

AN ABSTRACT OF THE THESIS OF

Nupan Kheaomaingam for the degree of Doctor of Philosophy in Physics presented on February 24, 2000.

Title: Contribution from Two On-shell Pion Exchange in Nucleon-Nucleon Scattering

Abstract approved: -

/ /

Philip J. Siemens

We used recent phenomenological form factors to calculate the effect of on-shell two-pion exchange in nucleon-nucleon scattering below the two-pion production threshold. As expected, the contributions of partial wave amplitudes are all negligible. We also noticed that the relativistic spin-operator decomposition of the scattering amplitude is not unique at certain momenta and angles.

Contribution from Two On-shell Pion Exchange in Nucleon-Nucleon Scattering

by

Nupan Kheaomaingam

A Thesis

submitted to

Oregon State University


in partial fulfillment of
the requirements for the
degree of

Doctor of Philosophy

Completed February 24, 2000
Commencement June 2000

Doctor of Philosophy thesis of Nupan Kheaomaingam presented on
February 24, 2000

APPROVED:



Major Professor, representing Physics

Chair of the Department of Physics

Redacted for privacy



Dean of the Graduate School

I understand that my thesis will become part of the permanent collection of Oregon State University libraries. My signature below authorizes release of my thesis to any reader upon request.

Redacted for privacy

Nupan Kheaomaingam, Author

ACKNOWLEDGMENT

I am grateful to Professor Phil Siemens for guidance and insightful discussion during this project. On many occasions, he helped me understand the physics behind the seemingly meaningless mathematical formulas. His kindness and patience also lessened the frustration of the slow progress of the project.

I am also grateful for the suggestion and perspective given by my thesis committee, Professor Al Stetz, Professor Henri Jansen, Professor David Griffiths, and Professor Burt Fein.

Thanks to many friends, especially Dr. Sirinmas Katchamart (สิริมาศ คัชมาตย์), for encouragement.

Finally, I acknowledge the Royal Thai Government, and Burapha University (มหาวิทยาลัยบูรพา) for financial support of my graduate studies.

TABLE OF CONTENTS

	<u>Page</u>
1 INTRODUCTION	1
2 THEORY	4
3 COMPUTATION AND RESULT	10
4 CONCLUSION	37
BIBLIOGRAPHY	38
APPENDICES	40
Appendix A Traces of Operators	41
Appendix B Source Code	43

LIST OF FIGURES

<u>Figure</u>	<u>Page</u>
2.1 Feynman diagram of TPE nucleon-nucleon interaction.	4
3.1 Region of integration for $\sqrt{s} = 2m_N + 0.01m_\pi$ at $\cos\theta = 0.5$. The units of k_1^2 and k_2^2 are $(GeV/c)^2$. The cross hatching il- lustrates the numerical integration scheme, but the actual grid used was much denser.	11
3.2 Region of integration for $\sqrt{s} = 2m_N + 0.9m_\pi$ at $\cos\theta = 0.5$. The units of k_1^2 and k_2^2 are $(GeV/c)^2$	12
3.3 Region of integration for $\sqrt{s} = 2m_N + 0.9m_\pi$ at $\cos\theta =$ 0.054 (pole). The units of k_1^2 and k_2^2 are $(GeV/c)^2$	12
3.4 Amplitude c_1 as function of k_1^2 and k_2^2 : $\sqrt{s} = 2m_N +$ $0.01m_\pi, \cos\theta = 0.5$	14
3.5 Amplitude c_s as function of k_1^2 and k_2^2 : $\sqrt{s} = 2m_N +$ $0.01m_\pi, \cos\theta = 0.5$	15
3.6 Amplitude c_{so} as function of k_1^2 and k_2^2 : $\sqrt{s} = 2m_N +$ $0.01m_\pi, \cos\theta = 0.5$	15
3.7 Amplitude c_t as function of k_1^2 and k_2^2 : $\sqrt{s} = 2m_N +$ $0.01m_\pi, \cos\theta = 0.5$	16
3.8 Amplitude c_1 as function of k_1^2 and k_2^2 : $\sqrt{s} = 2m_N +$ $0.9m_\pi, \cos\theta = 0.5$	16
3.9 Amplitude c_s as function of k_1^2 and k_2^2 : $\sqrt{s} = 2m_N +$ $0.9m_\pi, \cos\theta = 0.5$	17
3.10 Amplitude c_{so} as function of k_1^2 and k_2^2 : $\sqrt{s} = 2m_N +$ $0.9m_\pi, \cos\theta = 0.5$	17
3.11 Amplitude c_t as function of k_1^2 and k_2^2 : $\sqrt{s} = 2m_N +$ $0.9m_\pi, \cos\theta = 0.5$	18
3.12 Amplitude c_1 as function of k_1^2 and k_2^2 : $\sqrt{s} = 2m_N +$ $0.9m_\pi, \cos\theta = 0.054$ (pole).	18
3.13 Amplitude c_s as function of k_1^2 and k_2^2 : $\sqrt{s} = 2m_N +$ $0.9m_\pi, \cos\theta = 0.054$ (pole).	19

LIST OF FIGURES (Continued)

<u>Figure</u>	<u>Page</u>
3.14 Amplitude c_{so} as function of k_1^2 and k_2^2 : $\sqrt{s} = 2m_N + 0.9m_\pi, \cos\theta = 0.054$ (pole).	19
3.15 Amplitude c_t as function of k_1^2 and k_2^2 : $\sqrt{s} = 2m_N + 0.9m_\pi, \cos\theta = 0.054$ (pole).	20
3.16 Contribution from scalar operator as a function of $\cos\theta$ for $\sqrt{s} = 2m_N + 0.01m_\pi$	21
3.17 Contribution from spin-spin operator as a function of $\cos\theta$ for $\sqrt{s} = 2m_N + 0.01m_\pi$	21
3.18 Contribution from spin-orbit operator as a function of $\cos\theta$ for $\sqrt{s} = 2m_N + 0.01m_\pi$	22
3.19 Contribution from tensor operator as a function of $\cos\theta$ for $\sqrt{s} = 2m_N + 0.01m_\pi$	22
3.20 Contribution from scalar operator as a function of $\cos\theta$ for $\sqrt{s} = 2m_N + 0.9m_\pi$	23
3.21 Contribution from spin-spin operator as a function of $\cos\theta$ for $\sqrt{s} = 2m_N + 0.9m_\pi$	23
3.22 Contribution from spin-orbit operator as a function of $\cos\theta$ for $\sqrt{s} = 2m_N + 0.9m_\pi$	24
3.23 Contribution from tensor operator as a function of $\cos\theta$ for $\sqrt{s} = 2m_N + 0.9m_\pi$	24
3.24 Poles' vicinity of Figure 3.16.	25
3.25 Poles' vicinity of Figure 3.17.	25
3.26 Poles' vicinity of Figure 3.18.	26
3.27 Poles' vicinity of Figure 3.19.	26
3.28 Poles' vicinity of Figure 3.20.	27
3.29 Poles' vicinity of Figure 3.21.	27
3.30 Poles' vicinity of Figure 3.22.	28
3.31 Poles' vicinity of Figure 3.23.	28

LIST OF FIGURES (Continued)

<u>Figure</u>	<u>Page</u>
3.32 Effect of form factor on partial wave δ_{100}	34
3.33 Contribution of scalar operator for $\sqrt{s} = 2m_N + 0.9m_\pi$, at $\cos\theta = 0.5$ and where k_- is at the kinematic limit, $V(k_+, k_-) =$ 0. See equation (3.12) through equation (3.17).	34
3.34 Contribution of spin-spin operator for the same condition as Figure 3.33.	35
3.35 Contribution of spin-orbit operator for the same condition as Figure 3.33.	35
3.36 Contribution of tensor operator for the same condition as Fig- ure 3.33.	36

LIST OF TABLES

<u>Table</u>		<u>Page</u>
3.1	Traces of operators in center-of-mass system.	13
3.2	Eigenvalues and eigenvectors for $\sqrt{s} = 2m_N + 0.9m_\pi$	30
3.3	Comparison between methods of integration for the case $\sqrt{s} =$ $2m_N + 0.9m_\pi$	31
3.4	Phase shift from TPE for the case of $j = 0, 1$ in unit of degree.	32
3.5	Phase shift from TPE for the case of $j = 2$	33

CONTRIBUTION FROM TWO ON-SHELL PION EXCHANGE IN NUCLEON-NUCLEON SCATTERING

1. INTRODUCTION

The interaction of nucleons at large distances must be dominated by the exchange of pions. The exchange of a single pion reaches farthest, with a range of $\frac{\hbar c}{m_\pi} = 1.4$ fm. Uncorrelated two-pion exchange would have half that range, comparable to the r.m.s. charge radius of a single nucleon. At that distance the quark distributions of the nucleons must surely overlap significantly, so the exchange of multiple pions cannot be expected to give an accurate accounting of the nucleons' interactions.

Nevertheless the two-pion exchange amplitude has to play a role in the scattering, at least to unitarize the single-pion exchange. [1] It is also thought to play a role in the binding energy of nuclear matter, to which single-pion exchange does not contribute due to isospin symmetry. The facts that the intermediate nucleon is inhibited by the Pauli principle, and that its self-energy is modified by the nuclear medium, have long been argued to contribute to the saturation of nuclear matter at the observed average spacing of about 2 fm. [2]

There are two widely applied methods to estimate the two-pion exchange contribution to the scattering amplitude, dispersion theory and phenomenological field theory. Dispersion theory can be used to express the NN amplitude in terms of the observable rate for pion production in nucleon annihilation [3, 4],

$$N\bar{N} \rightarrow \pi\pi.$$

This has the advantage of including the correlations of the exchanged pions, but is handicapped by the difficulty of the experiments and their analysis. In phenomenological field theory, the pion-nucleon vertices are dressed with form factors fit to other scattering data [5–7].

In a tour de force of analysis, Hamilton and Oades [8] showed that the amplitude for two-pion exchange in nucleon scattering can be deduced by applying unitarity and causality to real multiple-scattering processes in which pairs of intermediate particles are on their energy shells. The case which has been extensively studied is when the intermediate nucleons are both on-shell,

$$NN \rightarrow NN \rightarrow NN,$$

i.e. true multiple scattering. However there is a second case with both pions on-shell. This contribution would be expected to be much smaller than the amplitude of nucleon rescattering, because the intermediate nucleons must be far off-shell. That leads both to large denominators from the off-shell propagators, and small vertex factors from the off-shell form factors. However this reasonable assumption could not be tested for lack of knowledge of the form factors for off-shell nucleons with on-shell pions. A recent analysis of

pion-nucleon scattering has yielded new information in the relevant kinematic regime [9]. Using this new knowledge, we calculate for the first time the contribution of on-shell two-pion exchange to nucleon-nucleon scattering. We are disappointed but not surprised to confirm that the amplitude for this process is too small to matter.

In the process of carrying out the computation, we developed new computational techniques which may be of interest in their own right. A judicious combination of computer programs using symbolic algebra and numerical methods proved effective in carrying out a very complicated computation.

Our computational technique includes a relativistic characterization of the spin structure of the scattering amplitude in terms of covariant operators. We are surprised to report that, under certain kinematic conditions, the resolution of the spin structure into operator amplitudes is not unique.

2. THEORY

The Feynman diagrams of the two-pion exchange interaction are shown in figure 2. A nucleon has incoming momentum p_1 and outgoing momentum p_3 , while another nucleon has incoming momentum p_2 and outgoing momentum p_4 . Momenta q_1 and q_2 are transferred by two intermediate pions. Nucleon 1 and 2 have intermediate momenta k_1 and k_2 respectively.

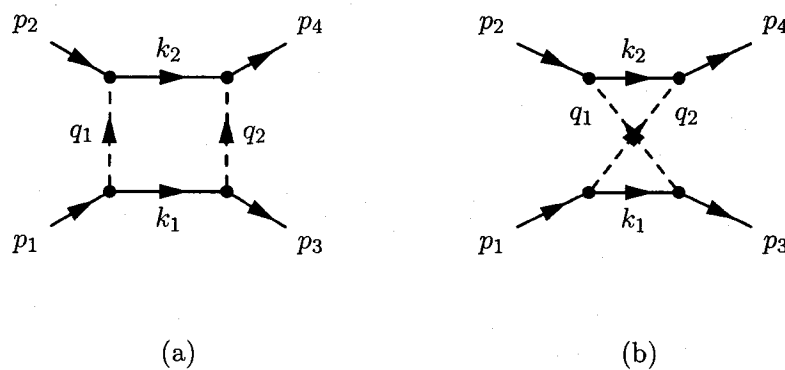


Figure 2.1: Feynman diagram of TPE nucleon-nucleon interaction. Solid lines represent external and intermediate nucleons. Dashed lines represent intermediate pions. (a) ladder diagram; (b) crossed diagram.

We have calculated both diagrams as functions of the external 4-momenta and spinors. However, we did not carry out the partial wave decomposition of the crossed diagram, which appeared similar to the ladder diagram. The

scattering amplitude M of the ladder diagram can be calculated using Feynman's rules described in a quantum field theory book such as [10]:

$$\begin{aligned}
M = & -i \left(\frac{f}{m_\pi} \right)^4 \int \frac{d^4 k_1}{(2\pi)^4} \\
& \times \bar{u}_1(p_3, s_3) (\gamma^5 \not{k}_1 \gamma^5 \not{k}_1 + m_N \gamma^5 \not{k}_1 \gamma^5 \not{k}_1) u_1(p_1, s_1) \\
& \times \bar{u}_2(p_4, s_4) (\gamma^5 \not{k}_2 \gamma^5 \not{k}_2 + m_N \gamma^5 \not{k}_2 \gamma^5 \not{k}_2) u_2(p_2, s_2) \\
& \times \frac{1}{(q_1^2 - m_\pi^2 - i\epsilon)(q_2^2 - m_\pi^2 - i\epsilon)(k_1^2 - m_N^2 - i\epsilon)(k_2^2 - m_N^2 - i\epsilon)}. \quad (2.1)
\end{aligned}$$

A Dirac spinor is

$$u(p, s) = \sqrt{p_0 + m_N} \begin{pmatrix} 1 \\ \frac{\vec{\sigma} \cdot \vec{p}}{p_0 + m_N} \end{pmatrix} \chi^s, \quad (2.2)$$

where χ^s is a Pauli spinor. The spinors' subscripts 1, 2 indicate nucleon 1 and 2. We have omitted the isospin factors and antisymmetrization, which are most conveniently included as factors multiplying the partial-wave amplitudes.

We can rewrite the above expression as

$$M = -i \left(\frac{f}{m_\pi} \right)^4 \bar{u}_1(p_3, s_3) \bar{u}_2(p_4, s_4) \left(\int \frac{d^4 k_1}{(2\pi)^4} Op D \right) u_1(p_1, s_1) u_2(p_2, s_2), \quad (2.3)$$

where D is the propagator part, and the vertices make the spin operator

$$Op = (\gamma^5 \not{k}_1 \gamma^5 \not{k}_1 + m_N \gamma^5 \not{k}_1 \gamma^5 \not{k}_1)_1 (\gamma^5 \not{k}_2 \gamma^5 \not{k}_2 + m_N \gamma^5 \not{k}_2 \gamma^5 \not{k}_2)_2. \quad (2.4)$$

Again, subscripts 1, 2 outside parentheses indicate with which nucleons the operators associate.

The propagator of an intermediate particle is

$$\frac{1}{q^2 - m^2 - i\epsilon} = P \frac{1}{q^2 - m^2} + i\pi \delta(q^2 - m^2), \quad (2.5)$$

where P is the principle value integral. As discussed in the introduction, we are interested in the case where the two intermediate pions are on shell. In

this case conservation laws require the nucleons to be off shell. Hence the propagator factor in the matrix element (2.3) is

$$D = -\pi^2 \delta(q_1^2 - m_\pi^2) \delta(q_2^2 - m_\pi^2) \frac{P}{k_1^2 - m_N^2} \frac{P}{k_2^2 - m_N^2}. \quad (2.6)$$

Causality requires an equal contribution from the term with all 4 principal values [11]. The terms with odd numbers of intermediate particles on-shell are vertex corrections and are not kinematically accessible.

The covariant variables, $k_1^2, k_2^2, q_1^2, q_2^2$, in the propagators and Dirac delta function arguments suggest the transformation of coordinates from 4-momentum k_1 . To do so, we have to find the Jacobian of the transformation, and transform the expression for the operator Op .

The 4 external momenta, together with the loop momentum k_1 , define a pentahedron in 4-momentum space; the 4 invariant momenta correspond to the edges of the pentahedron adjacent to the loop momentum. [12] The Jacobian J of the transformation is the volume V of the corresponding parallelepiped, $J = \frac{1}{16V}$. The square of this volume V is given by a determinant of rank 6:

$$V^2 = \det \begin{pmatrix} 0 & q_1^2 & q_2^2 & k_1^2 & k_2^2 & 1 \\ q_1^2 & 0 & t & p_1^2 & p_2^2 & 1 \\ q_2^2 & t & 0 & p_3^2 & p_4^2 & 1 \\ k_1^2 & p_1^2 & p_3^2 & 0 & s & 1 \\ k_2^2 & p_2^2 & p_4^2 & s & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 \end{pmatrix}, \quad (2.7)$$

in terms of covariant Mandelstam variables $s = (p_1 + p_2)^2 = (p_3 + p_4)^2$, $t = (p_3 - p_1)^2 = (p_4 - p_2)^2$, $u = (p_4 - p_1)^2 = (p_3 - p_2)^2$. The region of integration, unbounded in the 4-momentum space, is now bounded by the surface where the Jacobian vanishes.

In order to express the operator Op in term of covariant variables, we choose to parametrize it using four operators analogous to the nonrelativistic scalar (O_1), spin-spin (O_s), spin-orbit (O_{so}), tensor (O_t):

$$Op = c_1 O_1 + c_s O_s + c_{so} O_{so} + c_t O_t. \quad (2.8)$$

All of these operators depend only on external momenta and spins:

$$O_1 = \mathbb{1}_1 \mathbb{1}_2, \quad (2.9)$$

$$O_s = \gamma_1^\mu \gamma_{2\mu}, \quad (2.10)$$

$$O_{so} = (\gamma_1 + \gamma_2) \cdot (p_3 - p_4)(\gamma_1 + \gamma_2) \cdot (p_1 - p_2), \quad (2.11)$$

$$O_t = ((\gamma_1 + \gamma_2) \cdot (p_3 - p_4 - p_1 + p_2))^2 - \frac{1}{6}(\gamma_1 + \gamma_2)^2((p_1 - p_2)^2 + (p_3 - p_4)^2). \quad (2.12)$$

It is conventional to include 5 operators but two of these are not kinematically independent [8].

To parametrize Op , we operate with each of the four operators on Op and find the trace of the result. Therefore, we get a system of equations

$$\begin{pmatrix} \text{tr } O_1 & \text{tr } O_s & \text{tr } O_{so} & \text{tr } O_t \\ \text{tr } O_s & \text{tr } O_s^2 & \text{tr } O_s O_{so} & \text{tr } O_s O_t \\ \text{tr } O_{so} & \text{tr } O_{so} O_s & \text{tr } O_{so}^2 & \text{tr } O_{so} O_t \\ \text{tr } O_t & \text{tr } O_t O_s & \text{tr } O_t O_{so} & \text{tr } O_t^2 \end{pmatrix} \begin{pmatrix} c_1 \\ c_s \\ c_{so} \\ c_t \end{pmatrix} = \begin{pmatrix} \text{tr } Op \\ \text{tr } O_s Op \\ \text{tr } O_{so} Op \\ \text{tr } O_t Op \end{pmatrix}. \quad (2.13)$$

We can further separate M to be suitable for computation. First we integrate the continuum variables:

$$C_o = \left(\int \frac{dk_1^2 k_2^2 q_1^2 q_2^2}{(2\pi)^4} \left| \frac{\partial(k_1^0, k_1^x, k_1^y, k_1^z)}{\partial(k_1^2, k_2^2, q_1^2, q_2^2)} \right| c_o D \right). \quad (2.14)$$

Then we can sum over the spin,

$$M = -i \left(\frac{f}{m_\pi} \right)^4 \sum_{o=iso, s, so, t} C_o \bar{u}_1(p_3, s_3) \bar{u}_2(p_4, s_4) O_o u_1(p_1, s_1) u_2(p_2, s_2). \quad (2.15)$$

In the center of mass system where $\vec{p}_1 + \vec{p}_2 = \vec{p}_3 + \vec{p}_4 = 0$, we can decompose C_o as

$$C_o = \sum_{\lambda} C_{o\lambda} P_{\lambda}(\hat{p}_3 \cdot \hat{p}_1), \quad (2.16)$$

where the Legendre polynomial can be factored into spherical harmonics $Y_{\lambda m}(\Omega)$:

$$P_{\lambda}(\hat{p}_1 \cdot \hat{p}_3) = \frac{4\pi}{2\lambda + 1} \sum_{m=-\lambda}^{\lambda} Y_{\lambda m}^*(\hat{p}_3) Y_{\lambda m}(\hat{p}_1). \quad (2.17)$$

Then we can project out the partial wave amplitudes $M_{JLL'}$ using the equation:

$$\begin{aligned} M_{JLL'} &= \sum_{\lambda o} C_{o\lambda} \sum_{s_1, s_2, s_3, s_4} \sum_{m, m'} \int d\hat{p}_3 d\hat{p}_1 Y_{L'm}^*(\hat{p}_3) Y_{Lm'}(\hat{p}_1) \\ &\quad < \frac{1}{2}, s_1; \frac{1}{2}, s_2 | S, m_S > < S', m_{S'} | \frac{1}{2}, s_3; \frac{1}{2}, s_4 > \\ &\quad < J, m_J | L, m; S, m_S > < L', m'; S', m_{S'} | J, m_J > \\ &\quad O(p_1, p_3; s_1, s_2, s_3, s_4) P_{\lambda}(\hat{p}_1 \cdot \hat{p}_3). \end{aligned} \quad (2.18)$$

Then we can calculate the partial wave phase shifts from the relation between the S -matrix and the scattering amplitude M :

$$\delta_{JLL'} \approx e^{i\delta_{JLL'}} \sin \delta_{JLL'} = -\frac{|\vec{p}|}{p_0} \frac{M_{JLL'}}{16\pi}. \quad (2.19)$$

The phase shifts can then be used to construct the cross sections. In this final step the antisymmetrization and related isospin factors have to be carefully included.

The mixing parameter ϵ_J of a triplet state [13] can be found from

$$\epsilon_J = \arccos \sqrt{\frac{1}{1 + \left(\frac{2\delta_{J(J-1)(J+1)}}{\delta_{J(J-1)(J-1)} - \delta_{J(J+1)(J+1)}} \right)^2}}. \quad (2.20)$$

So far, we have treated nucleons and pions as point particles. However, from the standard model, we know that hadrons contain quarks and are extended particles. We dress the vertices with exponential form factors as described in [9]:

$$F_{\pi NN}^2(\Delta_q^2) = (1 + \alpha_1 \Delta_q^2 + \alpha_2 \Delta_q^4 + \alpha_3 \Delta_q^6) e^{-\sigma \Delta_q^4}, \quad (2.21)$$

where $\alpha_1 = 0.8, \alpha_2 = 2.0, \alpha_3 = 2.0, \sigma = 5.2$, and Δ_q^2 is the deviation of the 4-momentum from the mass shell:

$$\Delta_q^2 = q^2 - m^2. \quad (2.22)$$

The parameters were chosen to fit low energy pion-nucleon scattering data.[9]

3. COMPUTATION AND RESULT

In our calculation, we consider the center-of-mass system, where the external nucleons are on shell, *i.e.* $p_1^2 = p_2^2 = p_3^2 = p_4^2 = m_N^2$. We also choose the direction of the incoming nucleon as the z -direction, and the outgoing nucleons as the x - z plane:

$$p_1 = (p_0, 0, 0, |\vec{p}|), \quad (3.1)$$

$$p_2 = (p_0, 0, 0, -|\vec{p}|), \quad (3.2)$$

$$p_3 = (p_0, |\vec{p}| \sin \theta, 0, |\vec{p}| \cos \theta), \quad (3.3)$$

$$p_4 = (p_0, -|\vec{p}| \sin \theta, 0, -|\vec{p}| \cos \theta). \quad (3.4)$$

It is convenient to define some additional variables:

$$p_i = \frac{p_1 - p_2}{2} = (0, 0, 0, |\vec{p}|), \quad (3.5)$$

$$p_f = \frac{p_3 - p_4}{2} = (0, |\vec{p}| \sin \theta, 0, |\vec{p}| \cos \theta), \quad (3.6)$$

$$q = q_1 + q_2. \quad (3.7)$$

The Mandelstam variables are:

$$s = 4(m_N^2 + |\vec{p}|^2), \quad (3.8)$$

$$t = -2|\vec{p}|^2(1 - \cos \theta), \quad (3.9)$$

$$u = -2|\vec{p}|^2(1 + \cos \theta). \quad (3.10)$$

Then, the Jacobian $\left| \frac{\partial(k_1^0, k_1^x, k_1^y, k_1^z)}{\partial(k_1^2, k_2^2, q_1^2, q_2^2)} \right| = \frac{1}{16V}$ can be found from equation (2.7),

$$V^2 = t((k_1^2 - k_2^2)^2(4m_N^2 - s - t) + s(4(m_N^2 - m_\pi^2)^2 + s(4m_\pi^2 - t) - 2(k_1^2 + k_2^2)(2m_N^2 + 2m_\pi^2 - t) + (k_1^2 + k_2^2)^2)). \quad (3.11)$$

In practice, it is convenient to transform k_1^2 and k_2^2 to

$$k_- = k_1^2 - k_2^2, \quad (3.12)$$

$$k_+ = k_1^2 + k_2^2. \quad (3.13)$$

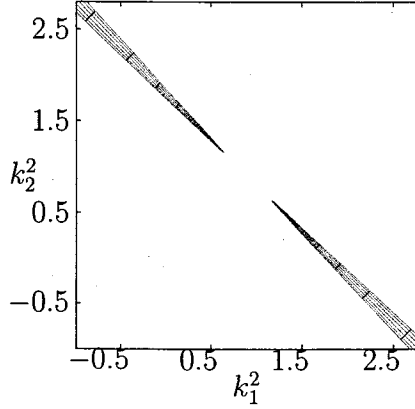


Figure 3.1: Region of integration for $\sqrt{s} = 2m_N + 0.01m_\pi$ at $\cos\theta = 0.5$. The units of k_1^2 and k_2^2 are $(\text{GeV}/c)^2$. The cross hatching illustrates the numerical integration scheme, but the actual grid used was much denser.

V^2 is then quadratic in k_+ ,

$$V^2(k_+, k_-) = Ak_+^2 + Bk_+ + C(k_-), \quad (3.14)$$

where

$$A = st, \quad (3.15)$$

$$B = -2st(2m_N^2 + 2m_\pi^2 - t), \quad (3.16)$$

$$C(k_-) = st(4(m_N^2 - m_\pi^2)^2 + s(4m_\pi^2 - t)) + tk_-^2(4m_N^2 - s - t). \quad (3.17)$$

It is straightforward to find the region where $V^2 > 0$. That defines the integration region where on-shell TPE interactions are kinematically allowed. Numerical examples of the integration region are illustrated in Figure 3.1, 3.2 and 3.3.

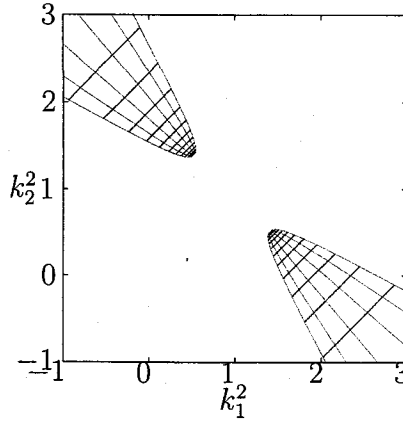


Figure 3.2: Region of integration for $\sqrt{s} = 2m_N + 0.9m_\pi$ at $\cos\theta = 0.5$. The units of k_1^2 and k_2^2 are $(\text{GeV}/c)^2$.

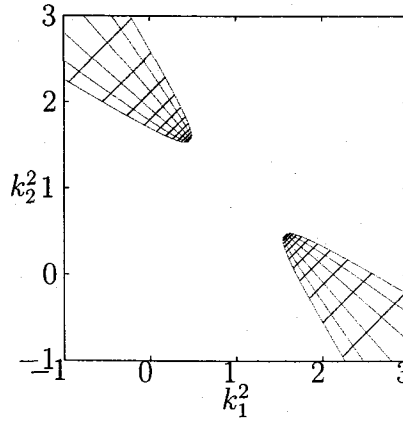


Figure 3.3: Region of integration for $\sqrt{s} = 2m_N + 0.9m_\pi$ at $\cos\theta = 0.054$ (pole). The units of k_1^2 and k_2^2 are $(\text{GeV}/c)^2$.

Next, we parameterize the operator by solving the system of equations (2.13) set up by the traces of the operators, expressed in table 3.1. In calculating the operators' traces, we applied a set of simple rules to a large amount of variables. We found that regular expression substitution [14], whether as in scripting language such as `perl` or `sed` or as an editor's command, was very useful for this kind of manipulation.

Operator	Trace
O_1	16
O_s	0
O_s^2	64
O_{so}	$32 \vec{p} ^2 \cos \theta$
O_{so}^2	$16 \vec{p} ^4(80 \cos^2 \theta + 24)$
$O_s O_{so}$	$128 \vec{p} ^2 \cos \theta$
O_t	$ \vec{p} ^2(-64 \cos \theta + \frac{224}{3})$
O_t^2	$ \vec{p} ^4(3584 \cos^2 \theta + 8704 \cos \theta + \frac{50816}{9})$
$O_t O_s$	$ \vec{p} ^2(-128 \cos \theta + \frac{896}{3})$
$O_t O_{so}$	$\frac{128}{3} \vec{p} ^4(33 \cos^2 \theta - 49 \cos \theta + 3)$
O_p	$16m_N^2(q_1 \cdot q_2)^2$
$O_s O_p$	$\text{tr } \gamma^\mu \not{q}_2 \not{k}_1 \not{q}_1 \text{tr } \gamma_\mu \not{q}_2 \not{k}_2 \not{q}_1$
$O_{so} O_p$	$4(2m_N^2 q_1 \cdot q_2 \text{tr } \not{p}_f \not{p}_i \not{q}_1 \not{q}_2 + \text{tr } \not{p}_f \not{q}_2 \not{k}_1 \not{q}_1 \text{tr } \not{p}_i \not{q}_2 \not{k}_2 \not{q}_1 + \text{tr } \not{p}_i \not{q}_2 \not{k}_1 \not{q}_1 \text{tr } \not{p}_f \not{q}_2 \not{k}_2 \not{q}_1)$
$O_t O_p$	$(8q^2 + \frac{8}{3} \vec{p} ^2)m_N^2(q_1 \cdot q_2)^2$ $+ 8\text{tr } \not{q}_2 \not{k}_1 \not{q}_1 \text{tr } \not{q}_2 \not{k}_2 \not{q}_1 + \frac{8}{3} \vec{p} ^2 \text{tr } \gamma^\mu \not{q}_2 \not{k}_1 \not{q}_1 \text{tr } \gamma_\mu \not{q}_2 \not{k}_2 \not{q}_1$

TABLE 3.1: Traces of operators in center-of-mass system.

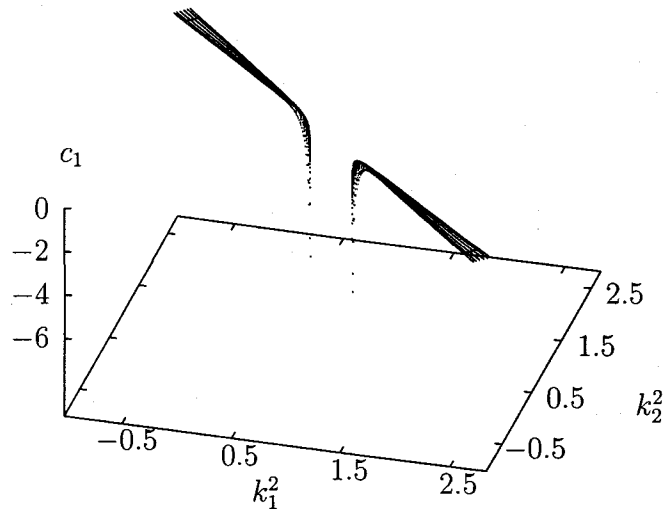


Figure 3.4: Amplitude c_1 as function of k_1^2 and k_2^2 : $\sqrt{s} = 2m_N + 0.01m_\pi$, $\cos \theta = 0.5$.

However, the expressions we get from substituting all covariant variables are too long. It is impractical to solve them analytically, even with the aid of a symbolic manipulation program. So, we solve the linear system numerically for each combination $(k_1^2, k_2^2, \cos \theta)$. [15] Then, after we get the operators' amplitudes, we integrate over $dk_1^2 dk_2^2$ to find the contribution at a particular scattering angle.

Examples of the integrands are shown in Figure 3.4 through 3.15. The contributions over k_1^2, k_2^2 show two main characteristics of exponential falloff, and divergences at the boundary of integration. The exponential characteristic comes from form factors in our formula. The divergences at the boundaries come from the Jacobian of the transformation in equation (2.7). These divergences are integrable, leading to finite integrals for most values of $\cos \theta$. Integration intervals were chosen to ensure a precision much better than 1 percent.

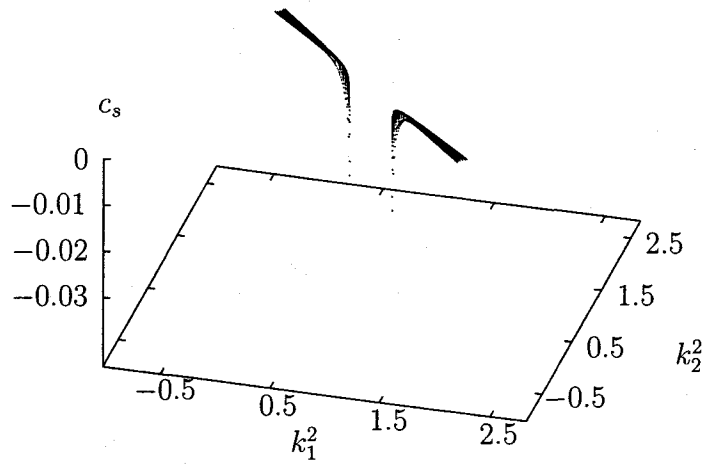


Figure 3.5: Amplitude c_s as function of k_1^2 and k_2^2 : $\sqrt{s} = 2m_N + 0.01m_\pi$, $\cos\theta = 0.5$.

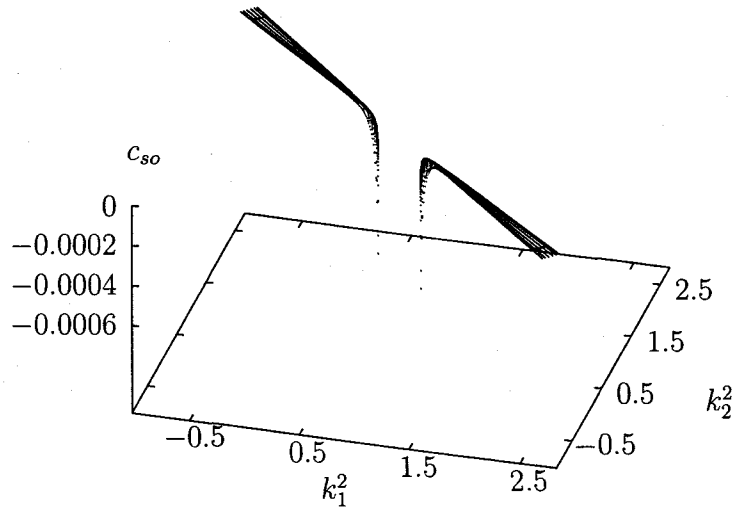


Figure 3.6: Amplitude c_{so} as function of k_1^2 and k_2^2 : $\sqrt{s} = 2m_N + 0.01m_\pi$, $\cos\theta = 0.5$.

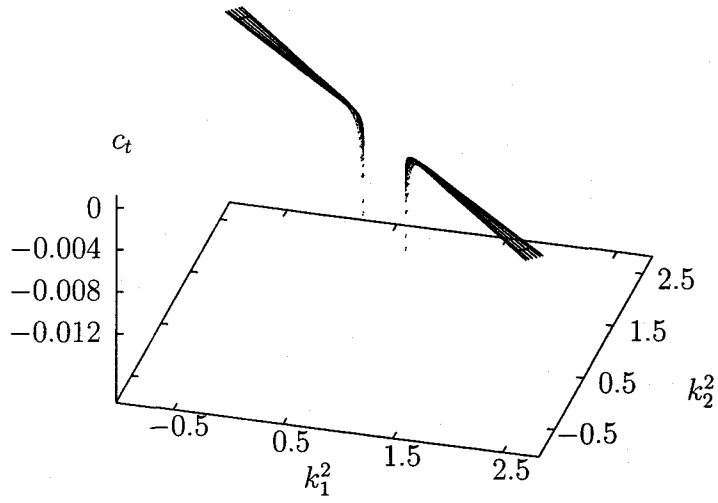


Figure 3.7: Amplitude c_t as function of k_1^2 and k_2^2 : $\sqrt{s} = 2m_N + 0.01m_\pi$, $\cos\theta = 0.5$.

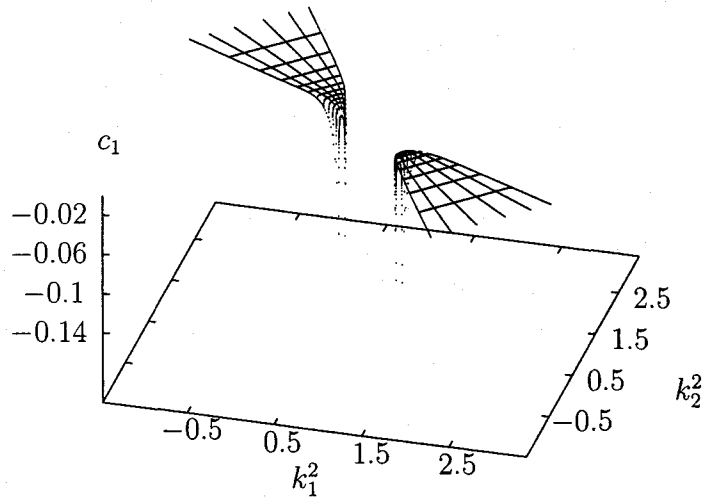


Figure 3.8: Amplitude c_1 as function of k_1^2 and k_2^2 : $\sqrt{s} = 2m_N + 0.9m_\pi$, $\cos\theta = 0.5$.

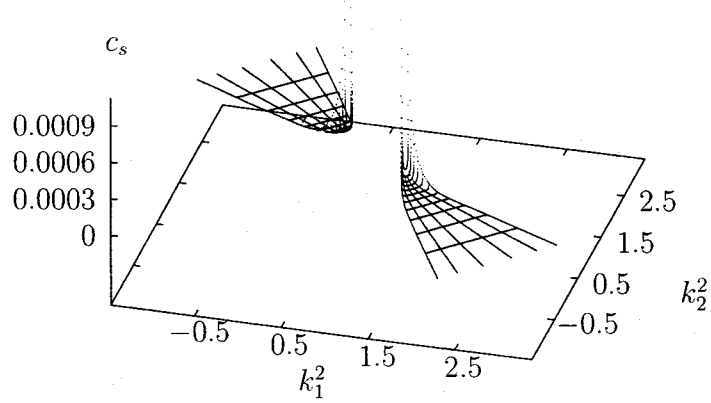


Figure 3.9: Amplitude c_s as function of k_1^2 and k_2^2 : $\sqrt{s} = 2m_N + 0.9m_\pi$, $\cos\theta = 0.5$.

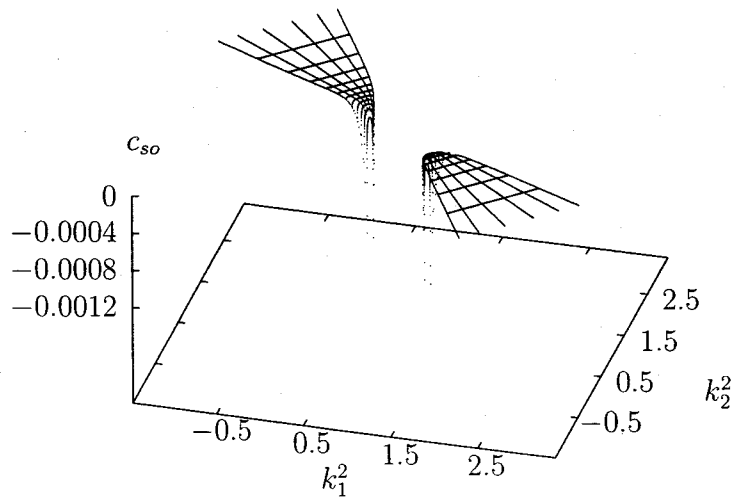


Figure 3.10: Amplitude c_{so} as function of k_1^2 and k_2^2 : $\sqrt{s} = 2m_N + 0.9m_\pi$, $\cos\theta = 0.5$.

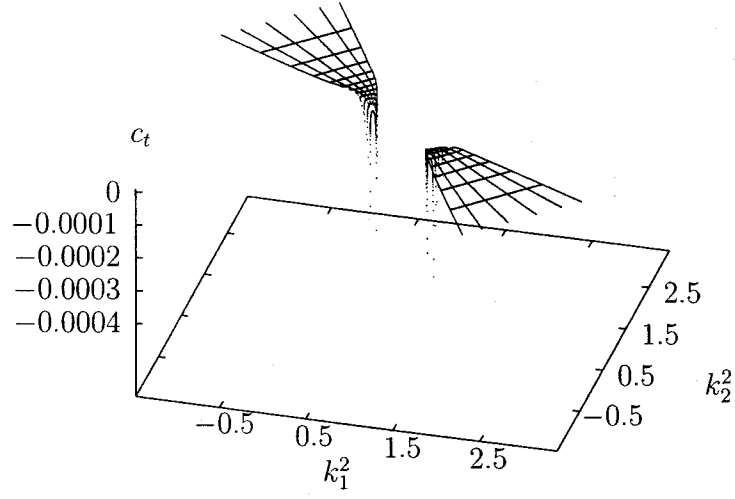


Figure 3.11: Amplitude c_t as function of k_1^2 and k_2^2 : $\sqrt{s} = 2m_N + 0.9m_\pi$, $\cos \theta = 0.5$.

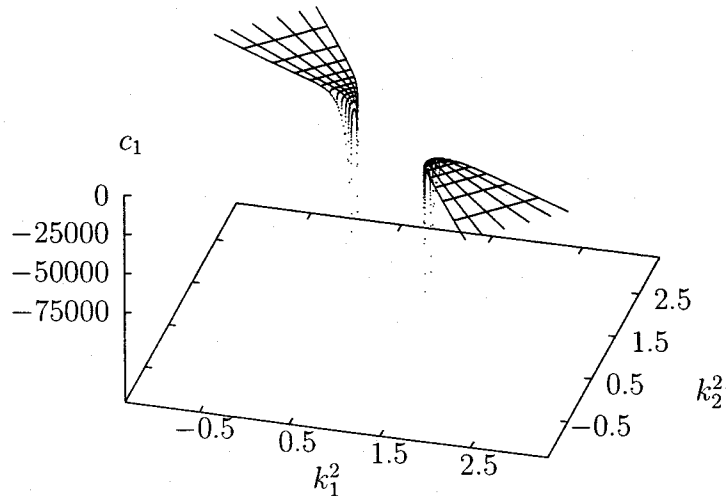


Figure 3.12: Amplitude c_1 as function of k_1^2 and k_2^2 : $\sqrt{s} = 2m_N + 0.9m_\pi$, $\cos \theta = 0.054$ (pole).

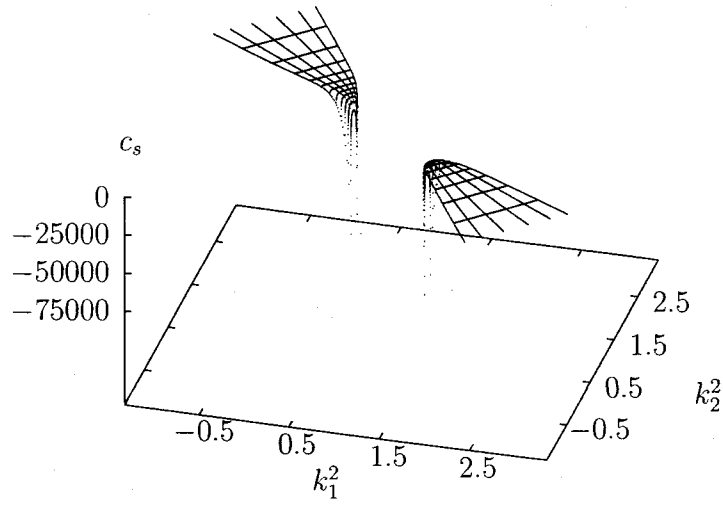


Figure 3.13: Amplitude c_s as function of k_1^2 and k_2^2 : $\sqrt{s} = 2m_N + 0.9m_\pi$, $\cos \theta = 0.054$ (pole).

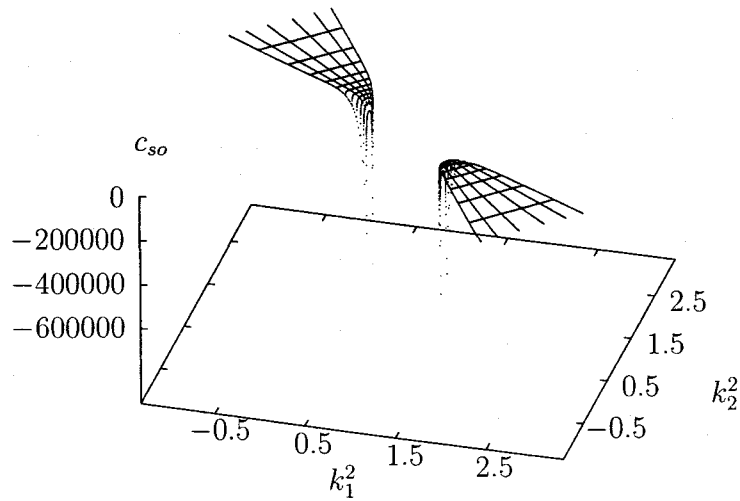


Figure 3.14: Amplitude c_{so} as function of k_1^2 and k_2^2 : $\sqrt{s} = 2m_N + 0.9m_\pi$, $\cos \theta = 0.054$ (pole).

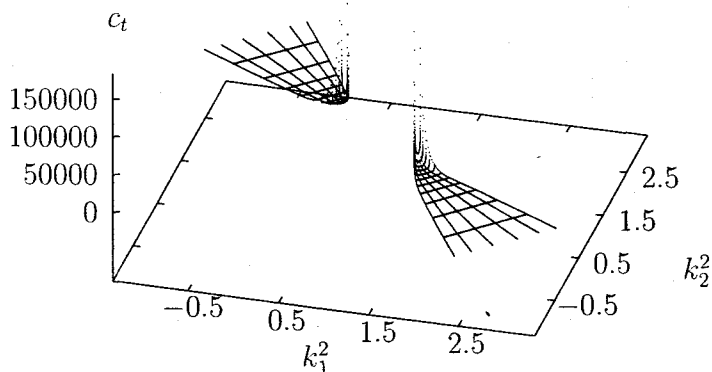


Figure 3.15: Amplitude c_t as function of k_1^2 and k_2^2 : $\sqrt{s} = 2m_N + 0.9m_\pi$, $\cos \theta = 0.054$ (pole).

Contributions from the operators are not smooth functions of $\cos \theta$ as we anticipated, as shown in Figure 3.16 through Figure 3.23 and especially by the enlarged plots near the poles' vicinity in Figure 3.24 through Figure 3.31. These poles occurred in every operator's contribution although they were not prominent at low energy, especially for scalar and spin-spin operators (Figure 3.16 and Figure 3.17). Closer investigation (Figure 3.24 and Figure 3.25) revealed poles at the same locations where they occurred in the rest of the operators. We had to understand the reason for these poles before deciding how to treat them numerically.

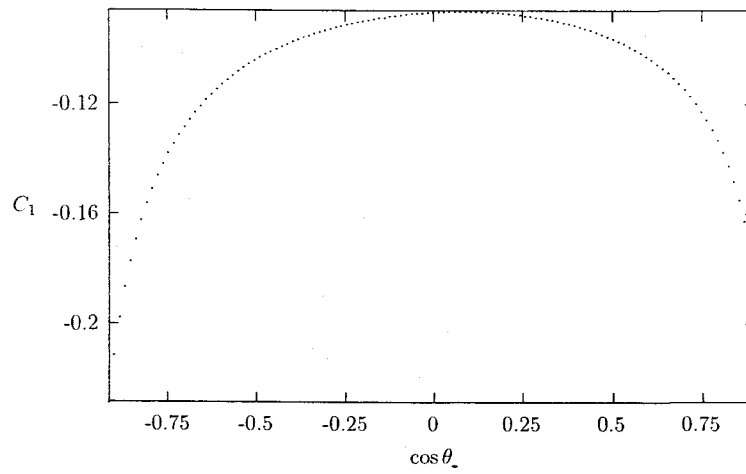


Figure 3.16: Contribution from scalar operator as a function of $\cos \theta$ for $\sqrt{s} = 2m_N + 0.01m_\pi$.

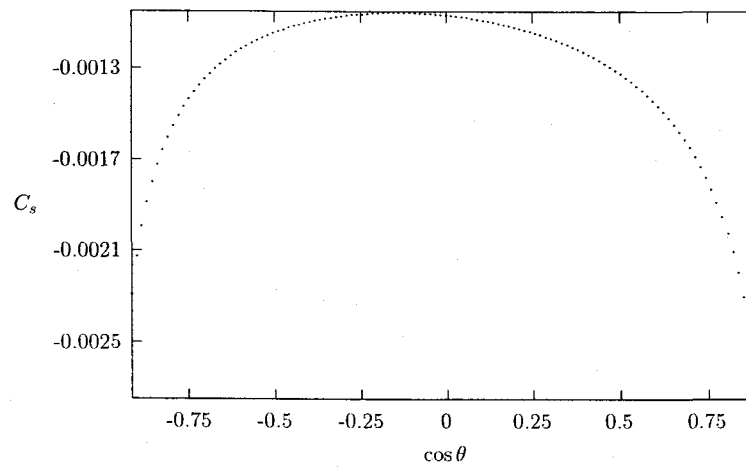


Figure 3.17: Contribution from spin-spin operator as a function of $\cos \theta$ for $\sqrt{s} = 2m_N + 0.01m_\pi$.

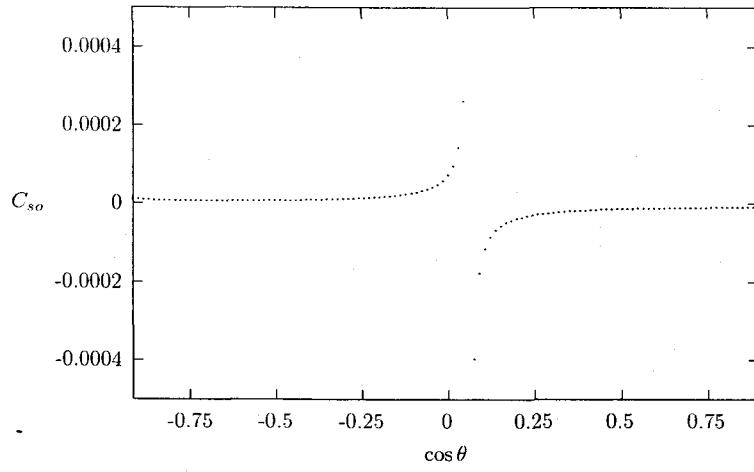


Figure 3.18: Contribution from spin-orbit operator as a function of $\cos \theta$ for $\sqrt{s} = 2m_N + 0.01m_\pi$.

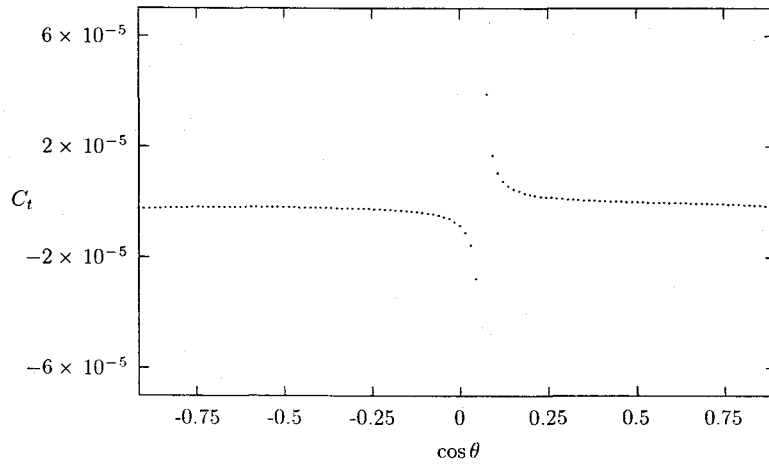


Figure 3.19: Contribution from tensor operator as a function of $\cos \theta$ for $\sqrt{s} = 2m_N + 0.01m_\pi$.

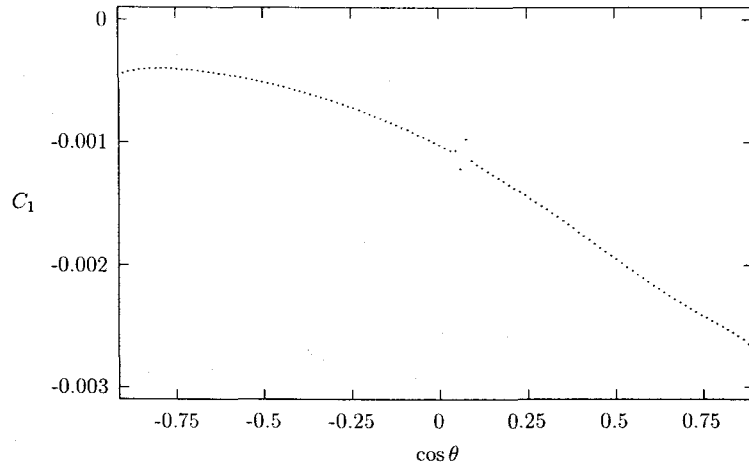


Figure 3.20: Contribution from scalar operator as a function of $\cos \theta$ for $\sqrt{s} = 2m_N + 0.9m_\pi$.

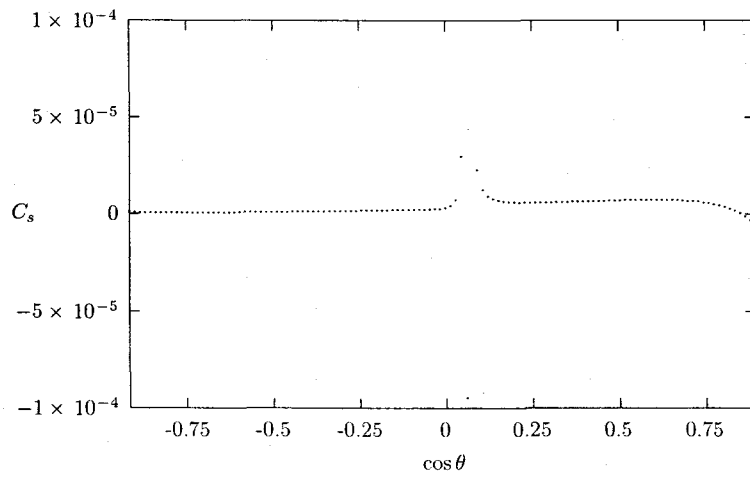


Figure 3.21: Contribution from spin-spin operator as a function of $\cos \theta$ for $\sqrt{s} = 2m_N + 0.9m_\pi$.

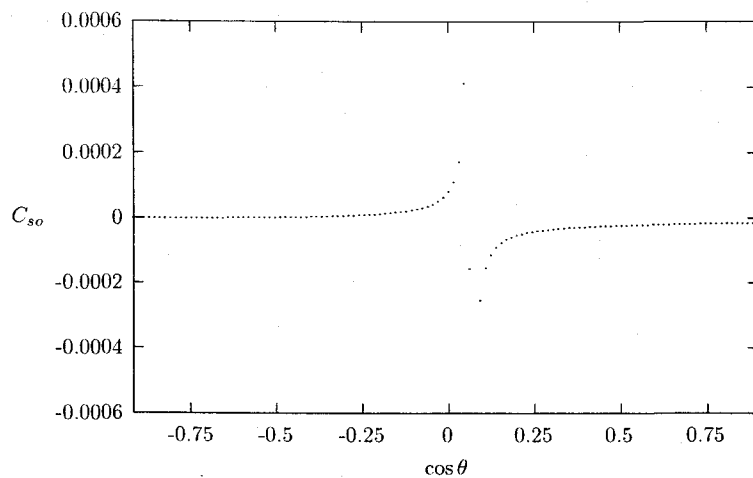


Figure 3.22: Contribution from spin-orbit operator as a function of $\cos \theta$ for $\sqrt{s} = 2m_N + 0.9m_\pi$.

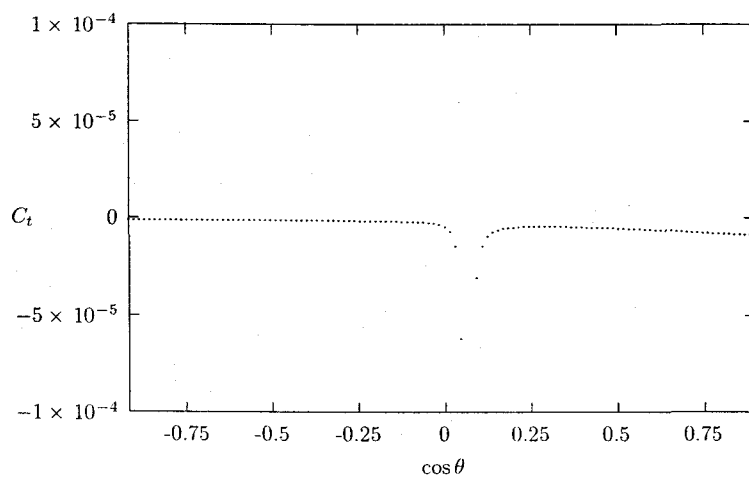


Figure 3.23: Contribution from tensor operator as a function of $\cos \theta$ for $\sqrt{s} = 2m_N + 0.9m_\pi$.

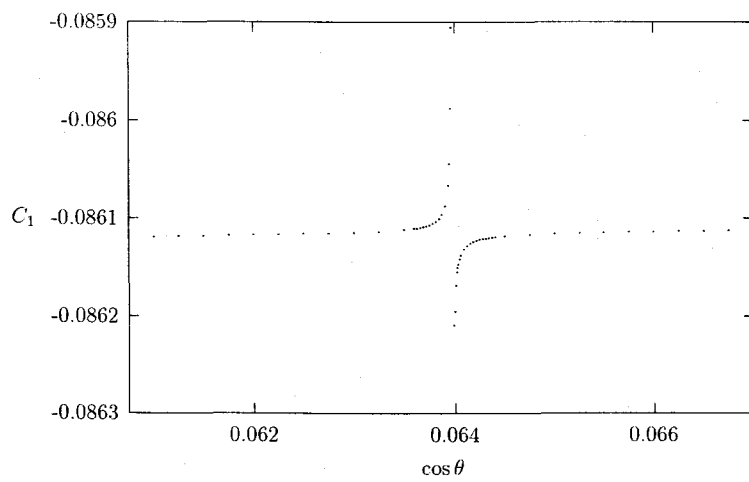


Figure 3.24: Poles' vicinity of Figure 3.16.

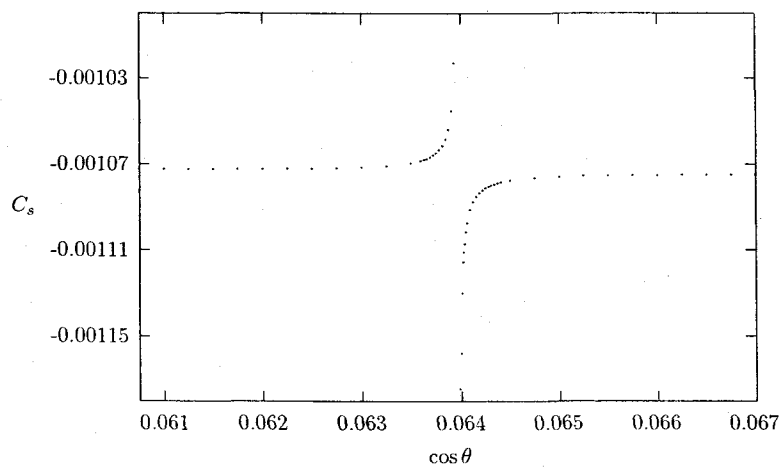


Figure 3.25: Poles' vicinity of Figure 3.17.

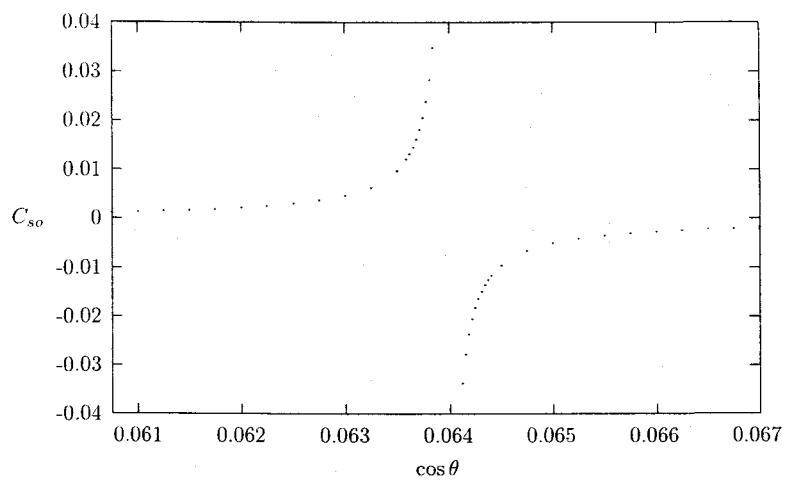


Figure 3.26: Poles' vicinity of Figure 3.18.

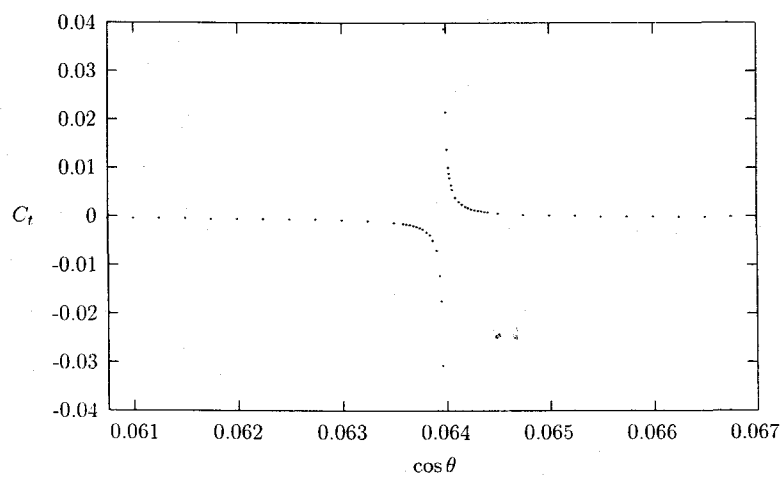


Figure 3.27: Poles' vicinity of Figure 3.19.

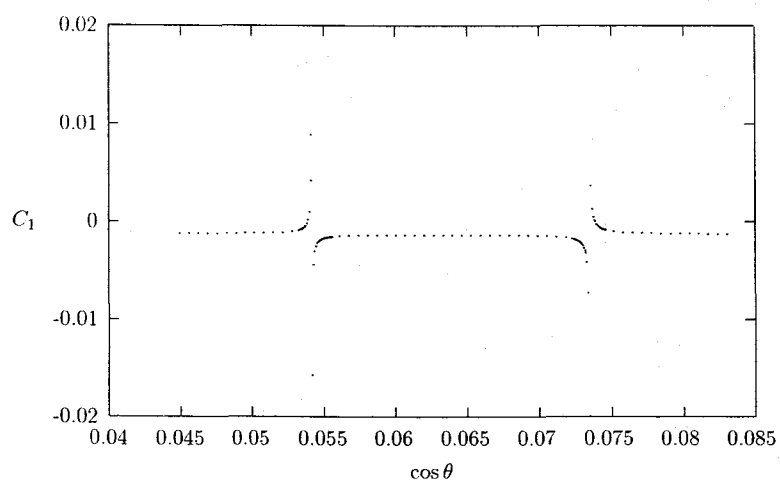


Figure 3.28: Poles' vicinity of Figure 3.20.

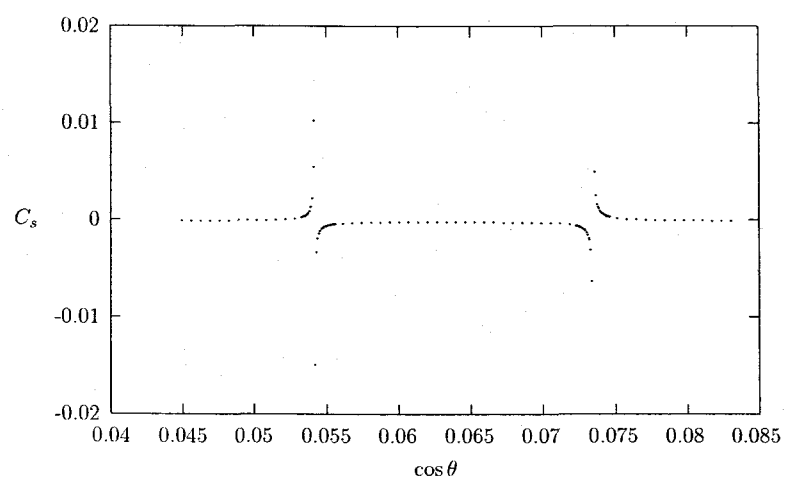


Figure 3.29: Poles' vicinity of Figure 3.21.

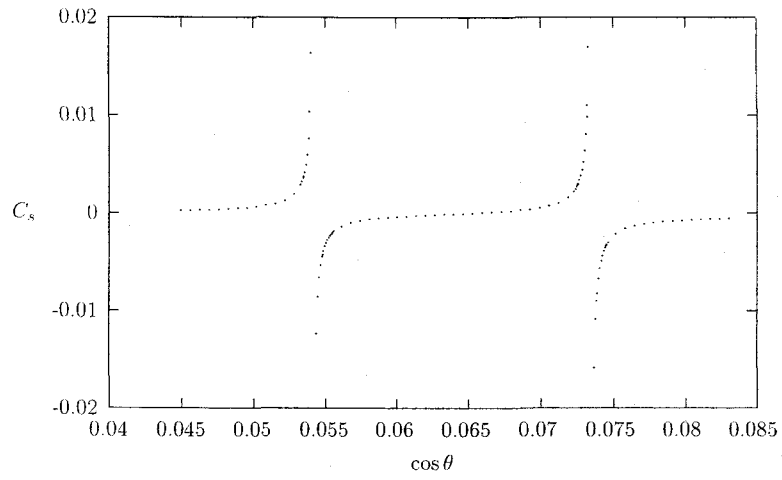


Figure 3.30: Poles' vicinity of Figure 3.22.

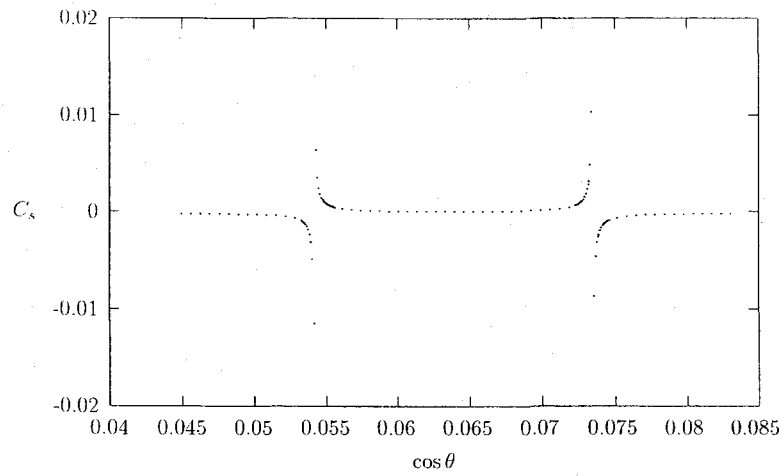


Figure 3.31: Poles' vicinity of Figure 3.23.

We would like to point out that these singularities were not due to the divergences at the boundaries of the integration. Figure 3.12 through 3.15 are operators' amplitudes at a pole. Their distributions do not differ greatly from Figure 3.8 through 3.11, which are operators' amplitudes at other $\cos \theta$ for the same energy. The differences are the sizes of the amplitudes. To understand these unexpected poles, we studied the eigenvalues and eigenvectors of the 4-dimensional spin-operator space.

From Figure 3.28 – 3.31, and Table 3.2, poles are located at $\cos \theta = 0.0541$ and $\cos \theta = 0.0734$ for $\sqrt{s} = 2m_N + 0.9m_\pi$. We can see that one of eigenvalues is zero in both poles. So, the determinant of the system we use to parameterize the operator is zero. Physically, this means that there is an ambiguity in the parameterization by operators at a particular angle. From linear algebra, we know that the solution of a system of equations is inversely proportional to the determinant of the system. For a system of zero determinant, the answer is not well-defined.

The equation of the zero eigenvalue is a homogeneous equation. We can change any of the components as long as the combination gives zero eigenvalue. Specifically, from table 3.2, we see that the eigenvectors of zero eigenvalues at the poles consist mainly of the spin-orbit operator. This implies an ambiguity in the amplitude of the spin-orbit operator at the poles.

Ambiguities are familiar in the partial wave analysis of scattering.[16] What we encounter seems to be another kind of ambiguity.

To treat the poles numerically we observe that they appear as localized irregularities on an otherwise smooth distribution. Since the singularities are first order we expect the net contribution to be small. We compare the results from two numerical integration methods, trapezoidal integration and Gaussian quadrature. [17] The result is shown in Table 3.3. The two dissimilar integration methods give quite similar results. Therefore, we believe that the

$\cos \theta$	eigenvalues	eigenvectors
0.0541	15.047	(-0.974, -0.077, 0.180, 0.106)
	113.323	(0.069, 0.565, 0.145, 0.808)
	42.162	(0.165, -0.809, 0.243, 0.508)
	0.000	(0.133, 0.136, 0.941, -0.276)
0.0734	15.301	(-0.974, -0.057, -0.196, 0.089)
	114.200	(0.068, 0.553, -0.130, 0.819)
	43.553	(-0.146, 0.815, 0.258, -0.497)
	0.000	(0.155, 0.159, -0.936, -0.269)

TABLE 3.2: Eigenvalues and eigenvectors for $\sqrt{s} = 2m_N + 0.9m_\pi$.

pole contributions are reasonably well estimated by both methods. However, trapezoidal integration is not convenient because it requires exact location of the poles so that we can remove small regions around the poles symmetrically. On the contrary, noticing that the poles always locate near $\cos \theta = 0$, we can avoid the poles by using even-order Gaussian quadrature whose Gaussian points lie far from zero. Therefore, we choose Gaussian quadrature in our study. It is not only more convenient but it is less sensitive to the poles.

The numerical results from our calculation are shown in Table 3.4 and Table 3.5. All the amplitudes are very small, corresponding to phase shifts much less than 1 degree. By contrast, the observed phase shifts are many degrees, nearly resonant in the $l = 0$ channels. OPE and off-shell TPE are similar to the observed amplitudes and many orders of magnitude larger than the on-shell TPE. In all channels, the contributions from on-shell TPE are

Method of integration	δ_{011}	δ_{100}	δ_{122}	δ_{102}	ϵ_1
Trapezoidal	5.40×10^{-5}	1.54×10^{-4}	3.20×10^{-6}	4.88×10^{-7}	6.48×10^{-3}
Gaussian	4.78×10^{-5}	1.44×10^{-4}	9.72×10^{-7}	4.52×10^{-7}	6.34×10^{-3}

TABLE 3.3: Comparison between methods of integration for the case $\sqrt{s} = 2m_N + 0.9m_\pi$.

negligibly small. In view of these we did not bother to refine our numerical estimates any further.

We only carried out the partial wave decompositions for the TPE ladder diagram. However, we did calculate the momentum-space spinor amplitudes for both direct and crossed diagrams shown in Figure 2.1. We found that both diagrams gave comparable amplitudes. Since these amplitudes were so small we feel confident that the effects of on-shell pion exchange are unimportant in nuclear scattering.

We also would like to know how the form factor affected our calculation. We hypothesized that the form factor's cut-off would lead to small on-shell contributions of two pion exchange. However, as seen in Figure 3.32, the contributions with form factor = 1 are still very small, even though they are much larger than with a realistic form factor. We conclude that kinematics alone are sufficient to make on-shell two-pion exchange small.

To illustrate the latter point, we consider the distribution of the operators' amplitude as shown in Figure 3.33 through 3.36. In the kinematic region of integration, the form factor is not needed to make the integrals converge. In scattering, k_1^2 and k_2^2 are far from singularities of on-shell poles.

s	δ_{011}	δ_{100}	δ_{122}	δ_{102}	ϵ_1
3.58	-1.28×10^{-4}	7.95×10^{-4}	3.32×10^{-5}	-1.09×10^{-8}	2.85×10^{-5}
3.63	-1.13×10^{-4}	6.65×10^{-4}	2.33×10^{-5}	9.36×10^{-8}	2.92×10^{-4}
3.68	-7.26×10^{-5}	5.53×10^{-4}	1.53×10^{-5}	2.37×10^{-7}	8.83×10^{-4}
3.74	-3.12×10^{-5}	4.50×10^{-4}	9.29×10^{-6}	3.71×10^{-7}	1.68×10^{-3}
3.79	2.51×10^{-6}	3.62×10^{-4}	5.45×10^{-6}	4.67×10^{-7}	2.62×10^{-3}
3.84	2.66×10^{-5}	2.89×10^{-4}	3.41×10^{-6}	5.20×10^{-7}	3.63×10^{-3}
3.90	4.20×10^{-5}	2.32×10^{-4}	2.67×10^{-6}	5.34×10^{-7}	4.65×10^{-3}
3.95	5.04×10^{-5}	1.88×10^{-4}	2.72×10^{-6}	5.20×10^{-7}	5.61×10^{-3}
4.00	5.40×10^{-5}	1.54×10^{-4}	3.20×10^{-6}	4.88×10^{-7}	6.48×10^{-3}
4.06	5.43×10^{-5}	1.27×10^{-4}	3.81×10^{-6}	4.48×10^{-7}	7.24×10^{-3}
4.11	5.26×10^{-5}	1.07×10^{-4}	4.41×10^{-6}	4.06×10^{-7}	7.90×10^{-3}
4.17	4.99×10^{-5}	9.10×10^{-5}	4.90×10^{-6}	3.66×10^{-7}	8.50×10^{-3}
4.22	4.67×10^{-5}	7.82×10^{-5}	5.25×10^{-6}	3.33×10^{-7}	9.12×10^{-3}
4.28	4.33×10^{-5}	6.79×10^{-5}	5.45×10^{-6}	3.08×10^{-7}	9.86×10^{-3}
4.33	3.99×10^{-5}	5.94×10^{-5}	5.52×10^{-6}	2.94×10^{-7}	1.09×10^{-2}
4.39	3.68×10^{-5}	5.23×10^{-5}	5.48×10^{-6}	2.94×10^{-7}	1.26×10^{-2}
4.45	3.38×10^{-5}	4.63×10^{-5}	5.34×10^{-6}	3.10×10^{-7}	1.51×10^{-2}
4.50	3.10×10^{-5}	4.12×10^{-5}	5.12×10^{-6}	3.45×10^{-7}	1.91×10^{-2}
4.56	2.84×10^{-5}	3.68×10^{-5}	4.84×10^{-6}	4.02×10^{-7}	2.51×10^{-2}

TABLE 3.4: Phase shift from TPE for the case of $j = 0, 1$ in unit of degree.

s	δ_{211}	δ_{233}	δ_{213}	ϵ_2
3.58	-1.10×10^{-4}	-2.92×10^{-6}	-2.22×10^{-8}	4.14×10^{-4}
3.63	-9.68×10^{-5}	-2.72×10^{-6}	-6.23×10^{-8}	1.33×10^{-3}
3.68	-6.22×10^{-5}	-2.07×10^{-6}	-1.12×10^{-7}	3.71×10^{-3}
3.74	-2.68×10^{-5}	-1.49×10^{-6}	-1.61×10^{-7}	1.27×10^{-2}
3.79	2.13×10^{-6}	-1.09×10^{-6}	-2.10×10^{-7}	1.30×10^{-1}
3.84	2.28×10^{-5}	-8.30×10^{-7}	-2.62×10^{-7}	2.22×10^{-2}
3.90	3.59×10^{-5}	-6.51×10^{-7}	-3.16×10^{-7}	1.73×10^{-2}
3.95	4.32×10^{-5}	-4.89×10^{-7}	-3.72×10^{-7}	1.71×10^{-2}
4.00	4.62×10^{-5}	-3.10×10^{-7}	-4.25×10^{-7}	1.83×10^{-2}
4.06	4.65×10^{-5}	-1.03×10^{-7}	-4.68×10^{-7}	2.01×10^{-2}
4.11	4.51×10^{-5}	1.23×10^{-7}	-4.96×10^{-7}	2.21×10^{-2}
4.17	4.27×10^{-5}	3.52×10^{-7}	-5.05×10^{-7}	2.38×10^{-2}
4.22	4.00×10^{-5}	5.68×10^{-7}	-4.92×10^{-7}	2.50×10^{-2}
4.28	3.71×10^{-5}	7.56×10^{-7}	-4.57×10^{-7}	2.52×10^{-2}
4.33	3.42×10^{-5}	9.06×10^{-7}	-4.01×10^{-7}	2.40×10^{-2}
4.39	3.15×10^{-5}	1.01×10^{-6}	-3.24×10^{-7}	2.12×10^{-2}
4.45	2.90×10^{-5}	1.07×10^{-6}	-2.27×10^{-7}	1.63×10^{-2}
4.50	2.66×10^{-5}	1.09×10^{-6}	-1.10×10^{-7}	8.61×10^{-3}
4.56	2.44×10^{-5}	1.07×10^{-6}	2.89×10^{-8}	2.48×10^{-3}

TABLE 3.5: Phase shift from TPE for the case of $j = 2$.

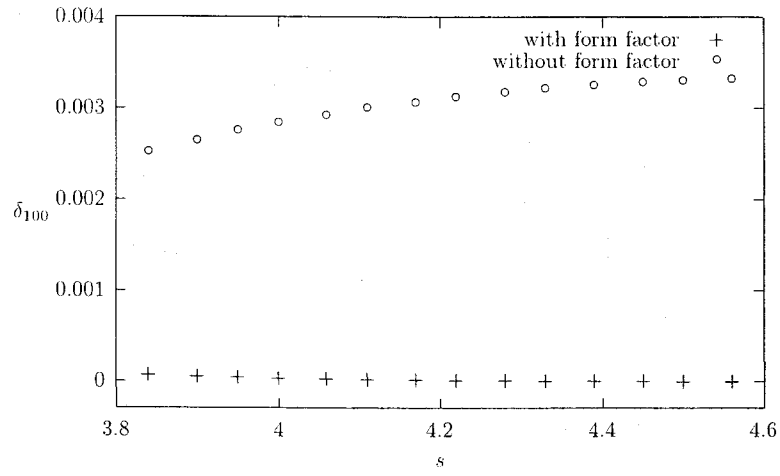


Figure 3.32: Effect of form factor on partial wave δ_{100} .

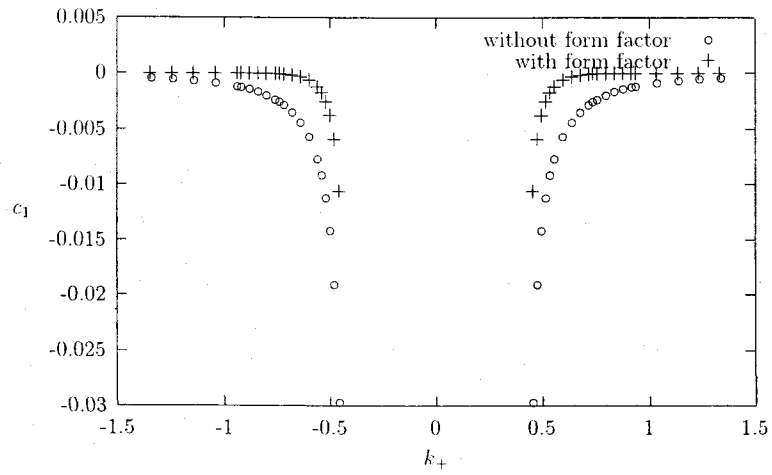


Figure 3.33: Contribution of scalar operator for $\sqrt{s} = 2m_N + 0.9m_\pi$, at $\cos \theta = 0.5$ and where k_- is at the kinematic limit, $V(k_+, k_-) = 0$. See equation (3.12) through equation (3.17).

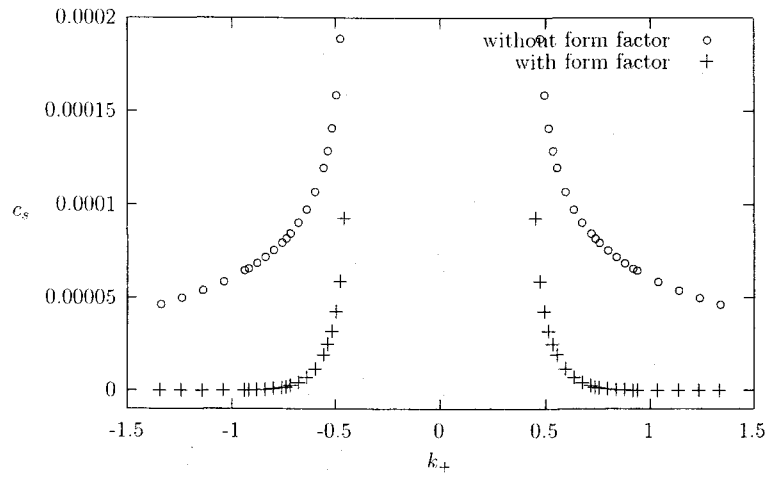


Figure 3.34: Contribution of spin-spin operator for the same condition as Figure 3.33.

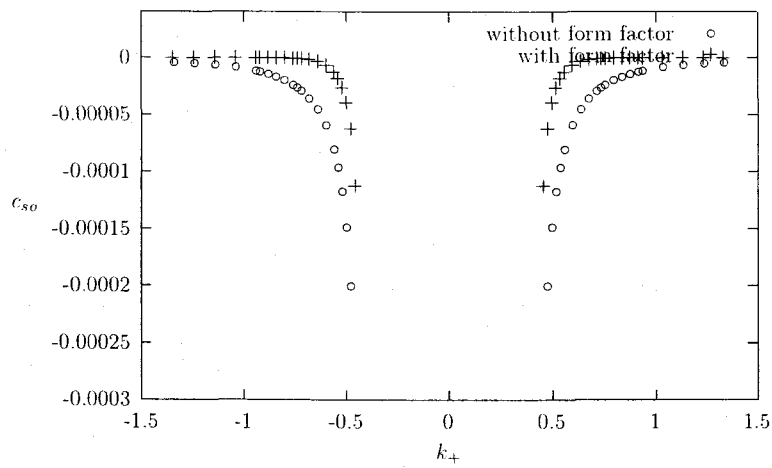


Figure 3.35: Contribution of spin-orbit operator for the same condition as Figure 3.33.

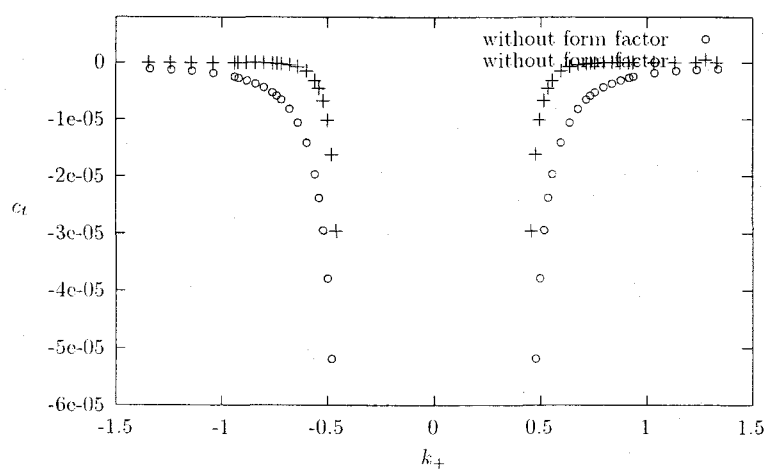


Figure 3.36: Contribution of tensor operator for the same condition as Figure 3.33.

4. CONCLUSION

The overall contribution from TPE ladders is small and smooth as expected. The contribution is negligible compared with one pion exchange and two pion exchange with on-shell nucleons.

There were ambiguities in parameterizing spin operators at particular angles. These ambiguities gave rise to the poles in spin operators' distributions over $\cos\theta$. However, the partial wave amplitudes calculated by integrating over $\cos\theta$ are finite and well defined, as are the spin-projection amplitudes at each angle. The fact that the spin-operator decomposition of the scattering amplitude is not always unique is our only truly surprising result.

BIBLIOGRAPHY

- [1] K. A. Brueckner and K. M. Watson, *Physical Review* **90**, 699-708 (1953).
- [2] L. C. Gomes, J. D. Walecka, and V. F. Weisskopf, *Annals of Physics* **3**, 241-274 (1958).
- [3] G. Brown and A. Jackson, *The Nucleon-Nucleon Interaction* (North-Holland, 1976).
- [4] R. V. Mau, in *Mesons in Nuclei, vol. 1*, edited by M. Rho and D. Wilkinson (North-Holland, 1979), p. 151.
- [5] M. H. Partovi and E. L. Lomon, *Physical Review D* **2**(9), 1999-2032 (1970).
- [6] R. Machleidt and G. Q. Li, *Physics Reports* **242**, 5-35 (1994).
- [7] R. Machleidt, K. Holinde, and Ch. Elster, *Physics Reports* **149**, 1-89 (1987).
- [8] J. Hamilton and G. C. Oades, *Nuclear Physics A* **424**, 447-483 (1984).
- [9] H. Wu, *Self-consistent Relativistic Model for Pion-nucleon Scattering*, Ph.D. thesis, Oregon State University (1994).
- [10] F. Gross, *Relativistic Quantum Mechanics* (John Wiley & Sons, 1993).
- [11] P. J. Siemens, M. Soyeur, G. D. White, L. J. Lantto, and K. T. R. Davies, *Physical Review C* **40**, 2641-2671 (1989).
- [12] P. J. Siemens and A. P. Vischer, *Annals of Physics* **238**(1), 167-199 (1995).
- [13] J. M. Blatt and L. C. Biedenharn, *Physical Review* **85**(3), 399-404 (1952).

- [14] J. E. Friedl, *Mastering Regular Expression* (O'Reilly & Associates, 1997).
- [15] C. W. Ueberhuber, *Numerical Computation 2* (Springer-Verlag, 1997).
- [16] V. G. J. Stoks, R. A. M. Klomp, M. C. M. Rentmesster, and J. J. de Swart, *Physical Review C* **48**(2), 792-814 (1993).
- [17] J. R. Rice, *Numerical Methods, Software, and Analysis* (Academic Press, 1993).
- [18] W. H. Press, S. A. Tenkolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes in C: The Art of Scientific Computing* (Cambridge University Press, 1993).
- [19] B. W. Kernighan and D. M. Ritchie, *The C Programming Language* (Prentice Hall, 1988).

APPENDICES

APPENDIX A: TRACE OF OPERATORS

We collect here the trace algebra used in our calculations:

$$\text{tr } \not{a} \not{b} = 4(a \cdot b), \quad (\text{A1})$$

$$\text{tr } \not{a} \not{b} \not{c} \not{d} = 4((a \cdot b)(c \cdot d) - (a \cdot c)(b \cdot d) + (b \cdot c)(a \cdot d)), \quad (\text{A2})$$

$$\begin{aligned} \text{tr } \gamma^\mu \not{a} \not{b} \not{c} \text{tr } \gamma_\mu \not{d} \not{e} \not{f} = 16 & ((a \cdot b)(c \cdot d)(e \cdot f) - (a \cdot b)(e \cdot c)(d \cdot f) \\ & + (a \cdot b)(f \cdot c)(d \cdot e) - (a \cdot c)(b \cdot d)(e \cdot f) \\ & + (a \cdot c)(b \cdot e)(d \cdot f) - (a \cdot c)(b \cdot f)(d \cdot e) \\ & + (b \cdot c)(a \cdot d)(e \cdot f) - (b \cdot c)(a \cdot e)(d \cdot f) \\ & + (b \cdot c)(a \cdot f)(d \cdot e)). \end{aligned} \quad (\text{A3})$$

Below are explicit form of traces of operators in Table 3.1:

$$\begin{aligned} \text{tr } O_s Op = (q_1 \cdot q_2)((k_1 \cdot k_2)(q_1 \cdot q_2) - (q_1 \cdot k_2)(q_2 \cdot k_1) - (q_1 \cdot k_1)(q_2 \cdot k_2)) \\ + m_\pi^2((q_1 \cdot k_1)(q_1 \cdot k_2) + (q_2 \cdot k_1)(q_2 \cdot k_2)), \end{aligned} \quad (\text{A4})$$

$$\begin{aligned} \text{tr } O_{so} Op = & 2m_N^2(q_1 \cdot q_2) \\ & \times (((p_1 \cdot p_3) + (p_2 \cdot p_4) - (p_1 \cdot p_4) - (p_2 \cdot p_3))(q_1 \cdot q_2) \\ & - ((q_2 \cdot p_3) - (q_2 \cdot p_4))((q_1 \cdot p_1) - (q_1 \cdot p_2)) \\ & + ((q_1 \cdot p_3) - (q_1 \cdot p_4))((q_2 \cdot p_1) - (q_2 \cdot p_2))) \\ & + ((k_2 \cdot p_1)(q_1 \cdot q_2) - (k_2 \cdot p_2)(q_1 \cdot q_2) - (q_1 \cdot p_1)(q_2 \cdot k_2) \\ & + (q_1 \cdot p_2)(q_2 \cdot k_2) - (q_1 \cdot k_2)(q_2 \cdot p_1) + (q_1 \cdot k_2)(q_2 \cdot p_2)) \quad (\text{A5}) \\ & \times ((k_1 \cdot p_3)(q_1 \cdot q_2) - (k_1 \cdot p_4)(q_1 \cdot q_2) - (q_1 \cdot p_3)(q_2 \cdot k_1) \\ & + (q_1 \cdot p_4)(q_2 \cdot k_1) - (q_1 \cdot k_1)(q_2 \cdot p_3) + (q_1 \cdot k_1)(q_2 \cdot p_4)) \\ & + ((k_1 \cdot p_1)(q_1 \cdot q_2) - (k_1 \cdot p_2)(q_1 \cdot q_2) - (q_1 \cdot p_1)(q_2 \cdot k_1) \\ & + (q_1 \cdot p_2)(q_2 \cdot k_1) - (q_1 \cdot k_1)(q_2 \cdot p_1) + (q_1 \cdot k_1)(q_2 \cdot p_2)) \\ & \times ((k_2 \cdot p_3)(q_1 \cdot q_2) - (k_2 \cdot p_4)(q_1 \cdot q_2) - (q_1 \cdot p_3)(q_2 \cdot k_2) \\ & + (q_1 \cdot p_4)(q_2 \cdot k_2) - (q_1 \cdot k_2)(q_2 \cdot p_3) + (q_1 \cdot k_2)(q_2 \cdot p_4)). \end{aligned}$$

$$\begin{aligned}
\text{tr } O_t O p &= (8(q_1 + q_2)^2 + \frac{8}{3}|\vec{p}|^2)m_N^2(q_1 \cdot q_2)^2 \\
&\quad + 8m_\pi^4((q_1 \cdot k_1) + (q_2 \cdot k_1))((q_1 \cdot k_2) + (q_2 \cdot k_2)) \\
&\quad + \frac{8}{3}|\vec{p}|^2((q_1 \cdot q_2)((k_1 \cdot k_2)(q_1 \cdot q_2) - (q_1 \cdot k_2)(q_2 \cdot k_1) - (q_1 \cdot k_1)(q_2 \cdot k_2)) \\
&\quad + m_\pi^2((q_1 \cdot k_1)(q_1 \cdot k_2) + (q_2 \cdot k_1)(q_2 \cdot k_2))). \tag{A6}
\end{aligned}$$

Then, we can express dot products in terms of covariant variables, using the relationship between momenta at the vertices as shown in Figure 2.1. For example, from $p_1 = k_1 + q_1$, we get three dot products for the ladder diagram:

$$k_1 \cdot q_1 = \frac{p_1^2 - k_1^2 - q_1^2}{2}, \tag{A7}$$

$$q_1 \cdot p_1 = -\frac{k_1^2 - q_1^2 - p_1^2}{2}, \tag{A8}$$

$$p_1 \cdot k_1 = -\frac{q_1^2 - p_1^2 - k_1^2}{2}. \tag{A9}$$

APPENDIX B: SOURCE CODES

In this appendix, we include source codes of the programs used in our calculation. The first program, `tpe.c` was for calculating the operators' amplitudes and integrating over internal loops. It took as input command line argument as follow:

```
# tpe epsilon kxstep
```

where *epsilon* is a parameter related to the Mandelstam variable s as

$$\sqrt{s} = 2m_N + \epsilon m_\pi, \quad (\text{B1})$$

and *kxstep* is the subinterval of integration over the internal loop. The output is a table of operators' contributions at each Gaussian point of $\cos\theta$. Each output line consists of the value of s , $\cos\theta$, the integration weight at the point, scalar, spin-spin, spin-orbit, and tensor operator's contributions, respectively.

The second program, `pwa.c`, scans each line from standard input for the data whose format has just been described. Hence, we can pipe the output from the first program as input for this program:

```
# tpe epsilon kxstep | pwa
```

The output is then phase shifts and mixture parameters.

In `pwa.c`, we use a function `plm(l,m,x)` which calculates the associated Legendre polynomial, $P_l^m(x)$. The code for this function, as well as other special functions, could be found in a numerical method text such as [18]. The subroutine `getline` which scans the input from standard input is not in C's standard library but could also be found in a C programming book such as [19].

Source for tpe.c

```
#include <stdio.h>

#include <stdlib.h>

#include <math.h>


#define Mn 0.93956563

#define Mpi 0.1349764

#define K_INT_RANGE 19.8

#define Mnsq (Mn*Mn)

#define Mpisq (Mpi*Mpi)

#define EPS 1E-32


#define GORDER 3

const double

    gpoint[GORDER]={0.238619186083197,
                    0.661209386466265,
                    0.932469514203512};

const double

    gweight[GORDER]={0.467913934572691,
                    0.360761573048139,
                    0.171324492379170};


double form_factor_sq(double delqsq) {
    double res,alpha1=0.8,alpha2=2.0,alpha3=2.0,sigma=5.2;
    res=(1.0 + alpha1*delqsq + alpha2*delqsq*delqsq \
        + alpha3*delqsq*delqsq*delqsq) \
        *exp(-sigma*delqsq*delqsq);
```

```

    return res;
}

main(int argc, char **argv){

    double s,t,u,costheta,epsilon;
    double denom,jacobian,Form;

    double Oiso,Os,Os2,Oso,Oso2,Ot,Ot2,OsOso,OtOs,OtOso;
    double p,p2,p4;
    double p1p3,p2p4,p1p4,p2p3;
    double
        dk1sq,dk2sq,
        k1sq,k2sq,q1sq,q2sq,
        Op, OsOp, OsoOp, OtOp,
        TRpfpiq1q2, TRpfq2k1q1, TRpfq2k2q1,
        TRpiq2k1q1, TRpiq2k2q1,
        k1k2, k1p1, k1p2, k1p3, k1p4, k2p1, k2p2, k2p3, k2p4,
        q1k1, q1k2, q1p1, q1p2, q1p3, q1p4, q1q2, q2k1, q2k2,
        q2p1, q2p2, q2p3, q2p4;
    double TrMat[16],TrVec[4],TrMatLapack[16],WORK[24];
    double C[4],Coef[4];

    int row,col;

    int i,j,lapack_info,lda=4,ldb=4,n=4,nrhs=4, ipiv[4],lwork=16;
    char trans='N',uplo='L';

```

```

double qa,qb,qc,qd;      // For quadratic equation

double kx,dkx,ky,kyi,kyf,kystep,kymid,kyrange,kxsq,kxsq_min;
double kxstep;
int kypoint,kylastpoint,totalpoint;
int sign=-1,tsign=-1;
int pt,tpt;

double beforeloop,contrib;
if(argc!=3){
    printf("Usage : %s epsilon kxstep\n",&argv[0][0]);
    exit(1);
}

epsilon=atof(&argv[1][0]);
kxstep=atof(&argv[2][0]);
s = (2*Mn + epsilon*Mpi)*(2*Mn + epsilon*Mpi);
for(tpt=0;tpt<GORDER;tpt++){
do
{
    tsign*=-1;
    costheta = tsign*gpoint[tpt];
    t = (2.0*Mnsq-0.5*s)*(1-costheta);
    u = (2.0*Mnsq-0.5*s)*(1+costheta);

    p = sqrt(s/4.0-Mnsq);
    p2=p*p; p4=p2*p2;

```

```

//external dotproducts
p1p3 = p2*costheta ;
p2p4 = p2*costheta ;
p1p4 = -p2*costheta ;
p2p3 = -p2*costheta ;

// Initialize result matrix
for(i=0;i<4;i++)
    Coef[i]=0.0;

q1sq=Mpisq; q2sq=Mpisq;

Oiso=16;
Os=0;
Os2=64;
Oso=32*p2*costheta;
Oso2=16*p4*(80*costheta*costheta + 24);
OsOso=128*p2*costheta;
Ot=p2*(-64*costheta+224.0/3.0);
Ot2=p4*(3584*costheta*costheta+8794*costheta+50816.0/9.0);
OtOs=p2*(-128*costheta+896.0/3.0);
OtOso=128.0/3.0*(33*costheta*costheta - 49*costheta +3);

TrMat[0]=Oiso;
TrMat[1]=Os;
TrMat[2]=Oso;
TrMat[3]=Ot;
TrMat[4]=Os;

```



```

TrMat[5]=0s2;
TrMat[6]=0s0so;
TrMat[7]=0t0s;
TrMat[8]=0so;
TrMat[9]=0s0so;
TrMat[10]=0so2;
TrMat[11]=0t0so;
TrMat[12]=0t;
TrMat[13]=0t0s;
TrMat[14]=0t0so;
TrMat[15]=0t2;
cparray(16,TrMat,TrMatLapack);
dgetrf_(&n,&n,TrMatLapack,&lda,ipiv,&lapack_info);

```

```

// We write jacobain in ky-quadratic form
// jacobian = sqrt( q_a*ky*ky + q_b*ky + q_c )
qa = 4*s;
qb = -4*(2*Mnsq + 2*Mpisq - t)*s;
qd = s*(4*Mnsq*Mnsq - 8*Mnsq*Mpisq \
      + 4*Mpisq*Mpisq + 4*Mpisq*s - s*t);
kxsq_min = (qb*qb/4.0/qa - qd)/4.0/u;
kx=fabs(sqrt(kxsq_min));
do
{
    beforeloop=Coef[3];
    kxsq = kx*kx;
    qc=4*u*kxsq + qd;

```

```

kymid=-qb/2.0/qa;
kyrange=fabs(sqrt(qb*qb - 4*qa*qc))/2.0/qa;
for(pt=0;pt<GORDER;pt++){
    sign=-1;
    do{          // We will have to do both plus and minus
        sign*=-1;
        ky=kymid + sign*gpoint[pt]*kyrange;
        k1sq=ky+kx;
        k2sq=ky-kx;

        // Proceed Calculation with k1sq k2sq as usual
        dk1sq=k1sq-Mnsq;
        dk2sq=k2sq-Mnsq;
        Form=form_factor_sq(dk1sq)*form_factor_sq(dk2sq);
        denom=((dk1sq)*(dk2sq));
        jacobian=16*fabs(sqrt(t*(qa*ky*ky + qb*ky + qc)));

        // Dotproducts
        // Explicit
        q1q2 = (t - q1sq - q2sq )*0.5;
        k1k2 = (k1sq - k2sq - u )*0.5;
        k1p1 = -(q1sq-Mnsq-k1sq)*0.5;
        q1k1 = (Mnsq-k1sq-q1sq)*0.5;
        q1p1 = -(k1sq-q1sq-Mnsq)*0.5;
        k1p3 = -(q2sq-k1sq-Mnsq)*0.5;
        q2k1 = -(Mnsq-q2sq-k1sq)*0.5;
        q2p3 = (k1sq-Mnsq-q2sq)*0.5;
    }
}

```

```

k2p2 = -(q1sq-k2sq-Mnsq)*0.5;
q1k2 = -(Mnsq-q1sq-k2sq)*0.5;
q2k2 = (Mnsq-k2sq-q2sq)*0.5;
q1p2 = (k2sq-Mnsq-q1sq)*0.5;
k2p4 = -(q2sq-Mnsq-k2sq)*0.5;
q2p4 = -(k2sq-q2sq-Mnsq)*0.5;
// Implicit
k1p4 = (k1k2+q2k1);
k2p1 = (k1k2+q1k2);
q2p2 = (q2k2-q1q2);
q1p4 = (q1k2+q1q2);
q2p1 = (q2k1+q1q2);
q1p3 = (q1k1-q1q2);
k1p2 = (k1k2-q1k1);
k2p3 = (k1k2-q2k2);

// Trace of operators
TRpfpiq1q2 = (p1p3+p2p4-p1p4-p2p3)*q1q2 \
              - (q2p3-q2p4)*(q1p1-q1p2) \
              + (q1p3-q1p4)*(q2p1-q2p2);
TRpfq2k1q1 = (q2p3-q2p4)*q1k1 \
              - (k1p3-k1p4)*q1q2 + (q1p3-q1p4)*q2k1;
TRpiq2k2q1 = (q2p1-q2p2)*q1k2 \
              - (k2p1-k2p2)*q1q2 + (q1p1-q1p2)*q2k2;
TRpiq2k1q1 = (q2p1-q2p2)*q1k1 \
              - (k1p1-k1p2)*q1q2 + (q1p1-q1p2)*q2k1;
TRpfq2k2q1 = (q2p3-q2p4)*q1k2 \
              - (k2p3-k2p4)*q1q2 + (q1p3-q1p4)*q2k2;

```

```

Op=16*Mnsq*q1q2*q1q2;
OsOp=q1q2*(k1k2*q1q2 - q1k2*q2k1 - q1k1*q2k2) + \
      Mpisq*(q1k1*q1k2 + q2k1*q2k2);
OsoOp= 2*Mnsq*q1q2*TRpfp1q1q2 \
      + TRpfq2k1q1*TRpiq2k2q1 \
      + TRpiq2k1q1*TRpfq2k2q1;
OtOp = (8*t + 8.0/3.0*p2)*Mnsq*q1q2*q1q2 \
      + 8*(Mpisq*Mpisq*(q1k1+q2k1)*(q1k2+q2k2)) \
      + 8.0/3.0*p2*(q1q2*(k1k2*q1q2-q1k2*q2k1 \
      - q1k1*q2k2) + Mpisq*(q1k1*q1k2+q2k1*q2k2));

TrVec[0]=Op;
TrVec[1]=OsOp;
TrVec[2]=OsoOp;
TrVec[3]=OtOp;
dgetrs_(&trans,&n,&nrhs,TrMatLapack,\
      &lدا,ipiv,TrVec,&lدا,&lapack_info);
if (lapack_info==0)
{
  for (i=0; i<4; i++)
  {
    C[i]=TrVec[i]*Form/jacobian/denom;
    Coef[i]+=gweight[pt]*kxstep*C[i];
  }
}
} while(sign==1);

```

```

    } // foreach ky point
    kx+=kxstep;

    contrib=fabs((Coef[3]-beforeloop)/Coef[3]);
}while (contrib>EPS); // while kx loop
printf("%1.16g %1.16g %1.16g",
        s, costheta, gweight[tpt]);
for (i=0; i<4; i++)
    printf(" %1.16g", Coef[i]);
printf("\n");
} while(tsign==1); // costheta loop
} // foreach tpoint = gausspoint
} // main

```

Source for pwa.c

```

#include <stdio.h>
#include <stdlib.h>
#include <math.h>

#define MAXLINE 256
#define MAX_J 5
#define M_N 0.93956563
#define M_PI 0.1349764
#define M_Nsq 0.8827835730772969

typedef int spin_mat[4];
const spin_mat mat_i={1,0,0,1};
const spin_mat mat_plus={0,1,0,0};
const spin_mat mat_minus={0,0,1,0};

```

```
const spin_mat mat_z={1,0,0,-1};
```

```
int sp_in[4]={1,-1,1,-1},
    sp_out[4]={1,1,-1,-1};
```

```
double Power(double x, int i) {
    if (i==2) return x*x;
    else return x*Power(x,i-1);
}
```

```
double CGspinspin(int s, int m_s, int s1, int s2) {
    double cgres;

    if (((s>1) || (abs(m_s)>1)) || ((abs(s1)!=1) || (abs(s2)!=1)))
        cgres=0.0;
    else
    {
        if(s==1)
        {
            if(m_s==1)
            {
                if((s1+s2)==2)
                    cgres=1.0;
                else
                    cgres=0.0;
            }
            else
                if(m_s==-1)
```

```

    {
        if((s1+s2)==-2)
            cgres=1.0;
        else
            cgres=0.0;
    }
    else // m_s==0
    {
        if(s1+s2!=0)
            cgres=0.0;
        else
            cgres=1.0/sqrt(2.0);
    }
}
else // s==0
{
    if((m_s!=0)|| (s1+s2!=0))
        cgres=0.0;
    else
        cgres=1.0*s1/sqrt(2.0);
}
}
return cgres;
}

```

```

int getline(char str[], int lim)
{
    int c,n;

```

```

for (n=0;n<lim-1 && (c=getchar())!=EOF && c!='\n';++n)
    str[n]=c;
if (c=='\n'){
    str[n]=c;
    ++n;
}
str[n]='\0';
return n;
}

```

```

double CGspin1(int j, int m, int l, int m_s) {
// Calculate Clebsh-Gordon coef with the case of spin = 1
// Hence, l = j-1, j, j+1 and m_s = -1,0,1

```

```

    double cgres;

```

```

    if ((l<0)|| (j<0))

```

```

        cgres=0.0;

```

```

    else

```

```

    {

```

```

        if (( abs(l-j)>1 ) || ( abs(m_s)>1 ))

```

```

        {

```

```

            cgres=0.0;

```

```

        }

```

```

    else

```

```

    {

```

```

        int indx=(l-j)*3 - (m_s)+4;

```



```

switch (indx) {
    case 0:
        cgres = sqrt((j+m)*(j+m-1)/((2.0*j-1)*2.0*j));
        break;
    case 1:
        cgres = sqrt((j-m)*(j+m)/(j*(2.0*j-1)) );
        break;
    case 2:
        cgres = sqrt((j-m)*(j-m-1)/(2.0*j*(2.0*j-1)));
        break;
    case 3: if (j==0)
        cgres=0.0;
        else
        cgres = -sqrt((j+m)*(j-m+1)/(2.0*j*(j+1)));
        break;
    case 4: if (j==0)
        cgres=0.0;
        else
        cgres = m/sqrt(j*(j+1)); break;
    case 5: if (j==0)
        cgres=0.0;
        else
        cgres = sqrt((j-m)*(j+m+1)/(2.0*j*(j+1)));
        break;

    case 6:
        cgres = sqrt((j-m+1)*(j+m+2)/((2.0*j+2)*(2.0*j+3)));
        break;

```

```

    case 7:
        cgres = -sqrt((j-m+1)*(j+m+1)/((j+1)*(2.0*j+3)));
        break;
    case 8:
        cgres = sqrt((j+m+1)*(j+m+2)/((2.0*j+2)*(2.0*j+3)));
        break;
    }
}
}

return cgres;
}

double yyy(int l1, int l2, int m, int ms){
    double pi;
    pi=2.0*asin(1.0);
    return sqrt((2*l1 + 1)*(2*l2+1)/12/pi)\
        *CGspin1(l1,0,l2,0)*CGspin1(l1,m,l2,ms);
}

double find_contrib_mix(int j,int m, int l1, int l2,
                        spin_mat mat1, spin_mat mat2,
                        int ms1, int ms2,
                        double mixcoef[MAX_J])
{
    int spindex1,spindex2,m1,m2,lambda;
    double res=0.0;
    lambda=j;
    for (lambda=j-2;lambda<=j+2;lambda++)

```

```

{
    for (m1=-1;m1<=1;m1++) for (m2=-1;m2<=1;m2++)
    {
        for (spindex1=0;spindex1<4;spindex1++){
            for (spindex2=0;spindex2<4;spindex2++){
                res+=CGspinspin(1,m1,sp_out[spindex1],sp_in[spindex1])
                    *CGspinspin(1,m2,sp_out[spindex1],sp_in[spindex2])
                    *CGspin1(j,m,l1,m1)*CGspin1(j,m,l2,m2)
                    *yyy(l1,lambda,m1,ms1)*yyy(l2,lambda,m2,ms2)
                    *mixcoef[lambda]
                    *mat1[spindex1]*mat2[spindex2];
            } // for spindex2
        } // for spindex1
    } // m1m2 double loop
} //lambda (double) loop
return res;
}

```

```

double find_contrib_nonmix(int j,int m, int l,
                           spin_mat mat1, spin_mat mat2)
{
    int spindex1,spindex2,m1,m2;
    double res=0.0;
    for (m1=-1;m1<=1;m1++) for (m2=-1;m2<=1;m2++)
    {
        for (spindex1=0;spindex1<4;spindex1++){
            for (spindex2=0;spindex2<4;spindex2++){
                res+=CGspinspin(1,m1,sp_out[spindex1],sp_in[spindex1])

```

```

        *CGspinspin(1,m2,sp_out[spindex1],sp_in[spindex2])
        *CGspin1(j,m,l,m1)*CGspin1(j,m,l,m2)
        *mat1[spindex1]*mat2[spindex2];
    } // for spindex2
} // for spindex1
} // m1m2 double loop
return res;
}

```

```

double contrib_nonmix(int j, int l, double p, double C1[MAX_J],
                     double Cs[MAX_J], double Ct[MAX_J])
{
    double nonmix;
    nonmix=(C1[l] - 8*p*p/3*Ct[l])
        *find_contrib_nonmix(j,1,l,mat_i,mat_i);
    nonmix+=(Cs[l] - 8*p*p/3*Ct[l])
        *find_contrib_nonmix(j,1,l,mat_plus,mat_plus);
    nonmix+=(Cs[l] - 8*p*p/3*Ct[l])
        *find_contrib_nonmix(j,1,l,mat_minus,mat_minus);
    nonmix+=(Cs[l] - 8*p*p/3*Ct[l])
        *find_contrib_nonmix(j,1,l,mat_z,mat_z);
    return nonmix;
} // contrib_nonmix

```

```

double contrib_mix(int j, int l1, int l2,
                  double p, double mixcoef[MAX_J])
{

```

```

double mix;

mix = find_contrib_mix(j,0,l1,l2,\
    mat_plus,mat_plus,-1,-1,mixcoef);
mix += -find_contrib_mix(j,0,l1,l2,\
    mat_plus,mat_minus,-1,1,mixcoef);
mix += sqrt(2.0)*find_contrib_mix(j,0,l1,l2,\
    mat_plus,mat_z,-1,0,mixcoef);
mix+= -find_contrib_mix(j,0,l1,l2,\
    mat_minus,mat_plus,1,-1,mixcoef);
mix+= find_contrib_mix(j,0,l1,l2,\
    mat_minus,mat_minus,1,1,mixcoef);
mix+= -sqrt(2.0)*find_contrib_mix(j,0,l1,l2,\
    mat_minus,mat_z,1,0,mixcoef);
mix+= sqrt(2.0)*find_contrib_mix(j,0,l1,l2,\
    mat_z,mat_plus,0,-1,mixcoef);
mix+= -sqrt(2.0)*find_contrib_mix(j,0,l1,l2,\
    mat_z,mat_minus,0,1,mixcoef);
mix+= 2.0*find_contrib_mix(j,0,l1,l2,\
    mat_z,mat_z,0,0,mixcoef);

return mix;
} // contrib_mix

main(){
    int spindex1,spindex2;
    int j=1,l,spin1,spin2;

    char line[MAXLINE];
    long points;

```

```

double s, costheta, weight, coef0, coef0s, coef0so, coef0t;
double PI, legendre;
double N, p, K, t;
double nonmix, mix, mix2;
double deg;
double S, S11, S22, S12, cos_eps;

double C1[MAX_J], Cs[MAX_J], Cso[MAX_J],
        Ct[MAX_J], mixcoef[MAX_J];
PI=2.0*asin(1.0);
coup=Power(0.95/M_PI,4)/64/Power(PI,6);
deg = 180/PI;
for (j=0; j<MAX_J; j++)
{
    C1[j]=0.0;
    Cs[j]=0.0;
    Cso[j]=0.0;
    Ct[j]=0.0;
}
// First we scan the input
while (getline(line, MAXLINE)>2) {
// Read data from STDIN
    sscanf(line, "%lf %lf %lf %lf %lf %lf %lf %lf",
           &s, &costheta, &weight, &coef0, &coef0s, &coef0so, &coef0t);
    t=(2.0*M_Nsq-0.5*s)*(1-costheta);
    for (j=0; j<MAX_J; j++)
    {

```

```

    legendre=weight*plm(j,0,costheta);
    // printf("Pj=%g\n",legendre);
    C1[j] += coef0*legendre;
    Cs[j] += coef0s*legendre;
    Cso[j] += coef0so*legendre;
    Ct[j] += coef0t*legendre;
}

} // done scanning

N=sqrt(s/4.0)+M_N;
p=sqrt(s/4.0-M_Nsq);
K=-2*PI*sqrt(s)/sqrt(s/4-M_Nsq)/coup;
for (j=0;j<MAX_J;j++)
    mixcoef[j]= 2.0*p*p*8*PI/12.0*(4.0 * Cso[j] - 8.0 * Ct[j]);

printf("%1.2e %1.2e",s,
    contrib_nonmix(0,1,p,C1,Cs,Ct)/K*deg);
for (j=1;j<3;j++)
{
    S11=contrib_nonmix(j,j-1,p,C1,Cs,Ct)/K;
    S22=contrib_nonmix(j,j+1,p,C1,Cs,Ct)/K;
    S12=contrib_mix(j,j-1,j+1,p,mixcoef)/K;

    S = 2.0*S12/(S11-S22);
    cos_eps = sqrt(1.0/(1.0+S*S));
    printf(" %1.2e %1.2e %1.2e %1.2e",
        S11*deg,S22*deg,S12*deg,acos(cos_eps));

```

```
}  
printf("\n");  
}
```