
          0, 255, 259, 519, smin[4], smax[4], eldat[4].normsig);
      case 4:
        plot(&eldat[3].xsc[0], &eldat[3].std[0], &elfit[3].xsc[0],
          767, 1023, 519, 779, smin[3], smax[3], eldat[3].normsig);
      case 3:
        plot(&eldat[2].xsc[0], &eldat[2].std[0], &elfit[2].xsc[0],
          511, 767, 519, 779, smin[2], smax[2], eldat[2].normsig);
      case 2:
        plot(&eldat[1].xsc[0], &eldat[1].std[0], &elfit[1].xsc[0],
          255, 511, 519, 779, smin[1], smax[1], eldat[1].normsig);
      default:
        plot(&eldat[0].xsc[0], &eldat[0].std[0], &elfit[0].xsc[0],
          0, 255, 519, 779, smin[0], smax[0], eldat[0].normsig);
        nsets = data.sets;
        break;
      }
      if (inflag) {
        switch (in.sets) {
        case 6:
          plot(&indat[5].xsc[0], &indat[5].std[0], &infit[5].xsc[0],
            767, 1023, 0, 259, imin[5], imax[5], indat[5].normsig);
        case 5:
          plot(&indat[4].xsc[0], &indat[4].std[0], &infit[4].xsc[0],
            511, 767, 0, 259, imin[4], imax[4], indat[4].normsig);
        case 4:
          plot(&indat[3].xsc[0], &indat[3].std[0], &infit[3].xsc[0],
            255, 511, 0, 259, imin[3], imax[3], indat[3].normsig);
        case 3:
          plot(&indat[2].xsc[0], &indat[2].std[0], &infit[2].xsc[0],
            0, 255, 0, 259, imin[2], imax[2], indat[2].normsig);
        case 2:
          plot(&indat[1].xsc[0], &indat[1].std[0], &infit[1].xsc[0],
            767, 1023, 259, 519, imin[1], imax[1], indat[1].normsig);
        default:
          plot(&indat[0].xsc[0], &indat[0].std[0], &infit[0].xsc[0],
            511, 767, 259, 519, imin[0], imax[0], indat[0].normsig);
          nsets += in.sets;
          break;
        }
      }
      if (nsets < 12) plotres(yint,nres,767,1023,0,259,rmin,rmax);
    }

    if (fittype != 's') {       /* never used for unpolarized data */
      switch (data.sets) {
      case 6:
      case 5:
      case 4:
        plot(&eldat[3].ay[0], &eldat[3].eay[0], &elfit[3].ay[0],
          767, 1023, 0, 390, amin[3], amax[3], eldat[3].normay);
      case 3:
        plot(&eldat[2].ay[0], &eldat[2].eay[0], &elfit[2].ay[0],
          511, 767, 0, 390, amin[2], amax[2], eldat[2].normay);
      case 2:
        plot(&eldat[1].ay[0], &eldat[1].eay[0], &elfit[1].ay[0],
          255, 511, 0, 390, amin[1], amax[1], eldat[1].normay);
      default:
        plot(&eldat[0].ay[0], &eldat[0].eay[0], &elfit[0].ay[0],
          0, 255, 0, 390, amin[0], amax[0], eldat[0].normay);
        break;
      }
    }
    if (ForcePlot == 1) {
      ForcePlot = 0;
```

```
        ans = 0;
        while (!(ans = getchar()));
        putchar('\030');                /* get vt100 screen back */
      }
    }
  }                                     /* endif (CallsPerPlot) */
}
```

```
/*****************************************************************
*   getbw.c  -  Calculate info for setup of Breit-Wigner function to
*               be computed by the function breit. Called initially by
*               initdat(), but may be recalled if needed.
*/

#include "resfit.h"
struct INFO bw;
extern int ninterp;
extern struct INFO data, resf;

void getbw()
{
  register int i;
  int bwptsck;

  if (debug>1) fprintf(fpdeb,"\n\t\t========== getbw begin ==========\n");
  if (debug) fprintf(fpdeb," getbw: \n");

        /* Calculate INFO for breit-wigner functions */
       /* previously defined bw.dx = data.dx/ninterp */
   /* if resf.xmax > resf.ext then should use resf.xmax in next two eqns. */
   /* since there will be an energy SHIFT as well as broadening caused */
   /* by the convolution                                               */
  bw.xmin = data.xmin - resf.ext;   /* starting energy for breit arrays */
  bw.xmax = data.xmax + resf.ext;   /* final energy for breit arrays    */
  bw.ext = bw.xmax - bw.xmin;       /* energy extent of breit-wigner    */

  bw.pts = 1 + NINT(bw.ext / bw.dx);
  bwptsck = (data.pts - 1)*ninterp + 2*resf.pts - 1; /* check # points */
  if (bw.pts != bwptsck) printf("\n***** problem w/ internal arrays \n");

  for (i = 0; i < bw.pts; i++) {
    bw.x[i] = bw.xmin + (i * bw.dx);     /* these do not change */
  }
  if (debug>1) fprintf(fpdeb,"\n\t\t========== getbw end ==========\n");
}
```

```
/*************************************************************************
*  getdat.c  - Read a text file containing experimental data into global
*              arrays. Set some global variables indicating what was read.
*/

#include "resfit.h"

void getdat(pinf,pdat)
struct INFO *pinf;              /* pointer to information structure */
struct XFCN pdat[];             /* pointer to array of data structures */
{
  REAL lab2cm(), enerMeV;
  register int i,j;
  char cbuf[MAXLINE];
  FILE *fp, *fopen();

  if (debug>1) fprintf(fpdeb,"\n\t\t========== getdat begin ==========\n");
  if (debug) fprintf(fpdeb," getdat: \n");

  IOPENR(fp,pinf->file);
 /*
  printf("Name of DATA File >"); scanf("%s", pinf->file);
  fp = fopen(pinf->file, "r");
  if (fp == NULL) {
    printf("***** unable to open >%s<\n", pinf->file);
    if (debug>1) fprintf(fpdeb," fopen returned >%d<\n",fp);
    exit(1);
  }
 */

      /* get info on data and data file */
  fgets(cbuf, MAXLINE, fp);
  sscanf(cbuf, "%d %d %lf %lf", &pinf->sets, &pinf->pts, &enerMeV, &pinf->dx);
  pinf->xmin = enerMeV * EUNIT;                 /* everything in eV */
  pinf->xmax = pinf->xmin + ((pinf->pts - 1) * pinf->dx);
  pinf->ext = pinf->xmax - pinf->xmin;

  if (debug>1) {
    fprintf(fpdeb," # dets=%d, # pts=%d\n", pinf->sets, pinf->pts);
    fprintf(fpdeb," min. x = %lf, dx = %lf\n", pinf->xmin, pinf->dx);
  }
      /* store actual values for x array */
  for (i = 0; i < pinf->pts; i++) {
    pinf->x[i] = pinf->xmin + (i * pinf->dx);
  }

      /* get kinematic and experimental info */
  fgets(cbuf, MAXLINE, fp);       /* masses and Q-value */
  sscanf(cbuf, "%lf %lf %lf %lf %lf",
       &pinf->mpi, &pinf->mti, &pinf->mpf, &pinf->mtf, &pinf->q);
  if (debug>1) fprintf(fpdeb," mpi mti mpf mtf %lf %lf %lf %lf \n",
            pinf->mpi, pinf->mti, pinf->mpf, pinf->mtf);

  fgets(cbuf, MAXLINE, fp);       /* 2*spins of system */
  sscanf(cbuf, "%d %d %d %d",
        &pinf->iap, &pinf->iat, &pinf->ibp, &pinf->ibt);
  if (debug>1) fprintf(fpdeb," 2 * spins - %d %d %d %d\n",
                pinf->iap, pinf->iat, pinf->ibp, pinf->ibt);

      /* get actual data */
  for (i = 0; i < pinf->sets; i++) {
    fgets(cbuf, MAXLINE, fp);
    sscanf(cbuf, "%lf", &pdat[i].angle);
    if (debug>1) fprintf(fpdeb," lab angle(deg) = %f\n",pdat[i].angle);
    pdat[i].angle =
      lab2cm(pdat[i].angle*DEG2RAD,pinf->mpi,pinf->mti,
          pinf->mpf,pinf->mtf,enerMeV,pinf->q);
    if (debug>1) fprintf(fpdeb," cm angle(rad) = %f\n",pdat[i].angle);
    for (j = 0; j < pinf->pts; j++) {
      fgets(cbuf, MAXLINE, fp);
      sscanf(cbuf, "%lf %lf ", &pdat[i].xsc[j], &pdat[i].std[j]);
      if (pdat[i].std[j] == 0.) {
        printf("** getdat: read point whose error is zero [%d,%d]\n",i,j);
        pdat[i].std[j] = 1.0;
      }
    }
  }
  (void) fclose(fp);
  if (debug>1) fprintf(fpdeb,"\n\t\t========== getdat end ==========\n");
}
```

```
/*************************************************************************
*   getres.c  -  Read in resolution function. Also gather information
*                about resolution function to fill INFO structure.
*                Also, interpolate resolution function to fine energy
*                steps set by the user in initdat. Interpolate such
*                that the energy range remains the same as original
*                input.
*/

#include "resfit.h"
struct INFO resf;

void getres(dx)
double dx;
{
  void norm(), dcspin_();
  int sdatin();
  double moment();
  register int i;
  int npts = 0;
  double x[ALEN], y[ALEN];

  if (debug>1) fprintf(fpdeb,"\n\t\t========== getres begin ==========\n");
  if (debug) fprintf(fpdeb," getres: \n");

  while (!npts) {
   printf("Filename for resolution function (keV steps): ");
   scanf("%s", resf.file);
   npts = sdatin(resf.file, x, y, NULL);        /* read resol. func.  */
  }

  resf.xmin = PDBIG; resf.xmax = -PDBIG;   /* find min and max x-values */
  for (i = 0; i < npts; i++) {
    x[i] *= RESUNITS;                    /* change to eV */
    if (x[i] < resf.xmin) resf.xmin = x[i];
    if (x[i] > resf.xmax) resf.xmax = x[i];
  }
  /* check suitability of resolution function and warn user if not right */
  if ((resf.xmin != x[0]) || (resf.xmax != x[npts-1]))
    printf("***** Caution! resolution function not increasing in x \n");

  resf.mean = moment(1,0.0,npts,x,y,NULL);        /* mean value  */

  resf.pts = 1 + (int)((resf.xmax - resf.xmin)/dx);    /* new # of points */
  if (resf.pts > ALEN) printf("***** CAUTION resf array > %d \n",ALEN);

      /* Calculate new x array in desired step size (dx) */
  for (i = 0; i < resf.pts; i++) {
    resf.x[i] = resf.xmin + i*dx;
    if (resf.x[i] > resf.xmax) printf("**** point outside x interval \n");
  }
  resf.xmax = resf.x[i-1];        /* new max value for even # of points */

              /* Interpolate to new x values */
  dcspin_(x, y, &npts, resf.x, resf.y, &resf.pts);
  norm(resf.y, resf.y, resf.pts);

  resf.dx = dx;                          /* new interval size (eV)   */
  resf.ext = resf.xmax - resf.xmin;      /* energy width of resf (ev) */

  printf("\n\t Resolution function has %d points \n", resf.pts);
  printf("\t in the x range (%lf, %lf) \n", resf.xmin, resf.xmax);
  printf("\t with a mean value of x = %lf .\n",resf.mean);
  if (debug > 2) {
    fprintf(fpdeb," \n\t x-values \t y-values \n");
    for (i = 0; i < resf.pts; i++) {
      fprintf(fpdeb,"\t %lf \t %lf \n", resf.x[i], resf.y[i]);
    }
  }

  if (debug>1) fprintf(fpdeb,"\n\t\t========== getres end ==========\n");
}
```

```
/*********************************************************************
*   horshift.c  - Calculate number of channels to shift fit to
*                 minimize chi squared. Requires fit and the data
*                 to be known at every point.
*/

#include "resfit.h"
extern struct INFO data;

REAL horshift(fit,dat,esqr)
REAL *fit, *dat, *esqr;
{
  register int i;
  REAL b[MAXPTS], c[MAXPTS], *quad() ;
  REAL const, linco, sqrco, *esht;

  if (debug>1) fprintf(fpdeb,"\n\t\t========== horshift begin ==========\n");
  if (debug) fprintf(fpdeb," horshift: \n");
  const = 0.0; linco = 0.0; sqrco = 0.0;

  for (i = 1; i < data.pts-1; i++) {
    b[i] = (fit[i+1] - fit[i-1])/2.;
    c[i] = ((fit[i+1] + fit[i-1])/2.)-fit[i];
    if (debug>2) fprintf(fpdeb,"values of fits %lf, %lf, %lf\n",
                 fit[i-1],fit[i],fit[i+1]);

    const += b[i]*(fit[i]-dat[i])/esqr[i];
    linco += (b[i]*b[i] + 2.*c[i]*(fit[i]-dat[i]))/esqr[i];
    sqrco += b[i]*c[i]/esqr[i];
  }
  if (debug>1)
    fprintf(fpdeb,"coeffs of quad.: %lf, %lf, %lf\n",sqrco,linco,const);

  esht = quad(3.0*sqrco,linco,const);  /*point to roots of quad. eqn.*/
  if (*esht == D_NOVALUE) return(0);
  if (debug>1) fprintf(fpdeb," highest (mag.) root of eqn = %lf\n",*(esht+1));
  if (debug>1) fprintf(fpdeb," lowest (magnitude) root of eqn = %lf\n",*esht);
  if (debug>1) fprintf(fpdeb,"\n\t\t========== horshift end ==========\n");

  return(*esht);
}
```

```
C*********************************************************************
C*   incstk.f -- FORTRAN subroutine to increase PORT stack size.
C*          Must be called before any other PORT routine which
C*          uses dynamic stack allocation. (e.g. dcspin).
C*          Must be linked with port library (-lport3)
C*
      subroutine incstk()

      common /CSTAK/DSTAK(10000)
      double precision DSTAK

C           Call PORT routine to initialize stack
C           (see "Framework" section of PORT manual)
      call ISTKIN(10000,4)

      return
      end
```

```
/*****************************************************************
*   initdat.c - Talk to user.  Find out how minuit has been set up,
*               where data is to be found, etc.  Call functions getdat()
*               and sdatin() to read in data and beam resolution function.
*               Set global variables controlling the rest of resfit.
*               This version for Mg(a,a).
*/

#include "resfit.h"
int nparm, ninterp, trunlhs = 0, inflag = 0, parmform, widflag;
REAL nsavgsig, nsavgay, gsqr;
REAL oldp[MAXPARM];
REAL qa[MAXDETS], qasqr[MAXDETS], qb[MAXDETS], qbsqr[MAXDETS];
char fittype = 's', noquad = 'n';
struct XFCN eldat[MAXDETS], indat[MAXDETS];
struct INFO data, in;
extern struct INFO resf, bw;

void initdat(p)
double *p;                      /* array of parameter values */
{
  void getdat(), getres();          /* read in data, resolution function */
  void getbw();                     /* & calculate values for Breit-Wigner */
  void smooth();                    /* smooth wings of data */
  void incstk_();           /* FORTRAN routine to increase PORT stack size */
  void backsl();            /* compute & return slope of non-res background*/
  int getnext_();               /* get number of external parameters */
  double csl2cm(), moment();
  REAL plm();
  int i, j, lalph, lbeta;
  REAL jres, ecm, multfac, sumfac;
  char ans[2];
  double mu, k;       /*incident channel reduced mass and relative momentum*/

  if (debug>1) fprintf(fpdeb,"\n\t\t========== initdat begin ==========\n");
  if (debug) fprintf(fpdeb," initdat:\n");

  ans[0] = '0';
  while ((tolower(ans[0]) != 'p') && (tolower(ans[0]) != 'c')) {
    printf("Are background amplitudes in polar (magnitude and phase)\n");
    printf("or cartesian (real and imaginary) form? ['p' or 'c'] ");
    scanf("%s",ans);
  }
  parmform = tolower(ans[0]);

  ans[0] = '0';
  while ((tolower(ans[0]) != 'p') && (tolower(ans[0]) != 't')) {
    printf("\nIs parameter one to be treated as the partial width for\n");
    printf("the outgoing channel or the total width ['p' or 't']? ");
    scanf("%s",ans);
  }
  widflag = tolower(ans[0]);

  incstk_();       /* FORTRAN subroutine to set PORT stack size (to 10000) */

  printf("\n For elastic data: ");
  getdat(&data,eldat);  /* read elastic data file; fill INFO data structure */

  if (((int)data.mpi == 4) && ((int)data.mpf == 4)) data.type = 1;
  if (((int)data.mpi == 1) && ((int)data.mpf == 1)) data.type = 2;
  if (((int)data.mpi == 1) && ((int)data.mpf == 4)) data.type = 3;
  printf("                    data type = %d\n", data.type);

  printf("Combined fit with inelastic data? "); scanf("%s",ans);
  if (tolower(ans[0]) == 'y') {
    inflag = 1;
    printf("\n For inelastic data: ");
    getdat(&in,indat);
    /* check that x-values for elastic and inelastic should be the same */
    if ((data.xmin != in.xmin) || (data.xmax != in.xmax) ||
       (data.dx != in.dx)) printf("**** PROBLEM with inelastic data ****\n");
    if (((int)in.mpi == 4) && ((int)in.mpf == 4)) in.type = 1;
    if (((int)in.mpi == 1) && ((int)in.mpf == 1)) in.type = 2;
    if (((int)in.mpi == 1) && ((int)in.mpf == 4)) in.type = 3;
    printf("                    data type = %d\n", in.type);
    if (in.type != 3) {
      printf("Wrong data type for (p,a) data. Continue? (y or n) ");
      scanf("%s",ans);
      if (tolower(ans[0]) == 'n') exit(1);
    }
  }

    /* smooth data on wings if desired; user is prompted for range */
/* printf(" Smooth data on wings? "); scanf("%s",ans);
```

```
if (tolower(ans[0]) == 'y') {
  smooth(&data,eldat);
  if (inflag) smooth(&in,indat);
 }
*/

      /* ENERGY SCALE FOR INTERNAL CONVOLUTION ARRAYS */
printf(" Data energy steps are %lf eV \n", data.dx);
printf(" --- divide these into how many intervals? > ");
scanf("%d",&ninterp);
bw.dx = data.dx / ninterp;          /* energy step for internal arrays */

 /* Get resolution function in fine energy steps & fill INFO resf struct */
if (p[RESP] != 0.0) {
  getres(bw.dx);
  if (debug>1) fprintf(fpdeb,"initdat: resf read\n");
}
else {
  printf(" **** Resolution function zero width - \n");
  printf("\t assume Lorentzian approximation \n");
  resf.xmin = 0.0; resf.xmax = 0.0;
  resf.pts = 0 ; resf.ext = 0.0;
  strcpy(resf.file, "Lorentz approx.");
}

/* printf("Total number of (fixed + variable) parms from minuit > ");
  scanf("%d",&nparm); */

for (i = 0; i < nparm; i++) oldp[i] = D_NOVALUE;

/* printf("J of resonance, L input channel, L exit channel > ");
scanf("%lf %d %d", &jres, &lalph, &lbeta); */
jres = 0.0; lalph = 0; lbeta = 0;

/* printf("Fit to sig, ay, or both < s | a | b > "); scanf("%s",ans);
fittype = tolower(ans[0]);
fittype = 's'; */

/* printf("ignore quadratic term for energy dependent background? > ");
scanf("%s",ans);
noquad = tolower(ans[0]);
noquad = 'n'; */
```

```
/* printf("Use total chi squared [as opposed to per degree of freedom]");
scanf("%s",ans); totchi = tolower(ans[0]); */

nsavgsig = 0.0; nsavgay = 0.0;      /* # channels for auto energy shift */

for (i = 0; i < data.sets; i++) {
  eldat[i].normsig = 1.0;   /* different norms allowed for each detector */
  eldat[i].normay = 1.0;
  eldat[i].bslp = 0.0;      /* init slopes of background */
  eldat[i].bint = D_NOVALUE;  /* y intercept            */
  eldat[i].btype = 'l';     /* default background is linear */
  eldat[i].cflag = 1;       /* all calcs initially on */
}
if (inflag) {
 for (i = 0; i < in.sets; i++) {
   indat[i].normsig = 1.0;  /* different norms allowed for each detector */
   indat[i].normay = 1.0;
   indat[i].bslp = 0.0;     /* init slopes of background */
   indat[i].bint = D_NOVALUE;  /* y intercept            */
   indat[i].btype = 'l';    /* default background is linear */
   indat[i].cflag = 1;      /* all calcs initially on */
 }
}

getbw();                /* Calculate INFO for breit-wigner functions */

if (debug>1){ fprintf(fpdeb,"\ndata.xmin = %lf \n",data.xmin);
        fprintf(fpdeb,"data.xmax = %lf \n",data.xmax);
        fprintf(fpdeb,"data.pts = %d \n",data.pts);
        fprintf(fpdeb,"data.ext = %lf \n",data.ext);
        fprintf(fpdeb,"data.dx = %lf \n",data.dx);
        fprintf(fpdeb,"resf.xmin = %lf \n",resf.xmin);
        fprintf(fpdeb,"resf.xmax = %lf \n",resf.xmax);
        fprintf(fpdeb,"resf.pts = %d \n",resf.pts);
        fprintf(fpdeb,"resf.ext = %lf \n",resf.ext);
        fprintf(fpdeb,"bw.xmin = %lf \n",bw.xmin);
        fprintf(fpdeb,"bw.xmax = %lf \n",bw.xmax);
        fprintf(fpdeb,"bw.pts = %d \n",bw.pts);
        fprintf(fpdeb,"bw.ext = %lf \n",bw.ext);
        fprintf(fpdeb,"bw.dx = %lf \n",bw.dx);
        fprintf(fpdeb,"ninterp = %d \n",ninterp);
```

```
        fprintf(fpdeb,"data.sets = %d \n",data.sets);
        if (inflag) fprintf(fpdeb,"in.sets = %d \n",in.sets);
}

      /* Quantities for calculating "geometric" factors */
mu = (data.mti * data.mpi)/(data.mti + data.mpi);    /* reduced mass    */
       /* center of mass incident particle energy */
ecm = (data.xmin/EUNIT) * data.mti / (data.mti + data.mpi);
k = sqrt(2.0 * ecm * mu) / PLANCK;    /* apx. const. in our ener range   */


      /* ANGLE DEPENDENT FACTORS FOR GENERATED DATA */

       /* average over initial spins */
sumfac = 1.0;   /* # of identical terms in sum over spin projections */
data.avin = sumfac/(double)((data.iap + 1)*(data.iat + 1));
in.avin = sumfac/(double)((in.iap + 1)*(in.iat + 1));
data.gfac = 1/(2.0*k);                  /* used later in xsex */
in.gfac = 1/(2.0*k);                  /* used later in xsex */
gsqr = 1/(4.0*k*k);

for (i = 0; i < data.sets; i++) {
    /* convert values of cross section (solid angle) to com */
  multfac =
    csl2cm(data.mpi,data.mti,data.mpf,data.mtf,ecm,data.q,eldat[i].angle);
  for (j = 0; j < data.pts; j++) {
    eldat[i].xsc[j] *= multfac;
    eldat[i].std[j] *= multfac;
        /* Calculate ERROR_SQUARED, DATA_OVER_ERROR_SQUARED */
    eldat[i].var[j] = eldat[i].std[j] * eldat[i].std[j];
    eldat[i].xov[j] = eldat[i].xsc[j] / eldat[i].var[j];
    /* eldat[i].ayesqr[j] = eldat[i].eay[j] * eldat[i].eay[j];
    if (eldat[i].ayesqr[j] == 0.) eldat[i].ayesqr[j] = 1.;
    eldat[i].aydee[j] = eldat[i].ay[j] / eldat[i].ayesqr[j]; */
  }

  qa[i] =  -1.0/(2.0*k);
  qasqr[i] = qa[i] * qa[i];
  qb[i] =  0.0;          /* the "spin-flip" term is zero */
  qbsqr[i] = qb[i] * qb[i];
}
```

```
if (inflag) {
 for (i = 0; i < in.sets; i++) {
    /* convert values of cross section (solid angle) to com */
   multfac = csl2cm(in.mpi,in.mti,in.mpf,in.mtf,ecm,in.q,indat[i].angle);
   for (j = 0; j < in.pts; j++) {
    indat[i].xsc[j] *= multfac;
    indat[i].std[j] *= multfac;
      /* Calculate ERROR_SQUARED, DATA_OVER_ERROR_SQUARED */
    indat[i].var[j] = indat[i].std[j] * indat[i].std[j];
    indat[i].xov[j] = indat[i].xsc[j] / indat[i].var[j];
    /* indat[i].ayesqr[j] = indat[i].eay[j] * indat[i].eay[j];
    if (indat[i].ayesqr[j] == 0.) indat[i].ayesqr[j] = 1.;
    indat[i].aydee[j] = indat[i].ay[j] / indat[i].ayesqr[j]; */
   }

   qa[i] =  -1/(2*k);
   qasqr[i] = qa[i] * qa[i];
   qb[i] =  0.0;        /* the "spin-flip" term is zero */
   qbsqr[i] = qb[i] * qb[i];
 }
}

  /* Calculate slopes of background cross sections if desired */
printf("\nCalculate background slopes for elastic data? "); scanf("%s",ans);
if (tolower(ans[0]) == 'y') backsl(&data,eldat);
if (inflag) {
 printf("Calculate background slopes for inelastic data? ");scanf("%s",ans);
 if (tolower(ans[0]) == 'y') backsl(&in,indat);
}

nparm = getnext_();
printf("\n\t total number of external parameters = %d \n", nparm);

if (debug > 2) {
  fprintf(fpdeb,"initdat: det leg1 leg2 qa qb\n");
  for (i = 0; i < data.sets; i++)
    fprintf(fpdeb,"\t%d %lf %lf %lf %lf\n",i,plm(lbeta,-1+1,eldat[i].angle),
        plm(lbeta,-1+1,eldat[i].angle),qa[i],qb[i]);
}
if (debug>1) fprintf(fpdeb,"\n\t\t========== initdat end ==========\n");
}
```

```
/*************************************************************************
*     interp.c - Uses PORT subroutine to fit to cubic spline and
*                interpolate. Files must be monotonically increasing in x.
*                Input and output have the approximately the
*                same energy ranges but not
*                necessarily the same number of points.
*/

#include "resfit.h"
extern struct INFO data;

int interp(x,y,n,xint,yint,dxi)
int n;
double *x, *y, *xint, *yint, dxi;
{
  void dcspin_();       /* PORT spline fitting subroutine (FORTRAN) */
  register int i;
  int ni;               /* for passing to FORTRAN routines */
  int linup;

  if (debug>1) fprintf(fpdeb,"\n\t\t========== interp begin ==========\n");
  if (debug) fprintf(fpdeb," interp: \n");

  if (debug>1) fprintf(fpdeb,"Input number of points is %d \n",n);
  if (debug>1) fprintf(fpdeb,"\t in the x-range (%lf,%lf) \n", x[0], x[n-1]);

    /* make sure that the x-values line up with the data points */
  linup = (int)(ABS(x[0] / data.dx));   /* truncate for int part */
  xint[0] = SIGN(x[0]) * linup * data.dx;
  ni = 1 + (int)((x[n-1] - xint[0])/dxi);  /* use approx. same energy range */
  if (ni > ALEN) printf("***** CAUTION: resf array > %d! \n",ALEN);

    /* calculate new x arrays with new step size for interpolation */
  for (i = 0 ; i < ni; i++) {
     xint[i] = xint[0] + (i * dxi);
     if (xint[i] > x[n-1]) printf("**** point outside x interval \n");
  }

  if (debug>1) {
     fprintf(fpdeb,"Final number of points is %d \n",ni);
     fprintf(fpdeb,"\t in the x-range (%lf,%lf) \n", xint[0], xint[ni-1]);
  }

        /* dcspin uses 6*n locations in the PORT stack */
  dcspin_(x,y,&n,xint,yint,&ni);    /* interpolate y-values for dist */

  if (debug>1) fprintf(fpdeb,"\n\t\t========== interp end ==========\n");

  return(ni);
}
```

```
/*****************************************************************
*  ipow.c  - Function to return integer to integer power.
*          Mainly used to evaluate things that lib function
*          pow(a,b) doesn't like; e.g. (-1) to an integer power.
*/

#include "/usr/ref/inc/stdinc.h"

int ipow(i,j)
int i, j;
{
      int k;
      for (k=1; j>0; j--) k=k*i;
      return(k);
}
```

```
/*****************************************************************
*  lab2cm.c  - Convert lab angle to center of mass angle.
*          Lab angle in radians; returned value in radians;
*          masses in amu's.
*/

#include "/usr/ref/inc/stdinc.h"

REAL lab2cm(labang, mpi, mti, mpf, mtf, elab, q)
REAL labang, mpi, mti, mpf, mtf, elab, q;
{
  REAL cmang, ecm, x;

  if (debug) fprintf(fpdeb," lab2cm: \n");
  if (debug>1) fprintf(fpdeb,"lab2cm: lab angle(rad) = %f\n",labang);

  ecm = elab * mti/(mpi+mti);
  x = sqrt((mpi*mpf/(mti*mtf))*ecm/(ecm+q));
  cmang = labang + asin(x * sin(labang));

  if (debug>1) fprintf(fpdeb,"lab2cm: cm angle(rad) = %f\n",cmang);
  return(cmang);
}
```

```
/*****************************************************************
*  lorentz.c - Fill arrays with the real and imaginary parts of a
*          Lorentzian function.  Array points are calculated in fine
*          energy steps, suitable for convolution performed elsewhere.
*/

#include "resfit.h"
extern struct INFO bw;

void lorentz(gam,er,lr,li)
REAL gam, er;                   /* total width and resonance energy */
REAL *lr, *li;                  /* pointers to real & imaginary arrays */
{
  register int i;
  REAL g2sqr, ediff, denom;

  if (debug>1) fprintf(fpdeb,"\n\t\t========== lorentz begin ==========\n");
  if (debug) fprintf(fpdeb,"lorentz:\n");

  g2sqr = gam * gam / 4.0;          /* (resonance_width over 2) squared */

   /*     EUNIT is 10e6 (defined in header file resfit.h)          */

  er *= EUNIT;                          /* change to eV */
  ediff = bw.xmin - er;

  for (i = 0; i < bw.pts; i++, ediff += bw.dx) {
    denom = ediff * ediff + g2sqr;
    if (ABS(denom) < PDSMAL) denom = PDSMAL;
    *(lr+i) = ediff / denom;
    *(li+i) = 1.0 / denom;
  }
  if (debug>1) fprintf(fpdeb,"\n\t\t========== lorentz end ==========\n");
}
```

```
/*****************************************************************
*   low.c  -  Function to return the lowest of three integers
*/

#include "/usr/ref/inc/stdinc.h"

low(i, j, k)
int i, j, k;
{
     return((i<=j) ? ((i<=k) ? i : k) : ((j<=k) ? j : k));
}
```

```
/*********************************************************************
*   moment.c -- find moments of distributions
*/

#include "/usr/ref/inc/stdinc.h"

double moment(n,axis,npts,x,y,var)
int n, npts;            /* nth moment, # points in probability function */
double axis, *x, *y, *var;   /* axis, x-value, y-value, variance */
{
  register int i;
  double sum1, sum2, pow();

  if (debug) fprintf(fpdeb," moment: \n");

  if (var == NULL) {                /*theoretical distribution*/
    if (debug>1)
       fprintf(fpdeb,"moment: No errors - theoretical distribution \n");
    for (i=0, sum1=0.0, sum2=0.0 ; i < npts ; i++) {
      sum1 += pow((x[i]-axis),(double)n)*y[i];
      sum2 += y[i];
    }
  }
  else {                        /*data distribution*/
    if (debug>1)
       fprintf(fpdeb,"moment: Errors included - data distribution \n");
     for (i=0, sum1=0.0, sum2=0.0 ; i < npts ; i++) {
       sum1 += pow((x[i]-axis),(double)n)*y[i]/(var[i]);
       sum2 += y[i]/(var[i]);
     }
    }
  return(sum1/sum2);
}
```

```
/*********************************************************************
*   norm.c -- function to provide normalized discrete distributions
*/

#include "/usr/ref/inc/stdinc.h"

void norm(y,ynorm,n)
double *y, *ynorm;    /* arrays of y, and normalized y values */
int n;                /* number of points */
{
  register int i;
  double sum;

  if (debug) fprintf(fpdeb,"norm: \n");

  for (sum = 0.0, i = 0 ; i < n ; i++)  sum += y[i];

  if (debug>1) fprintf(fpdeb,"norm: sum is %lf \n", sum);

  /* produce normalized y-values - note: sensible if ynorm[]=y[] */
  for (i=0 ; i < n ; i++) ynorm[i] = y[i]/sum;
}
```

```
/*************************************************************************
*    normar.c -- function to provide normalized (area = 1) distributions
*/

#include "/usr/ref/inc/stdinc.h"

void normar(x,y,ynorm,n)
double *x, *y, *ynorm;    /* arrays of x, y, and normalized y values */
int n;                    /* number of (x,y) points */
{
  register int i;
  double sum, dx;

  if (debug) fprintf(fpdeb,"normar: \n");

  for (sum=0.0, i=0 ; i < n ; i++)  sum += y[i];

  dx = (ABS(x[n-1]-x[0])) / (n-1);
  sum *= dx;

  if (debug>1) fprintf(fpdeb,"normar: area is %lf\n", sum);

  /* produce normalized y-values - note: sensible if ynorm[]=y[] */
  for (i=0 ; i < n ; i++) ynorm[i] = y[i]/sum;
}
```

```
/*************************************************************************
*   normfit.c  -  Calc normalization to produce best fit to data.
*             Formula is from an analytical expression for the
*             derivative of chi-square with respect to a parameter
*             multiplying THE FIT.
*/

#include "resfit.h"
extern REAL minnorm, maxnorm;
extern struct INFO data;

REAL normfit(fit,dat,ersq)
REAL *fit, *dat, *ersq;          /* generated data, observed data, square
                                    of error of observed data          */
{
  register int i;
  REAL norm, sum1, sum2;

  if (debug>1) fprintf(fpdeb,"\n\t\t========== normfit begin ==========\n");
  if (debug) fprintf(fpdeb," normfit: \n");
  for (i=0, sum1=0., sum2=0.; i < data.pts; i++, fit++, dat++, ersq++) {
    sum1 += (*fit)*(*dat)/(*ersq);
    sum2 += (*fit)*(*fit)/(*ersq);
  }
  norm = ((sum2 > 0.) ? sum1/sum2 : 1.);          /* avoid div by zero */
  norm = max(norm, minnorm);
  norm = min(norm, maxnorm);
  if (debug>1) fprintf(fpdeb,"        norm = %f \n", norm);
  if (debug>1) fprintf(fpdeb,"\n\t\t========== normfit end ==========\n");

  return(norm);
}
```

```
/*************************************************************************
*   plm.c  -  Return value of associated legendre polynomial of cos(theta).
*         Uses recursion formulas from e.g. Arfken (12.87) & (12.88).
*         Indices are integers, theta is real and in radians.
*         WARNING: there is some ambiguity as to the definition of the
*         Legendre polynomials. See Arfken (or similar) for clarification.
*/

#include "/usr/ref/inc/stdinc.h"

REAL plm(L,ML,angle)
int L,ML;
REAL angle;                             /* angle in radians */
{
  REAL fact();
  REAL p[9][9], x, xsqr, coef;
  int n, m;

  if (debug) fprintf(fpdeb," plm: \n");
  if ((L < ML) || (L > 9)) {
    if (debug>1) fprintf(fpdeb," plm: L=%d, ML=%d \n", L, ML);
    printf(" plm: returning error value \n");
    return(D_NOVALUE);
  }

  x = cos(angle);
  xsqr = x * x;

  p[0][0] = 1.0;
  p[1][0] = x;
  p[1][1] = sin(angle);
  p[2][0] = .5 * (3. * xsqr - 1.);
  p[2][1] = 3. * p[1][0] * p[1][1];
  p[2][2] = 3. * (1. - xsqr);

  if (ML < 0) {
    ML = -ML;
    coef = ((ML % 2) ? -1.0 : 1.0);       /* -1 if ML is odd, else +1 */
    coef *= fact(L-ML) / fact(L+ML);
  }
  else
    coef = 1.0;

  if (L < 3) {
    if (debug>1)
      fprintf(fpdeb," plm: L=%d, ML=%d, returning %lf\n",L,ML,p[L][ML]);
    return(coef * p[L][ML]);
  }

  for (n=3; n <= L; n++) {
    p[n][0] = (x*(2*n-1)*p[n-1][0] - (n-1)*p[n-2][0]) / n;
    if (ML != 0)
      p[n][1] = (x*(2*n-1)*p[n-1][1] - n*p[n-2][1]) / (n-1);
  }

  if (ML < 2) {
    if (debug>1)
      fprintf(fpdeb," plm: L=%d, ML=%d, returning %lf\n",L,ML,coef*p[L][ML]);
    return(coef * p[L][ML]);
  }

  for (m=2; m <= ML; m++)
    p[L][m] = ((2*(m-1)*x / sqrt(1-xsqr)) * p[L][m-1]) -
          ((L*(L+1) - (m-1)*(m-2)) * p[L][m-2]);
  if (debug>1)
    fprintf(fpdeb," plm: L=%d, ML=%d, returning %lf\n",L,ML,coef*p[L][ML]);

  return(coef * p[L][ML]);
}
```

```
/*************************************************************************
*  plot.c  -  Collection of plotting routines for excitation functions
*         for plotting on tektronix have (1024,780) pixel resolution.
*         Some possible configurations:
*
*         779 _____________________        779 _____________________
*             |     |     |     |              |         |         |
*             |     |     |     |          519 |---------|---------|
*         390 |-----|-----|-----|              |         |         |
*             |     |     |     |          259 |---------|---------|
*           0 |_____|_____|_____|              |_________|_________|
*             0    341   682   1023            0         511       1023
*
*         779 _____________________        779 _____________________
*             |   |   |   |   |              |   |   |   |   |
*             |   |   |   |   |          519 |---|---|---|---|
*         390 |---|---|---|---|              |   |   |   |   |
*             |   |   |   |   |          259 |---|---|---|---|
*             |___|___|___|___|              |___|___|___|___|
*           0   255 511 767 1023           0   255 511 767 1023
*/

#include "resfit.h"
REAL smin[MAXDETS], smax[MAXDETS], amin[MAXDETS], amax[MAXDETS];
REAL imin[MAXDETS], imax[MAXDETS];
REAL rmin, rmax;
extern int inflag;
extern struct XFCN eldat[], indat[];
extern struct INFO data, in, resf;

/***************** first function - putXY() ******************************/
void putXY(x,y)
int x, y;
{
  unsigned char hx, lx, hy, ly;
  static unsigned char old_hx = 0, old_hy = 0, old_ly = 0;

  hy = 0x20 | y >> 5;
  ly = 0x60 | (y & 0x1f);
  hx = 0x20 | x >> 5;
  lx = 0x40 | (x & 0x1f);

  if (hy != old_hy) putchar(hy);
  if ((hx != old_hx) || (ly != old_ly)) putchar(ly);
  if (hx != old_hx) putchar(hx);
  putchar(lx);

  old_hy = hy; old_ly = ly; old_hx = hx;
}

#define GSTART '\035'

/***************** second function - plot() ******************************/
void plot(d, e, f, xpmin, xpmax, ypmin, ypmax, ymin, ymax, norm)
REAL *d, *e, *f, ymax, ymin, norm;
int xpmin, xpmax, ypmin, ypmax;
{
  void putXY();
  register int i, x, y;
  register double tmp, xscale, yscale;

                /* get scale factors */
  xscale = (xpmax-xpmin)/(data.pts-1);
  for (i=0; i< data.pts; i++) {
    tmp = f[i] * norm;
    if (tmp > ymax) ymax = tmp;
    if (tmp < ymin) ymin = tmp;
  }
  yscale = (ypmax-ypmin)/(ymax-ymin);

                /* PLOT DATA */
  for (i=0; i< data.pts; i++) {
    putchar(GSTART);
    x = xpmin + i * xscale;
    y = ypmin + (int) ((d[i] - e[i] - ymin) * yscale);
    putXY(x,y);
    y += 2 * (int)(e[i] * yscale);
    putXY(x,y);
  }

                /* PLOT FIT */
  putchar(GSTART);
  for (i=0; i< data.pts; i++) {
    x = xpmin + i * xscale;
    y = ypmin + (int) ((f[i] * norm - ymin) * yscale);
    putXY(x,y);
  }
```

```
                /* DRAW BOX */
  putchar(GSTART);
  putXY(xpmin,ypmin); putXY(xpmax,ypmin);
  putXY(xpmax,ypmax); putXY(xpmin,ypmax); putXY(xpmin,ypmin);

  putchar(GSTART); putXY(1,760); putchar('\037');
}

/******************** third function - initplot() ************************/
void initplot()
{
  register int i, j;

  rmin = PDBIG; rmax = -PDBIG;
  for (i = 0; i < resf.pts; i++) {
    rmax = max(rmax, resf.y[i]);
  }
  rmax *= 2.0;
  rmin = -(.01 * rmax);

  for (i = 0; i < data.sets; i++) {
    smin[i] = 1.0e10; smax[i] = -1.0e10;
    amin[i] = 1.0e10; amax[i] = -1.0e10;
    for (j = 0; j < data.pts; j++) {
      smin[i] = min(smin[i], eldat[i].xsc[j] - eldat[i].std[j]);
      smax[i] = max(smax[i], eldat[i].xsc[j] + eldat[i].std[j]);
      amin[i] = min(amin[i], eldat[i].ay[j] - eldat[i].eay[j]);
      amax[i] = max(amax[i], eldat[i].ay[j] + eldat[i].eay[j]);
    }
  }

  if (inflag) {
    for (i = 0; i < in.sets; i++) {
      imin[i] = 1.0e10; imax[i] = -1.0e10;
      for (j = 0; j < in.pts; j++) {
        imin[i] = min(imin[i], indat[i].xsc[j] - indat[i].std[j]);
        imax[i] = max(imax[i], indat[i].xsc[j] + indat[i].std[j]);
      }
    }
  }
}
```

```
/***************** fourth function - plotres() ***************************/
void plotres(y, n, xpmin, xpmax, ypmin, ypmax, ymin, ymax)
REAL *y, ymax, ymin;
int n, xpmin, xpmax, ypmin, ypmax;
{
  void putXY();
  register int i, ix, iy;
  register double xscale, yscale;

                /* get scale factors */
  xscale = (xpmax-xpmin)/(n-1);          /* in pixels */
  yscale = (ypmax-ypmin)/(ymax-ymin);

                /* PLOT FIT */
  putchar(GSTART);
  for (i = 0; i < n; i++) {
    ix = xpmin + i * xscale;
    iy = ypmin + (int) ((y[i] - ymin) * yscale);
    putXY(ix,iy);
  }

                /* DRAW BOX */
  putchar(GSTART);
  putXY(xpmin,ypmin); putXY(xpmax,ypmin);
  putXY(xpmax,ypmax); putXY(xpmin,ypmax); putXY(xpmin,ypmin);

  putchar(GSTART); putXY(1,760); putchar('\037');
}
```

```
/*****************************************************************
*       quad.c  -  Return solutions to a quadratic equation. Arguments
*                  are a,b - coefficients of square and linear terms,
*             and c, constant term. Returns pointer to solutions -
*             lowest value first. In order for the values (things
*             that are pointed to) to be there after function exit,
*             must declare them as static. Usual way is to pass the
*             pointer, but this seeemed like more fun at the time.
*/

#include "/usr/ref/inc/stdinc.h"

REAL *quad(a,b,c)
REAL a, b, c;
{
        static REAL x[2];       /* values remain in memory after exit:
                                   necessary when returning pointers */
        REAL bsqr, foac, temp;

        if (debug) fprintf(fpdeb," quad: \n");
        if (a == 0.0) {
          x[0] = -c/b; x[1] = x[0]; return(x);
        }

        bsqr = b*b; foac = 4.0*a*c;
        if (bsqr < foac) {
          printf("quad: equation has no real solutions\n");
          x[0] = D_NOVALUE; x[1] = x[0]; return(x);
        }

        x[0] = ((-b) + sqrt(bsqr-foac))/(2.*a);
        x[1] = -x[0]-b/a;

        if (ABS(x[1]) < ABS(x[0])) {
          temp = x[1]; x[1] = x[0]; x[0] = temp;
        }
        return(x);
}
```

```
/*****************************************************************
* resfit.h  -  Header for resonance fitting package of functions
*           designed to work with MINUIT
*/

#include "/usr/ref/inc/stdinc.h"

#define MAXDETS 6          /* detectors - can have elastic and inelastic */
#define MAXUSRPTS 200      /* max points in excitation functions         */
#define MAXPTS  (MAXUSRPTS+2) /* points in excitation function + 2       */
#define RAWPTS ALEN        /* points in internal arrays for high res     */

#define PLANCK 20.4455     /* in units of MeV/c, mb^1/2, amu (?)         */
#define EUNIT 1.0E6        /* convert E_Resonance, escale[0] to std units */
#define RESUNITS 1.0E3     /* convert resolution function energies to eV */
#define TOFILE 0
#define TOSTDOUT 1

struct XFCN {
  int cflag;               /* flag to calculate or not                   */
  int btype;               /* type of fit to backgroud:                  */
                           /*      (l)inear or (e)xponential             */
  REAL angle;              /* detector angle                             */
  REAL normsig, normay;    /* arbitrary normalizations                   */
  REAL bslp, bint;         /* slope & intercept of back xsection         */
  REAL xsc[MAXPTS],        /* cross-section                              */
     std[MAXPTS],          /*   err in cross-section (std dev)           */
     var[MAXPTS],          /*   std*std (variance)                       */
     xov[MAXPTS];          /*   xsc/var                                  */
  REAL ay[MAXPTS],         /* analyzing power                            */
     eay[MAXPTS],          /*   err in analyzing power                   */
     ayesqr[MAXPTS],       /*   eay*eay                                  */
     aydee[MAXPTS];        /*   ay/(eay*eay)                             */
};

struct INFO {
  int pts, sets;           /* number of points and data sets             */
  int iap, iat, ibp, ibt;  /* twice spins                                */
  int type;                /* data type: 1=(a,a), 2=(p,p), 3=(p,a)       */
  double gfac;             /* geometric factor; currently 1/(2k)         */
  double avin;             /* factor for average over initial spins      */
  double x[ALEN], y[ALEN]; /* arrays of x and y values                   */
```

```
  double xmin, xmax;          /* minimum and maximum x values        */
  double dx, ext, mean;       /* x interval, x extent, mean value    */
  double mpi, mti, mpf, mtf, q;  /* masses and q-value                */
  char file[30];
};

/*
* Correspondence between array of parameters passed
* by MINUIT and parameters used to construct artificial excitation
* functions.
* In input to MINUIT, parameters must be numbered in the following order
* (but beginning with 1 instead of 0--as expected for FORTRAN.)
*/
    /* resonance parameters */
#define GO     0 /* partial width for exit channel (std_units)        */
#define GAM    0 /* some programs use total width for 1st parameter   */
#define WID    0 /* generic parameter for either of the above         */
#define BRI    1 /* branching ratio for entrance channel (std_units)  */
#define BRO    2 /* branching ratio for exit channel (std_units)      */
#define GIGO   2 /* ratio of entrance to exit partial widths (std_units) */
#define ER     3 /* resonance energy (std_units*EUNIT)                */
#define RESP   4 /* parameter describing resolution function (std_units) */
#define DK     5 /* energy shift for possible second "pygmy" resonance */
#define PK     6 /* probability of second "pygmy" resonance           */

    /* background parms:
     *  prm1  magnitude of amplitude or real part of amplitude
     *  prm2  phase of amplitude or imaginary part of amplitude
     *  prm3  parameter included in background cross section
     */

/* For some experiments, half of the data sets will contain inelastic data */

#define DET0  7     /* start of background parms for data set 1  */
#define DET1  10    /* start of background parms for data set 2  */
#define DET2  13    /* start of background parms for data set 3  */
#define DET3  16    /* start of background parms for data set 4  */
#define DET4  19    /* start of background parms for data set 5  */
#define DET5  22    /* start of background parms for data set 6  */
#define DET6  25    /* start of background parms for data set 7  */
#define DET7  28    /* start of background parms for data set 8  */
#define DET8  31    /* start of background parms for data set 9  */
#define DET9  34    /* start of background parms for data set 10 */
#define DET10 37    /* start of background parms for data set 11 */
#define DET11 40    /* start of background parms for data set 12 */

#define MAXPARM 49  /* present MINUIT limit (I think)            */
```

```
/*************************************************************************
*   sdatin.c -- function to read single data set from a file.
*           Returns number of (x,y) points.
*/

#include "/usr/ref/inc/stdinc.h"

int sdatin(fname,a,b,c)
char *fname;                        /* file name to read from */
double *a, *b, *c;                  /* arrays for data */
{
  register int i, j, k, n;
  char cbuf[MAXLINE];
  FILE *fp, *fopen();

  if (debug) fprintf(fpdeb," sdatin: \n");

  OPENR(fp,fname);                  /* open file for read */
  if (fp == NULL) return(0);

  n = 0;
  while (fgets(cbuf,MAXLINE,fp) != NULL) {    /* read lines until end */
     for (k=0 ; isspace(cbuf[k]) ; k++) ;  /* find 1st character on line */
        /* see if it's a number ( negative or decimal ) */
     if (isdigit(cbuf[k]) || (cbuf[k] == '-') || (cbuf[k] == '.')) {
       j = n++;                     /* point index */
       if (c == NULL) {
         sscanf(cbuf, "%lf %lf", &a[j], &b[j]);
       }
       else {
         c[j] = 0.0;   /* initialize errors in case there are none */
         sscanf(cbuf, "%lf %lf %lf", &a[j], &b[j], &c[j]);
       }
     }
  }
  (void) fclose(fp);

  if (debug > 3) {
    fprintf(fpdeb,"sdatin: data read in.\n ");
    fprintf(fpdeb,"\t a \t b \t c \n");
    if (c == NULL) {
      for (j = 0; j < n; j++) {
        fprintf(fpdeb,"  %lf %lf  - \n",a[j], b[j]);
      }
    }
    else {
      for (j = 0; j < n; j++) {
        fprintf(fpdeb,"  %lf %lf %lf \n", a[j], b[j], c[j]);
      }
    }
  }
  return(n);
}
```

```
/*************************************************************************
*   shifter.c - Calculates shift in resonance energy to minimize chi
*               squared function. Uses analytical expression for the
*               derivative of chi squared in terms of the energy shift.
*               This procedure should not be used while Minuit is trying
*               to fit, since it affects chi squared value directly.
*/

#include "resfit.h"
extern REAL nsavgsig, nsavgay;
extern char fittype;
extern struct XFCN eldat[], elfit[];
extern struct INFO data;

REAL shifter()
{
  register int n;
  REAL horshift(), ns[MAXDETS];
  REAL chidet(), chiang[MAXDETS], sum1, sum2;

  if (debug>1) fprintf(fpdeb,"\n\t\t========== shifter begin ==========\n");
  if (debug) fprintf(fpdeb," shifter: \n");

  for (n=0, sum1=0, sum2=0; n < data.sets; n++) {
     ns[n] = horshift(elfit[n].xsc, eldat[n].xsc, eldat[n].var);
     chiang[n] =
        chidet(eldat[n].normsig,elfit[n].xsc,eldat[n].xsc,eldat[n].var);
     sum1 += ns[n]/chiang[n];
     sum2 += 1.0/chiang[n];
  }
    nsavgsig = sum1/sum2; nsavgay = 0.0;

  if (debug>1) fprintf(fpdeb," shift of energy by %lf channels\n",nsavgsig);
  if (debug>1) fprintf(fpdeb,"\n\t\t========== shifter end ==========\n");

  return (nsavgsig*data.dx/EUNIT);          /* energy shift in MeV */
}
```

```
/*************************************************************************
*   smooth.c - take a running average of points on the wings of the
*              excitation function. Loses one point at each end.
*              Called as an option in initdat.
*/

#include "resfit.h"

smooth(pinf,pdat)
struct INFO *pinf;
struct XFCN pdat[];
{
  register int i, j;
  int nrhs, nlhs;

  if (debug>1) fprintf(fpdeb,"\n\t\t========== smooth begin ==========\n");
  if (debug) fprintf(fpdeb," smooth: ");

  printf("smooth for how many points on left hand (low energy) side? > ");
  scanf("%d", &nlhs);
  printf("\t\t on right hand (high energy) side? > ");
  scanf("%d", &nrhs);

  if ((nlhs > (int)(pinf->pts/2)) || (nlhs > (int)(pinf->pts/2)))
     printf("**** # of smoothed points > half the total energy range!");

  for (i = 0; i < pinf->sets; i++) {
    for (j = 1; j < nlhs; j++) {
      pdat[i].xsc[j-1] =
         0.25*pdat[i].xsc[j-1] + 0.5*pdat[i].xsc[j] + 0.25*pdat[i].xsc[j+1];
      pdat[i].std[j-1] =
        0.25*pdat[i].std[j-1] + 0.5*pdat[i].std[j] + 0.25*pdat[i].std[j+1];
      pdat[i].var[j-1] = pdat[i].std[j-1] * pdat[i].std[j-1];
      pdat[i].xov[j-1] = pdat[i].std[j-1]/pdat[i].var[j-1];
    }
  }

  for (i = 0; i < pinf->sets; i++) {
    for (j = pinf->pts - nrhs - 1; j < pinf->pts - 1; j++) {
      pdat[i].xsc[j-1] =
         0.25*pdat[i].xsc[j-1] + 0.5*pdat[i].xsc[j] + 0.25*pdat[i].xsc[j+1];
      pdat[i].std[j-1] =
```

```
      0.25*pdat[i].std[j-1] + 0.5*pdat[i].std[j] + 0.25*pdat[i].std[j+1];
    pdat[i].var[j-1] = pdat[i].std[j-1] * pdat[i].std[j-1];
    pdat[i].xov[j-1] = pdat[i].std[j-1]/pdat[i].var[j-1];
  }
 }

 pinf->pts = pinf->pts - 2;        /* lose one point at each end */
 pinf->xmin += pinf->dx;
 pinf->xmax -= pinf->dx;
 pinf->ext = pinf->xmax - pinf->xmin;

 for (i = 0; i < pinf->pts; i++) {
   pinf->x[i] = pinf->xmin + i * pinf->dx;
 }

 if (debug>1) fprintf(fpdeb,"\n\t\t========== smooth end ==========\n");
}
```

```
/*********************************************************************
*  tranparm.c - Convert background parms for input into xsex. The flag
*           parmform tells whether the parameters from the parm array
*           are to be interpreted as polar or cartesian form. In
*           either case, parm[2] is simply a real search parameter.
*/

#include "resfit.h"
extern int parmform;

void tranparm(parm, ar, ai, mag, bprm)
REAL parm[], *ar, *ai, *mag, *bprm;
{
 if (debug>1) fprintf(fpdeb,"\n\t\t========== tranparm begin ==========\n");
 if (debug) fprintf(fpdeb," tranparm: \n");

 switch (parmform) {

   case 'p':          /* convert magnitude/phase to real/imaginary */
     if (debug>1) fprintf(fpdeb,"parameter form is %c \n", parmform);
     *ar = parm[0] * cos(parm[1] * DEG2RAD);
     *ai = parm[0] * sin(parm[1] * DEG2RAD);
     *mag = parm[0] * parm[0];
     *bprm = parm[2];
     break;

   case 'c':          /* relabel real/imaginary for input to xsex */
     if (debug>1) fprintf(fpdeb,"parameter form is %c \n", parmform);
     *ar = parm[0];
     *ai = parm[1];
     *mag = (parm[0] * parm[0]) + (parm[1] * parm[1]);
     *bprm = parm[2];
     break;

   default:
     printf("***** Unexpected value of parameter form flag!");
     break;
 }

 if (debug>1) fprintf(fpdeb,"\n\t\t========== tranparm end ==========\n");
}
```

```
/*****************************************************************
*    trun.c -    truncate points on the wings of the excitation function.
*                Called as an option from fcn.
*                Since elastic and inelastic must have same number of
*                data points, the truncation must be the same for both.
*/

#include "resfit.h"
extern int inflag;
extern struct INFO data, in;
extern struct XFCN eldat[], indat[];

int trun()
{
  register int i, j;
  int nrhs, nlhs;

  if (debug>1) fprintf(fpdeb,"\n\t\t========== trun begin ==========\n");
  if (debug) fprintf(fpdeb," trun: ");

  printf("truncate how many points on left hand (low energy) side? > ");
  scanf("%d", &nlhs);
  printf("\t\t on right hand (high energy) side? > ");
  scanf("%d", &nrhs);

  if ((nlhs > (int)(data.pts/2)) || (nrhs > (int)(data.pts/2))) {
    printf("**** # of truncated points > half the total energy range!\n");
  }
  else {
    printf("\n truncating %d points left and %d points right \n",nlhs,nrhs);
  }

  data.pts = data.pts - nlhs - nrhs;
  in.pts = data.pts;

  for (i = 0; i < data.sets; i++) {
    for (j = 0; j < data.pts; j++) {
      eldat[i].xsc[j] = eldat[i].xsc[j+nlhs];
      eldat[i].std[j] = eldat[i].std[j+nlhs];
      eldat[i].var[j] = eldat[i].std[j] * eldat[i].std[j];
      if (debug>2) printf("variance[%d,%d] = %lf\n",i,j,eldat[i].var[j]);
      eldat[i].xov[j] = eldat[i].xsc[j]/eldat[i].var[j];
    }
  }
  if (inflag) {
    for (i = 0; i < in.sets; i++) {
      for (j = 0; j < in.pts; j++) {
        indat[i].xsc[j] = indat[i].xsc[j+nlhs];
        indat[i].std[j] = indat[i].std[j+nlhs];
        indat[i].var[j] = indat[i].std[j] * indat[i].std[j];
        if (debug>2) printf("variance[%d,%d] = %lf\n",i,j,indat[i].var[j]);
        indat[i].xov[j] = indat[i].xsc[j]/indat[i].var[j];
      }
    }
  }

  data.xmin = data.xmin + (nlhs * data.dx);
  data.xmax = data.xmin + (data.pts-1) * data.dx;
  /* also data.xmax = data.xmax - (nrhs * data.dx); */
  data.ext = data.xmax - data.xmin;
  if (inflag) {
    in.xmin = data.xmin;
    in.xmax = data.xmax;
    in.ext = data.ext;
  }

  for (i = 0; i < data.pts; i++) {
    data.x[i] = data.xmin + i * data.dx;
    if (inflag) in.x[i] = data.x[i];
  }

  if (debug>1) fprintf(fpdeb,"\n\t\t========== trun end ==========\n");

  return(nlhs);
}
```

```
/*****************************************************************
*  wrap.c  - Wrap things up:  write a text file suitable for input to
*          plotting program. Include identifying info about this run
*          of resmin.  May be called several times, saving
*          intermediate results.
*/

#include "resfit.h"
extern int WrapTo, ninterp, inflag;
extern struct XFCN eldat[], elfit[], indat[], infit[];
extern struct INFO data, in, resf;

void wrap(nparm,nvp,parm,chisqr)
INTEGER nparm, nvp;       /* total # of parameters and # of variable param. */
REAL parm[], chisqr;
{
  register int i,j;
  double chipdf;
  char *date();
  FILE *fp, *fopen();

  if (debug>1) fprintf(fpdeb,"\n\t\t========== wrap begin ==========\n");
  if (debug) fprintf(fpdeb," wrap: \n");
  if (WrapTo == TOFILE) {             /* can be reset by user in fcn() */
    printf("\t writing to file resmin.out.....\n");
    fp = fopen("resmin.out", "a");
    if (fp == NULL)
      { printf("wrap: can't open resmin.out\n"); exit(1); }
  }
  else {
    fp = stdout;
  }

  fprintf(fp,"\n @@@@@ New File - %d data points   ", data.pts);
  fprintf(fp,"--  %s ", date());
  fprintf(fp,"\n Results using data file(s) - %s ", data.file);
  if (inflag) fprintf(fp," and %s \n", in.file);
  fprintf(fp," with resolution function - %s. \n", resf.file);
  fprintf(fp," >> interpolating %d intervals per data step \n", ninterp);

  chipdf = chisqr/((data.pts * data.sets) - nvp - 1);
  if (inflag) chipdf = chisqr/((data.pts*(data.sets+in.sets)) - nvp - 1);
  fprintf(fp,"* * * total chi-square for fit: %f \n", chisqr);
  fprintf(fp,"* * * chi-square per degree of freedom: %f \n\n", chipdf);
  for (i = 0; i < nparm; i++)
    fprintf(fp,"* * * parm %d = %f \n", i+1, parm[i]);
  fprintf(fp,"\n");

  fprintf(fp," $$$$ for elastic (or only) data $$$$ \n");
  for (i = 0; i < data.sets; i++) {
    fprintf(fp,"##### detector %d, cross-section data and fits \n", i+1);
    fprintf(fp,"======= includes normalization of %lf \n", eldat[i].normsig);
    fprintf(fp,"======= and background slope of %lf \n", eldat[i].bslp);
    fprintf(fp,"\n  Energy \t  Sigma(CoM) \t  Error \t  Fit(CoM)");
    fprintf(fp,"\n   (MeV) \t   (mb/sr)  \t   (CoM) \t   (mb/sr) ");
    fprintf(fp,"\n  ======= \t  ========== \t  ======= \t  ======= \n");
    for (j = 0; j < data.pts; j++) {
      fprintf(fp, "%10.6f \t %7.3f \t %7.4f \t %7.3f \n", data.x[j]/EUNIT,
        eldat[i].xsc[j], eldat[i].std[j], elfit[i].xsc[j]*eldat[i].normsig);
    }
    fprintf(fp,"\n");
  }
  if (inflag) {
    fprintf(fp,"      $$$$ for inelastic data $$$$ \n");
    for (i = 0; i < in.sets; i++) {
      fprintf(fp,"##### detector %d, cross-section data and fits \n", i+1);
      fprintf(fp,"======= includes normalization of %lf \n", indat[i].normsig);
      fprintf(fp,"======= and background slope of %lf \n", indat[i].bslp);
      fprintf(fp,"\n  Energy \t  Sigma(CoM) \t  Error \t  Fit(CoM)");
      fprintf(fp,"\n   (MeV) \t   (mb/sr)  \t   (CoM) \t   (mb/sr) ");
      fprintf(fp,"\n  ======= \t  ========== \t  ======= \t  ======= \n");
      for (j = 0; j < data.pts; j++) {
        fprintf(fp, "%10.6f \t %7.3f \t %7.4f \t %7.3f \n", data.x[j]/EUNIT,
        indat[i].xsc[j], indat[i].std[j], infit[i].xsc[j]*indat[i].normsig);
      }
      fprintf(fp,"\n");
    }
  }

  /* for (i = 0; i < data.pts; i++) {
    fprintf(fp, " %10.6f ", data.x[i]/EUNIT);
    for (j = 0; j < data.sets; j++) {
      fprintf(fp, "\t %7.3f ", elfit[j].xsc[i]);
    }
```

```
    fprintf(fp, "\n ");
  } */

  if (WrapTo == TOFILE) fclose(fp);
  if (debug>1) fprintf(fpdeb,"\n\t\t========== wrap end ==========\n");
  if (defile) {
    fclose(fpdeb);
    fpdeb = stdout;
    debug = 0;
  }
}
```

```
/*****************************************************************************
*  wrtasdat.c  - write a text file containing experimental fit into
*                file suitable to be reinput as data. Can be called
*                as option from fcn.
*/

#include "resfit.h"

void wrtasdat(pinf,pdat,pfit)
struct INFO *pinf;
struct XFCN pdat[], pfit[];
{
  REAL cm2lab(), csl2cm(), enerMeV, ecm, cscm2l, angle;
  register int i,j;
  char filename[30];
  FILE *fp, *fopen();

  if (debug>1) fprintf(fpdeb,"\n\t\t========== wrtasdat begin ==========\n");
  if (debug) fprintf(fpdeb," wrtasdat: \n");

  printf("Name of file for test data > "); scanf("%s", filename);
  fp = fopen(filename, "w");
  if (fp == NULL) {
    printf("***** unable to open >%s<\n", filename);
    if (debug>1) fprintf(fpdeb," fopen returned >%d<\n",fp);
    exit(1);
  }

      /* print info on data and data file */
  enerMeV = pinf->xmin / EUNIT;
  fprintf(fp, "%d %d %lf %lf \n", pinf->sets, pinf->pts, enerMeV, pinf->dx);
  ecm = enerMeV * pinf->mti / (pinf->mti + pinf->mpi);

      /* print kinematic and experimental info */
  fprintf(fp, "%lf %lf %lf %lf %lf \n",
       pinf->mpi, pinf->mti, pinf->mpf, pinf->mtf, pinf->q);

      /* 2*spins of system */
  fprintf(fp, "%d %d %d %d \n", pinf->iap, pinf->iat, pinf->ibp, pinf->ibt);

      /* print actual fit */
  for (i = 0; i < pinf->sets; i++) {
```

```
    angle = RAD2DEG *
cm2lab(pdat[i].angle,pinf->mpi,pinf->mti,pinf->mpf,pinf->mtf,enerMeV,pinf->q);
    if (angle < 0.0) angle += 180.0;
    fprintf(fp, "%lf \n", angle);
    cscm2l = 1.0 /
csl2cm(pinf->mpi,pinf->mti,pinf->mpf,pinf->mtf,ecm,pinf->q,pdat[i].angle);
    for (j = 0; j < pinf->pts; j++) {
      fprintf(fp, "%lf \t", cscm2l * pfit[i].xsc[j]);
      fprintf(fp, "%lf \n", cscm2l * pdat[i].std[j]);
    }
  }
  (void) fclose(fp);
  if (debug>1) fprintf(fpdeb,"\n\t\t ========== wrtasdat end ==========\n");
}
```

```
/*********************************************************************
*   wrtparm.c - Writes parameters in form to be used for input
*               to MINUIT. May be called from FCN.
*/

#include "resfit.h"
extern int nparm;

void wrtparm(p)
double *p;
{
  register int i, j;
  double pran, er, up, lw;
  char filename[30], ans[2];
  FILE *fopen(), *fp;

  if (debug>1) fprintf(fpdeb,"\n\t\t========== wrtparm begin ==========\n");
  if (debug) fprintf(fpdeb," wrtparm: \n");

  printf("What percent (+/-) range for parameters? ");
  scanf("%lf",&pran);
  pran *= 0.01;
  up = 1.0 + pran;
  lw = 1.0 - pran;
  er = 0.1 * pran;

  printf("\n Input filename for parameter file > "); scanf("%s",filename);
  printf(" Writing parameters to file %s ... \n", filename);

  fp = fopen(filename,"a");

  fprintf(fp,"1, wid,%7.2f,%7.2f,%7.2f,%7.2f\n",p[0],er*p[0],lw*p[0],up*p[0]);
  fprintf(fp,"2, bri,%6.3f,%6.3f,%6.3f,%6.3f\n",p[1],er*p[1],lw*p[1],up*p[1]);
  fprintf(fp,"3, bro,%6.3f,%6.3f,%6.3f,%6.3f\n",p[2],er*p[2],lw*p[2],up*p[2]);
  fprintf(fp,"4, eres,%7.4f,%7.4f,%7.4f,%7.4f\n",p[3],er*p[3],lw*p[3],up*p[3]);
  fprintf(fp,"5, resf,%5.2f,%5.2f,%5.2f,%5.2f\n",p[4],er*p[4],lw*p[4],up*p[4]);
  fprintf(fp,"6, dk,%7.4f,%7.4f, -0.01, 0.01\n", p[5], .01 * ABS(p[5]));
  fprintf(fp,"7, pk,%7.4f,%7.4f, -0.5, 0.5 \n", p[6], .01 * ABS(p[6]));
  for (i = 7; i < nparm; i++) {
    fprintf(fp,"%2d, p%2d,%8.3f,%7.4f,%8.3f,%8.3f\n",
            i+1, i+1, p[i], er*ABS(p[i]), lw*p[i], up*p[i]);
  }
```

```
  fprintf(fp,"\n");

  fclose(fp);
  if (debug>1) fprintf(fpdeb,"\n\t\t========== wrtparm end ==========\n");
}
```

```
/*****************************************************************
*  xscx.c  - Calculate artificial excitation functions for a single
*          detector (angle). The effect of the energy resolution
*          function is NOT included here, but previously accounted for
*          in the passed values lr and li.  Xsex calculates cross
*          section and analyzing power as the interference between
*          resonant and non-resonant scattering amplitudes.
*          The non-resonant amplitudes are parameterized by the passed
*          arguments. The non-resonant amplitudes are assumed independent
*          of energy and thus representable by a scaler data type, while
*          resonant amplitudes are represented by arrays. However, a
*          constant background slope expressed in (mb/sr)/channel is
*          allowed.
*/

#include "resfit.h"
extern int conidx, xcalc, useint;
extern REAL gsqr;       /* geometric terms from initdat */
extern char fittype;

void xsex(ar, ai, mag, prm3, pdat, pinf, gam, gixgo, lr, li, pyr, pyi, pk, fit)
REAL ar, ai;      /* non-resonant amplitudes for this detector */
REAL mag;         /* background parm magnitude squared (ar*ar + ai*ai) */
REAL prm3;        /* parameter 3; may represent non-res xsection */
struct XFCN *pdat;  /* pointer to data structure for a particular angle */
struct INFO *pinf;  /* pointer to info structure */
REAL gam;         /* total width */
REAL gixgo, pk;    /* product entrance & exit part widths, pygmy probability*/
REAL lr[], li[];   /* convoluted integrals, real and imaginary parts */
REAL pyr[], pyi[];  /* integrals for pygmy resonance */
double fit[];      /* pointer to resulting array */
{
  register int i, j;
  REAL c0, c1, c2, c0sl;

  if (debug>1) fprintf(fpdeb,"\n\t\t========== xsex begin ==========\n");
  if (debug) fprintf(fpdeb," xsex: \n");
  if (debug > 3) fprintf(fpdeb,"xsex: \n sig \n");

  if (fittype != 's') {
    printf("This program for cross section fits only!!");
    exit(0);
```

```
}

/* if (pinf->type == 1) { */          /* for (a,a) */
/* else if (pinf->type == 3) { */     /* for (p,a) */
/* else if (pinf->type == 2) { */     /* for (p,p) */

switch (xcalc) {

  case 1:
    if (debug>1) fprintf(fpdeb,"calc flag, xcalc = %d \n", xcalc);
    if (useint == 'y') {
      if (debug>1) fprintf(fpdeb,"using intercept for back xsec\n");
      c0 = pdat->bint * prm3;         /* prm3 should really be = 1 */
    }
    else {
      if (debug>1) fprintf(fpdeb,"calculate back xsec from amps\n");
      c0 = pinf->avin * (mag + prm3);
    }
    if (debug>1) {
      fprintf(fpdeb,"average factor, avin = %lf\n",pinf->avin);
      fprintf(fpdeb,"geometric factor, gfac = %lf\n",pinf->gfac);
      fprintf(fpdeb,"\t \t, ar = %lf\n",ar);
      fprintf(fpdeb,"\t \t, ai = %lf\n",ai);
      fprintf(fpdeb,"product of widths, gixgo = %lf\n",gixgo);
      fprintf(fpdeb,"total width, gam = %lf\n",gam);
    }
    c1 = pinf->avin * 2.0 * pinf->gfac * ar * sqrt(gixgo);
    c2 = pinf->avin * pinf->gfac*(pinf->gfac*gixgo - ai*gam*sqrt(gixgo));

    if (debug>1) fprintf(fpdeb,"c0, c1, c2 = %lf, %lf, %lf\n",c0,c1,c2);
    break;

  case 2:
    if (debug>1) fprintf(fpdeb,"calc flag, xcalc = %d \n", xcalc);
    if (useint == 'y') {
      if (debug>1) fprintf(fpdeb,"using intercept for back xsec\n");
      c0 = pdat->bint * prm3;         /* prm3 should really be = 1 */
    }
    else {
      if (debug>1) fprintf(fpdeb,"calculate back xsec from amps\n");
      c0 = prm3;
    }
    c1 = ar;
    c2 = ai;

    if (debug>1) fprintf(fpdeb,"c0, c1, c2 = %lf, %lf, %lf\n",c0,c1,c2);
    break;

  default:
    printf("**** illegal value for calculation mode!\n");
    c0 = 0.0; c1 = 0.0; c2 = 0.0;
    break;
}

if (pdat->bslp == 0.0) {
  for (i = 0, j = conidx; i < pinf->pts; i++, j++) {

    fit[i] = ((c0 + (c1*lr[j] + c2*li[j])) * (1-pk))
           + ((c0 + (c1*pyr[j] + c2*pyi[j])) * pk);

    if (debug > 3) fprintf(fpdeb,"xsex: %lf\n",fit[i]);
  }
}
else {                                /* non-zero slope */
  if (debug > 1) printf("background type = %c\n", pdat->btype);
  for (i = 0, j = conidx; i < pinf->pts; i++, j++) {
    if (pdat->btype == 'e') c0sl = c0 * exp(pdat->bslp*i);
    else c0sl = c0 + i*pdat->bslp;

    fit[i] = ((c0sl + (c1*lr[j] + c2*li[j])) * (1-pk))
           + ((c0sl + (c1*pyr[j] + c2*pyi[j])) * pk);

    if (debug > 3) fprintf(fpdeb,"xsex: %lf\n",fit[i]);
  }
}

if (debug>1) fprintf(fpdeb,"\n\t\t========== xsex end ==========\n");
}
```

REFERENCES

[Abr72] M. Abramowitz & I. A. Stegun, eds., Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables, 10th printing (1972) National Bureau of Standards Applied Mathematics Series, **55**, 998.

[Ajz87] F. Ajzenberg-Selove, "Energy Levels of Light Nuclei A = 18-20," Nuclear Physics **A475** (1987) 1-198.

[Arf70] George Arfken, Mathematical Methods for Physicists, Academic Press, New York (1970).

[Ash76] Neil W. Ashcroft & N. David Mermin, Solid State Physics, Saunders College, Philadelphia, Holt, Rinehart, and Winston (1976).

[Bak84] Steve Baker and Robert D. Cousins, "Clarification of the use of Chi-square and Likelihood Functions in Fits to Histograms," Nuclear Instruments and Methods in Physics Research **221** (1984) 437-442.

[Ban66] A. P. Banford, The Transport of Charged Particle Beams, E. & F. N. Spon, Ltd., London (1966).

[Bea79] K. G. Beauchamp, C. K. Yuen, Digital Methods for Signal Analysis, George Allen & Unwin, Ltd. (1979).

[Bev69] Philip R. Bevington, Data Reduction and Error Analysis for the Physical Sciences, McGraw-Hill, New York (1969).

[Bic75] Hans Bichsel and Roberta P. Saxon, Physical Review **A11** (1975) 1286.

[Ble78] M. E. Bleck, Ph.D. Dissertation, Duke University, 1978 (unpublished), available from University Microfilms International, 300 N. Zeeb Rd., Ann Arbor, MI 48106.

[Blo67] R. Bloch, R. E. Pixley, and P. Truöl, "Isobaric Spin Impurity of the Lowest T=2 State in ^{20}Ne", Physics Letters **25B** (1967) 215.

[Boh69] A. Bohr & B. R. Mottleson, Nuclear Structure, Volume 1, Benjamin, New York (1969).

[Bon68] B. Bonsnjakovic *et al.*, "High-resolution Investigation of the Reaction ^{35}Cl(p,a_0)^{32}S", Nuclear Physics **A110** (1968) 17.

[Bri27] P. W. Bridgman, "The Compressibility and Pressure Coefficient of Resistance of Ten Elements", Proceedings of the American Academy of Arts and Science **62** (1927) 207-226, Reprinted in Collected Experimental Papers, Volume IV, Harvard University Press, Cambridge (1964).

[Bro74] D. Allan Bromley, ed., Large Electrostatic Accelerators, North-Holland Publishing Company (1974).

[Bur85] C. S. Burrus and T. W. Parks, DFT/FFT and Convolution Algorithms, John Wiley & Sons, New York (1985) 83.

[Cer68] J. Cerny, Annual Reviews of Nuclear Science **18** (1968) 27.

[Chi89] Charles B. Childs, Gurtej Sandhu, & Wei-Kan Chu, "Process for Increasing Gold-Diamond Adhesion and for Improving the Surface Smoothness of Gold Films on Diamond", Final Program and Presentation Summaries, SDIO/IST - ONR Diamond Technology Initiative Symposium, Crystal City, Virginia (1989).

[Chu78] W. K. Chu & Nicolet, Backscattering Spectrometry, Academic Press, New York (1978).

[Com83] E. Comay and J. Jänecke, "Shell-Model Coulomb Energies," Nuclear Physics **A410** (1983) 103.

[Cos64] D. G. Costello, J. G. Skofronick, A. L. Morsell, D. W. Palmer, & R. G. Herb, "Atomic Effects on Nuclear Resonance Reaction Yield Curves of Aluminum and Nickel," Nuclear Physics **51** (1964) 113.

[Cro88] E. R. Crosson, private communication (1988).

[Ead71] W. T. Eadie *et al.*, Statistical Methods in Experimental Physics, North-Holland Publishing Company, Amsterdam (1971).

[Ehr51] Joachim B. Ehrman, "On the Displacement of Corresponding Energy Levels of ^{13}C & ^{13}N," Physical Review **81** (1951) 412.

[Ell69] J. P. Elliott, "Nuclear Symmetries," in Isospin in Nuclear Physics, ed. D. H. Wilkinson, North-Holland Publishing Company, Amsterdam (1969) 75.

[End78] P. M. Endt and C. Van der Leun, "Energy Levels of $A = 21 - 44$ Nuclei (VI)," Nuclear Physics **A310** (1978) 1.

[Fle70] R. Fletcher, "A new approach to variable metric algorithms", Computer Journal **13** (1970) 317.

[Fre78] S. J. Freedman *et al.*, "Decays of the Lowest $T = 2$ State in ^{44}Ti," Physical Review **C17** (1978) 2071.

[Fre79] S. J. Freedman *et al.*, "Decays of the Lowest $T = 2$ States in $A = 4N$ Nuclei from ^{8}Be to ^{44}Ti," Physical Review **C19** (1979) 1907.

[Fuk81] Eiichi Fukushima and Stephen B. W. Roeder, Experimental Pulse NMR - A Nuts and Bolts Approach, Addison Wessley, Reading, Mass. (1981).

[Gar69] G. T. Garvey, "Present Status of the Isobaric Multiplet Mass Equation," in

Nuclear Isospin, Proceedings of the Conference on Nuclear Isospin, Pacific Grove, California, ed. J. D. Anderson, S. D. Bloom, J. Cerny, W. W. True, Academic Press, New York (1969).

[Gib73] J. F. Gibbons and S. Mylroie, "Estimation of impurity profiles in ion-implanted amorphous targets using joined half-Gaussian distributions", Applied Physics Letters **22** (1973) 568.

[Gou83] C. R. Gould and N. R. Roberson, "VAX 11/780 Data Acquisition Facility at Triangle Universities Nuclear Laboratory", IEEE Transactions on Nuclear Science **NS-30** (1983) 3758.

[Gra72] Dwight E. Gray, coordinating editor, American Institute of Physics Handbook, 3rd edition, McGraw-Hill Book Company, New York (1972).

[Gro81] H. Grote, "Data Analysis for Electronic Experiments", from lectures presented at the 1980 CERN School of Computing, Vraona, Attiki, Greece (1980).

[Har70] J. C. Hardy, H. Brunnader, & Joseph Cerny, "$T = 2$ and $T = 3$ Analog States, $28 \leq A \leq 40$," Physical Review **C1** (1970) 561.

[Heg77] J. C. P. Heggie and H. H. Bolotin, "Properties of the Lowest $T = 2$ State in ^{24}Mg," Australian Journal of Physics **30** (1977) 407.

[Hen69] Ernest M. Henley, "Charge Independence and Charge Symmetry of Nuclear Forces," in Isospin in Nuclear Physics, ed. D. H. Wilkinson, North-Holland Publishing Company, Amsterdam (1969) 15.

[Hes87] S. K. B. Hesmondhalgh *et al.*, "Measurement of the Alpha Particle Decay Width of the First 0^+, $T = 2$ State in ^{40}Ca", Physics Letters **B183** (1987) 35.

[Hin75] F. Hinterberger *et al.*, Nuclear Instruments and Methods, **130** (1975) 347.

[Hin81] F. Hinterberger *et al.*, Nuclear Physics **A352** (1981) 93.

[Hin84] F. Hinterberger and P. D. Eversheim *et al.*, " ^{20}Ne + n -> ^{21}Ne ($T = 3/2$) Resonances," Nuclear Physics **A424** (1984) 200.

[Hoo64] B.W. Hooten, "The Deposition of Thin Films of Red Phosphorus," Nuclear Instruments and Methods **27** (1964) 338.

[How82] C. R. Howell and S. A. Wender, Nuclear Instruments and Methods **195** (1982) 443.

[Iko75] P.G. Ikossi, Ph.D. Dissertation, University of North Carolina (1975) (unpublished), available from University Microfilms International, 300 N. Zeeb Rd., Ann Arbor, MI 48106.

[Iko79] P. G. Ikossi *et al.*,"The α-Width of the Lowest $T = 2$ State in ^{28}Si," Nuclear Physics **A319** (1979) 109.

[Jam72] F. James, "Function Minimization," reprinted from the Proceedings of the 1972 CERN Computing and Data Processing School, Pertisau, Austria (1972)

[Jam80] Frederick James, "Interpretation of the Shape of the Likelihood Function Around its Minimum," Computer Physics Communications **20** (1980) 29.

[Jam81] F. James, "Determining the Statistical Significance of Experimental Results," from Lectures presented at the 1980 CERN School of Computing, Vraona, Attiki, Greece (1980).

[Jän69] Joachim Jänecke, "Systematics of Coulomb Energies and Excitation Energies of Isobaric Analog States," in Isospin in Nuclear Physics, ed. D.H. Wilkinson, North-Holland Publishing Company, Amsterdam (1969) 297.

[Jän84] J. Jänecke & E. Comay, "Shell Model Coulomb Energies with Deformation", Physics Letters **140B** (1984) 1.

[Jun82] C. R. Jungmann *et al.*, Nuclear Physics **A386** (1982) 287.

[Kua67] H. M. Kuan, D. W. Heikkinen, K. A. Snover, F. Reiss, S. S. Hanna, Physics Letters **25B** (1967).

[Law 65] G. P. Lawrence, R. K. Beauchamp, and J. L. McKibben, "Direct Extraction of Negative Ion Beams of Good Intensity from a Duoplasmatron", Nuclear Instruments and Methods **32** (1965) 357.

[Lee72] H. C. Lee and R. Y. Cusson, Annals of Physics **72** (1972) 353.

[Lem87] S. Lemieux and K. Keeter, private communication (1987).

[Lew62] H. W. Lewis, "Straggling Effects on Resonant Yields," Physical Review **125**, (1962) 937.

[Liv69] John J. Livingood, The Optics of Dipole Magnets, Academic Press, New York (1969).

[Loz84] W. R. Lozowski, private communication (1984).

[Lud86] Lonnie C. Ludeman, Fundamentals of Digital Signal Processing, Harper & Row, New York (1986) 28.

[Lyn68] J. E. Lynn, The Theory of Neutron Resonance Reactions, Clarendon Press, Oxford (1968) 23.

[Mar69] Pierre Marmier and Eric Sheldon, Physics of Nuclei and Particles, Vol. I, Academic Press (1969).

[Mat64] Jon Mathews and R. L. Walker, Mathematical Methods of Physics, W. A. Benjamin, Inc., New York (1964).

[Max67] S. H. Maxman, "Target Preparation Techniques," Nuclear Instruments and Methods **50** (1967) 53.

[McC68] I. E. McCarthy, Introduction to Nuclear Theory, John Wiley & Sons, Inc., New York (1968).

[McC79] James H. McClellan, Charles M. Rader, Number Theory in Digital Signal Processing, Prentice-Hall (1979).

[McD78] A. B. McDonald *et al.*, "Isospin-forbidden Particle Decays in Light Nuclei (IV). Total width of the lowest $T = 2$ level of ^{24}Mg," Nuclear Physics **A305** (1978) 151.

[McG70] Robert L. McGrath *et al.*, "Isospin-Forbidden Decay Properties of the Lowest $T = 2$ States of ^{20}Ne, ^{24}Mg, ^{28}Si, and ^{40}Ca," Physical Review **C1** (1970) 184.

[Mer70] Eugen Merzbacher, Quantum Mechanics, 2nd edition, John Wiley & Sons, New York (1970).

[Mid74] R. Middleton, "A Survey of Negative Ion Sources for Tandem Accelerators", Nuclear Instruments and Methods **122** (1974) 35.

[Moo89] T. M. Mooney, Ph.D. Dissertation, University of North Carolina (1989) (unpublished), available from University Microfilms International, 300 N. Zeeb Rd., Ann Arbor, MI 48106.

[Mor76] R. Moreh, O.Shahal, and V. Volterra, "Effect of Molecular Binding on the Resonance Scattering of Photons from the 6.324 MeV Level in ^{15}N," Nuclear Physics **A262** (1976) 221.

[Mor79] R. Moreh, "Experiments in Solid State Physics Using Resonant Scattering of Gamma-rays," Nuclear Instruments and Methods **166** (1979) 45.

[Mug79] A. H. F. Muggeleton, "Preparation of Thin Nuclear Targets," Journal of Physics **E12** (1979) 780.

[Nel65] J. A. Nelder and R. Mead, "A simplex method for function minimization," Computer Journal **7** (1965) 308.

[Nus82] Henri J. Nussbaumer, Fast Fourier Transform and Convolution Algorithms, Springer-Verlag (1982).

[Oma75] M. A. Omar, Elementary Solid State Physics, Addison-Wesley (1975) 77.

[Orm86] W. E. Ormand, Ph. D. dissertation, Michigan State University (1986) (unpublished), available from University Microfilms International, 300 N. Zeeb Road, Ann Arbor, MI 48106.

[Osb78] J. L. Osborne, E. G. Adelberger, & K. A. Snover, "Total Width of the Lowest $T = 2$ State in ^{24}Mg," Nuclear Physics **A305** (1978) 144.

[Pap77] Athanasios Papoulis, Signal Analysis, McGraw-Hill (1977).

[Pre62] M. A. Preston, Physics of the Nucleus, Addison-Wesley Publishing Company, Reading, Massachusetts (1962).

[Pre86] William H. Press *et al.*,Numerical Recipes, Cambridge University Press (1986).

[Pri82] D. M. Pringle *et al.*, "Decay of the Lowest $T = 2$ State in ^{40}Ca," Physics Letters **115B** (1982) 291.

[Rie67] F. Riess, W. J. O'Connell, D. W. Heikkinen, H. M. Kuan, & S. S. Hanna, "Observation of $T = 2$ Level in ^{24}Mg as a Compound-Nucleus Resonance," Physical Review Letters **19** (1967) 367.

[Rob78] R. G. H. Robertson *et al.*, "Mass of Lowest $T = 2$ State of ^{12}C," Physical Review **C17** (1978) 1535.

[Sat80] G. R. Satchler, Introduction to Nuclear Reactions, John Wiley & Sons, New York (1980).

[Sat83] G. R. Satchler, Direct Nuclear Reactions, Oxford University Press, New York (1983).

[Sch74] B. Schorr, Computer Physics Communications **7** (1974) 215.

[Sei40] Frederick Seitz, The Modern Theory of Solids, 1st edition, McGraw-Hill Book Company, Inc., New York and London (1940).

[Sli78] C. P. Slichter, Principles of Magnetic Resonance, 2nd ed., Springer-Verlag, Berlin (1978).

[Sop69] J. M. Soper, "Isospin Purity of Low-lying Nuclear States," in Isospin in Nuclear Physics, ed. D. H. Wilkinson, North-Holland Publishing Company, Amsterdam (1969) 231.

[Tak66] S. Takayanagi *et al.*, "On the Preparation of Magnesium Targets from MgO," Nuclear Instruments and Methods **45** (1966) 345.

[Tay82] John R. Taylor, An Introduction to Error Analysis, University Science Books, Oxford University Press (1982).

[Tay83] John R. Taylor, Scattering Theory: The Quantum Theory of Nonrelativistic Collisions, Robert E. Krieger Publishing Company, Malabar, Florida (1979 & 1983).

[Tho89] W. J. Thompson, private communication (1989).

[Tze78] Huw-Shing Tzeng *et al.*, "Preparation of Thin Red-Phosphorus," Nuclear Instruments and Methods **150** (1978) 143.

[Van85] J. Vanhoy, private communication (1985).

[Var86] R. L. Varner, Ph.D. Dissertation, University of North Carolina (1986) (unpublished), available from University Microfilms International, 300 N. Zeeb Rd., Ann Arbor, MI 48106.

[Ver73] J. Vernotte *et al.*, "Investigation of the Lowest $T = 2$ State of ^{32}S in the ^{31}P + p Reactions," Physical Review **C8** (1973) 178.

[Wap85] A. H. Wapstra and G. Audi, "The 1983 Atomic Mass Evaluation," Nuclear Physics **A432** (1985) 1.

[Wes84] C. R. Westerfeldt *et al.*, "A System for Producing High-Resolution Polarized and Unpolarized Beams with a Tandem Accelerator," Nuclear Instruments and Methods in Physics Research **219** (1984) 284.

[Wes84b] C. R. Westerfeldt, private communication (1984).

[Wes89] C. R. Westerfeldt, private communication (1989).

[Wic58] G. C. Wick, "Invariance Principles of Nuclear Physics," Annual Review of Nuclear Science **8** (1958) 1.

[Wil69] D. H. Wilkinson, ed., Isospin in Nuclear Physics, North-Holland Publishing Company (1969).

[Wil69a] D. H. Wilkinson, "Historical Introduction to Isospin," in Isospin in Nuclear Physics, ed. D.H. Wilkinson, North-Holland Publishing Company (1969) 1.

[Wil82] J. F. Wilkerson, Ph.D. Dissertation, University of North Carolina (1982) (unpublished), available from University Microfilms International, 300 N. Zeeb Rd., Ann Arbor, MI 48106.

[Wil83] J. F. Wilkerson *et al.*, "Isospin and Isobar Components in Light Nuclei," Physical Review Letters **51** (1983) 2269.

[Wil83a] J. F. Wilkerson, T. B. Clegg, and E. J. Ludwig, "An Energy Calibration of the TUNL Dual-90° Magnet Analyzing System," Nuclear Instruments and Methods **207** (1983) 2269.