Strangeness Production Via Pion Rescattering in Ultrarelativistic Heavy-Ion Collisions

by
Ronald Richard Marquardt

SUBMITTED TO THE DEPARTMENT OF PHYSICS
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF
BACHELOR OF SCIENCE IN PHYSICS

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 1989

Signature of Author: ______________________
Department of Physics
May 12, 1989

Certified by: ______________________
Robert J. Ledoux
Assistant Professor, Department of Physics
Thesis Supervisor

Accepted by: ______________________
Aron Bernstein
Chairman, Undergraduate Thesis Committee

Strangeness Production Via Pion Rescattering in Ultrarelativistic Heavy-Ion Collisions

by

Ronald Richard Marquardt

Submitted to the Department of Physics
on May 12, 1989 in partial fulfillment of the
requirements for the Degree of
Bachelor of Science in Physics

ABSTRACT

Two models were developed to study the effects of pion rescattering on the observed K^+/π^+ ratio in relativistic heavy-ion collisions. The first model simulated the formation and decay of 13 N^* and 10 Δ pion-nucleon resonances in the limit of infinite cross-section. A maximum K^+/π^+ ratio of $\simeq 6\%$ was found for the decay products of these resonances. The second model was used to determine the effect of pion rescattering in a full simulation of nucleus-nucleus collisions. For 14.5 A·GeV/c Si-Au reactions, the K^+/π^+ ratio was found to change from $2.75 \pm 0.04\%$ for data without rescattering to $2.57 \pm 0.04\%$ with rescattering included. Neither of these results can explain the enhanced (19–24%) K^+/π^+ ratios observed by the E-802 collaboration for 14.5 A·GeV/c Si-Au collisions, possibly indicating the production of high baryon-density nuclear matter at these incident energies.

Thesis Advisor: Robert J. Ledoux
Assistant Professor, Department of Physics

Acknowledgements

I have many people to thank for their varied contributions to this thesis and my education. First and foremost, my family has been a constant source of support and encouragement throughout my four years at MIT. For their commitment and sacrifice, words alone cannot convey my appreciation.

Having worked in the Heavy Ion Group for almost three years, I must thank each and every one of its members for their assistance. Prof. Bob Ledoux has supervised this thesis and I thank him for his many discussions and helpful guidance. He has given me the freedom to approach problems in my own fashion, as well as the direction necessary to keep me going. He is a good educator and friend.

I am indebted to Prof. Lee Grodzins for sponsoring me as a UROP student during my first two years. Additionally, George Stevens and Steve Steadman have always been supportive and have helped to foster my education. George, especially, had the patience to endure my "help" during my first few months in the group, teaching me the basics of the electronics and detectors used in our experiments.

While all of the graduate students have, at one point or another, taken the time to explain some aspect of the physics, or answer a question regarding the experiment, I am especially indebted to Richard Morse and Martin Sarabura for their helpful and insightful discussions on a variety of topics related to this thesis.

Lastly, I must thank Mr. Horace T. Bawden, now retired, of Charlotte Wood Junior High School, Danville, CA. His inspired teaching, rigorous and challenging lessons, and kind, gentle wit long ago caught the heart of a wide-eyed eleven year-old and instilled in it an intense interest in the sciences and a deep passion to attend MIT. This thesis is a result of that interest, and the culmination of that passion.

Anybody who has been seriously engaged in scientific work of any kind realizes that over the entrance to the gates of the temple of science are written the words: Ye must have faith. *It is a quality which the scientist cannot dispense with.*

— Max Planck

The most beautiful thing we can experience is the mysterious. It is the source of all true art and science.

— Albert Einstein

That theory is worthless! It isn't even wrong!
— Wolfgang Pauli

Contents

List of Figures

List of Tables

Chapter I

Introduction

Recent advances in accelerator technology have opened a new era in experimental nuclear physics — the study of high energy, relativistic heavy-ion collisions. Both the Super Proton Synchrotron (SPS) at CERN and the Alternating Gradient Synchrotron (AGS) at the Brookhaven National Laboratory (BNL) are now capable of accelerating ions as massive as ^{32}S to momenta exceeding[1] [9, 25] 10 A·GeV/c, a regime previously accessible only through the study of rare cosmic ray events. Nuclear collisions at these energies provide the unique opportunity to study nuclear matter *in extremis*, conditions existing naturally only during the earliest moments in the creation of the universe.

These extreme, violent nuclear interactions result in phenomena wholly distinct from that observed at lower energies. Unlike traditional nuclear studies which probe the collective nature of the nucleus under the addition of energy, momentum, and angular

[1] Both facilities utilize a fixed target. SPS operates at a maximum of 200 A·GeV/c ($\sqrt{s}$ = 19.4 A·GeV), while the AGS has an upper limit of 14.5 A·GeV/c ($\sqrt{s}$ = 5.4 A·GeV).

momentum, relativistic heavy-ion experiments study the quark substructure of the individual nucleons — a proving ground for the theory of Quantum Chromodynamics (QCD) [14]. In theory, the impact of the energetic nuclei liberates quarks from the unobserved *quark sea*. They and the valence quarks of the original nucleons interact as individual particles, forming a *flux tube* that materializes in a high multiplicity shower of particles. A study of these products is thus an investigation of the color force that mediated their formation. An exciting limit to these energetic interactions predicted by QCD is the creation of a *quark-gluon plasma* [19], a phase transition in which a portion (or all) of the nucleons in the projectile and target become a deconfined plasma of disassociated constituent quarks. However, at the present, the research into hot, dense nuclear matter has primarily been theoretical, applying Quantum Chromodynamics to the many-body problem of colliding nuclei. Without the proper experimental investigation and physical data, these predictions remain speculative.

The E-802 experiment at the Brookhaven AGS is designed to measure the characteristics of produced particles in relativistic heavy-ion collisions. Using beams of ^{28}Si and ^{16}O on targets of Au, Ag, Cu, and Al, the collaboration[2] operates a 25 msr single-arm spectrometer with good particle identification, augmented by global event characterization detectors. While the data is still being analyzed, early results indicate an enhancement, relative to pp and pA reactions, of the K^+/π^+ ratio in certain kinematic regions. Although this observation was anticipated by theorists, several interpretations

[2]Researchers from ANL, BNL, Columbia, Hiroshima, INS, Kyushu, MIT, U.C. Berkeley and Riverside, and Tokyo are members of this international effort.

of its origin have been advanced [7, 13, 29]. Perhaps the simplest, and least extraordinary, explanation suggested has been the theory that pion-nucleon resonances formed within the volume of the hot, dense nuclear matter produce additional kaons, altering the final state particle distributions. The work presented herein investigates the effect of pion-nucleon resonances on the K^+/π^+ ratio, with the goal of properly interpreting the current experimental data.

Several Monte Carlo computer codes which have been developed to numerically simulate the formation of pion-nucleon resonances in high energy, heavy-ion experiments are presented in this thesis. The first, entitled `PiNRP`, determines the decay products of Δ and N^* resonances created by pions of known momenta incident on a fixed distribution of nucleons. A more sophisticated simulation, `NNEveSim`, applies the `PiNRP` program to the study of complete nucleus-nucleus interactions. The results of these two simulations allow for a better understanding of the existing E-802 data and test the theory that strangeness enhancement in ultrarelativistic nucleus-nucleus collisions may be attributed to the secondary formation of pion-nucleon resonant states.

Chapter II

Physics Background

Ultrarelativistic heavy-ion physics is the newest endeavor in a science that has long provided numerous experimental and theoretical challenges—nuclear physics. Although a fledgling field, boasting collisions of a more violent and catastrophic nature, relativistic heavy-ion nuclear physics shares with its lower energy counterparts the common goal of characterizing the so-called *strong* force. One of the fundamental forces of nature, the strong force is responsible for the strong, albeit short-ranged, attraction between nucleons, overcoming Coulomb repulsion and binding stable nuclei. Unlike the forces of electromagnetism or gravity, however, there exists no readily soluble theory analogous to Maxwell's equations or Newton's Law of Gravity to fully predict the effect of the strong force. Years of theoretical and experimental work have yielded only a qualitative description of strong interactions, and a crude, phenomenological understanding of nuclear reactions. To further expand upon this knowledge, high energy, heavy-ion physics seeks to examine nuclei under highly exotic, extreme conditions. Such investigations promise to test the theory of Quantum Chromodynamics and probe the quark substructure of the individual nucleons, observing the strong force at a very basic and fundamental level.

II–1 High Energy Nuclear Physics

High energy nuclear physics offers the unique ability to study nuclear matter at high densities ($\rho > \rho_0 = \rho_{NM} = 0.15$ fm^{-3}) and temperatures ($T > T_0 = 16$ MeV) [22]. Previous experimental probes consisting of electrons, pions, or protons, and even low energy nuclei, were insufficient to excite target nuclei well above their ground state in all but localized regions. With the new accelerator technology, it should now be possible to study previously unobserved phases of nuclear matter. Figure II–1 (from [22]) shows a rough phase diagram for nuclear matter with several reaction trajectories originating from (ρ_0, T_0) and evolving in time with various reactions. Note that the density axis is logarithmic whereas the temperature is scaled linearly. Consider, for example, the trajectory given for neutron stars and supernovae. Several of the trajectories from nucleus-nucleus collisions yield temperatures and densities comparable to those found in these extreme astrophysical phenomena, although in a much smaller volume. While the exact trajectories are hypothetical, and subject to experimental confirmation, it is obvious from the diagram that collisions of this nature subject nuclei to conditions far from their ground state. At these high energies, and high baryon densities, several phase transitions are expected to ultimately lead to a deconfined quark-gluon plasma. The exact nature of these transitions and the properties of this plasma are both essential to the further understanding of the elementary properties of nuclear forces, and of the quark constituents of the individual nucleons [3, 9, 16, 19, 22, 25, 28].

An examination of the phase diagram (as it currently understood theoretically) allows some insight into the dynamics of nuclear reactions in various energy regimes. The relationship between baryon density and mean nuclear temperature can be related to a pictorial depiction of a typical collision in the nucleus-nucleus center of mass. Current fixed target experiments operate in the region of phase space near 5-10 A·GeV. The trajectories for these reactions yield densities that are greater than that of normal nuclear matter. At these energies, thus, both nuclei more or less come to rest in the center of mass frame, a reaction mechanism titled the *nuclear stopping regime.* It is currently believed, from a preliminary analysis of experimental data, that the E-802 experiment (at center of mass energies for an Si-Au system of 5.4 A·GeV) results in nearly maximum nuclear stopping. In sharp contrast are those reactions, not yet attainable experimentally, which fall below (in density) the line marked "central regions" in Figure II–1. These reactions in the central rapidity region have average baryon densities that are very much less than ρ_0, but very high temperatures and energy densities. Collisions of this nature are best described in the center of mass as two regions of relatively high baryon density, the *fragmentation regions*, moving away from each other with a hot, baryon-free region, the *central region*, existing between. Because density is measured via the number of baryons per unit volume, i.e. baryon density, it is possible to have a nearly zero density, while having a finite energy, if the dominant constituent of the central region is non-baryonic matter composed from the disassociated quarks of participant nucleons as well as quark-

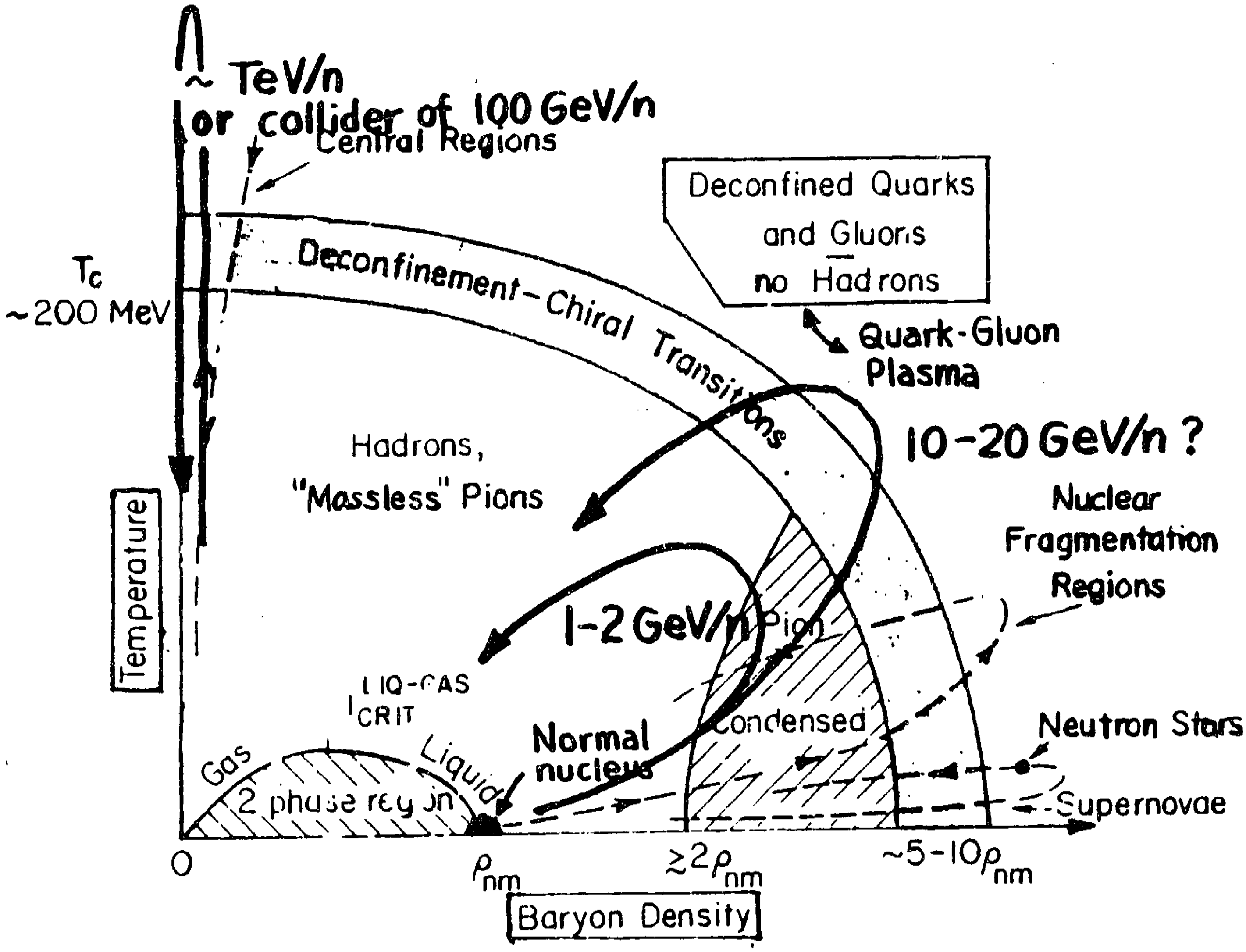


Figure II-1: The theoretical phase diagram for nuclear matter (from [22]).

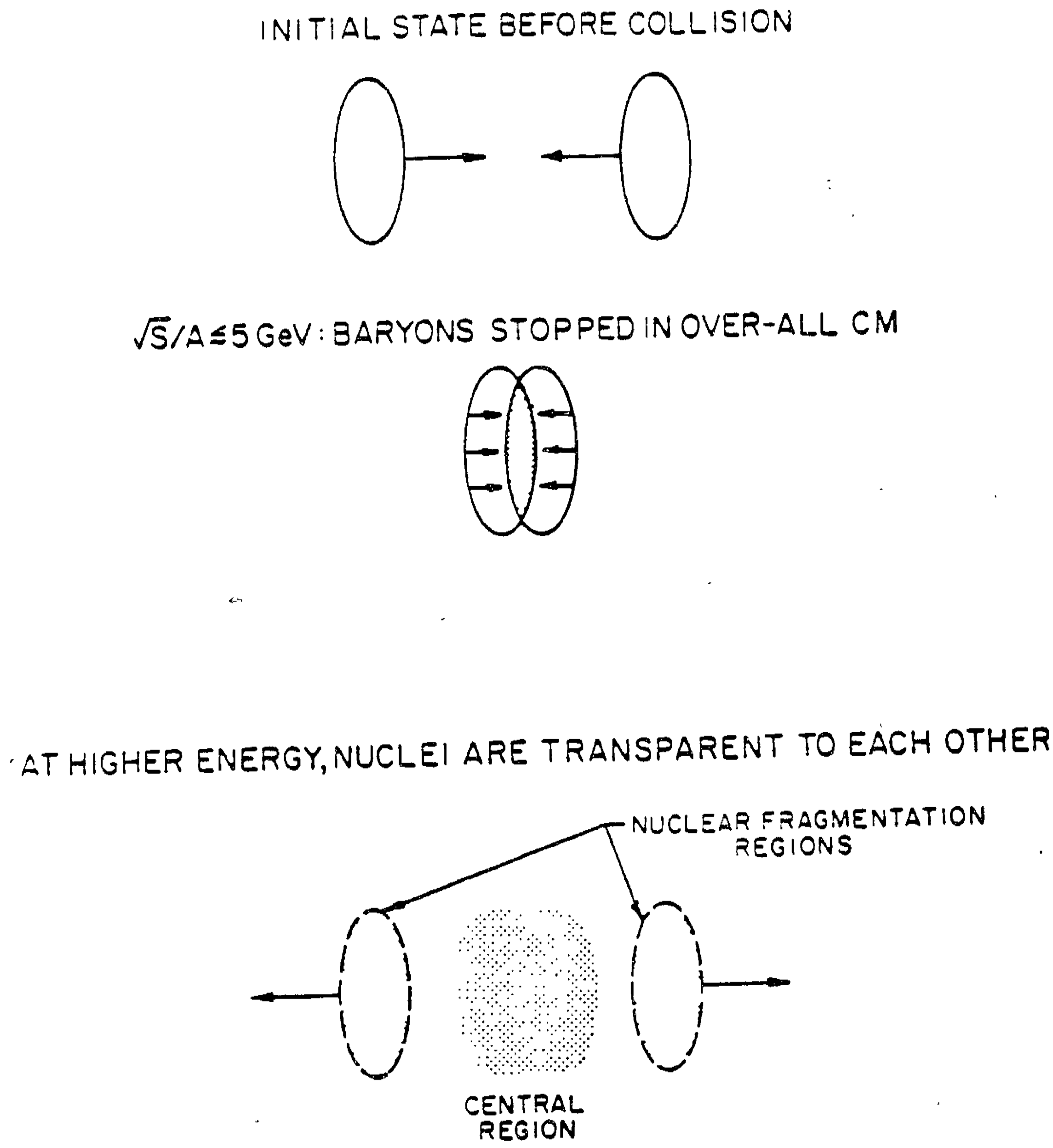


Figure II–2: Nuclear transparency in high energy nuclear collisions (from [17]) antiquark pairs produced from the vacuum. The central region thus is responsible for most of the vast number of produced particles that hadronize or freeze-out subsequent to the collision. In the energy regime in which a clear and definite central region is formed, nuclei are *transparent* as they literally pass through each other, leaving a hot, baryon-free, particle-producing region in their wake. Figure II–2 (from [17]) schematically shows the difference between full nuclear stopping and the production of a central region.

II–2 Strangeness Production: Indications of the Reaction Process

Experimentally, it is the particles that are produced and subsequently detected that provide the only clues to determining the dynamics of a nuclear reaction. From the type, energy, momentum, and distribution of these particles, the nature of the collision can be inferred. Many types of particles are produced in high energy, heavy-ion experiments; particles not seen in more traditional, lower energy, nuclear physics. At higher energies, more massive quarks are available for excitation, providing unique and informative signatures for experimental detection. Strange quark-antiquark pairs require $\simeq 2*150 =$ 300 MeV [19] to produce from the vacuum, an order of magnitude greater than that required for the light up and down quarks. These strange quarks manifest themselves in the form of produced kaons, Λ's, and Σ's. Produced positive kaons prove to be experimentally interesting due to their smaller cross-section for interaction with nucleons, relative to that for pions. For example, the K^+p cross-section is roughly 20 mb, while the π^+p cross-section climbs above 200 mb, on resonance. As a result, kaons have a mean-free path that is about 10 times longer than that of pions. Because they interact less frequently, kaons presumably carry more information about the earliest moments of the reaction, less tainted by intermediate reactions.

While a great deal might be learned solely from observed kaons, the statistics currently prevent this approach from being experimentally viable. Instead, a combination of kaon and pion data, among others, can be used to examine the properties of the nuclei during the reaction. Typically, quantities such as the K^+/π^+ and K^-/π^- particle ratios, defined as the ratio of the number of produced kaons to the number of produced pions, are used as experimental observables. Much can be inferred from the knowledge of these ratios. For example, consider a reaction resulting in near maximum nuclear stopping (c.f. Section II–1), where high baryon densities are achieved. In these circumstances, up and down quarks, most of them valence quarks from the nuclei, should be much more abundant than their antiparticles. Thus, when an $s\bar{s}$ pair is created from vacuum, the anti-strange ($\bar{s}$) quark has a much greater probability to locate an up (u) quark to form a K^+ than the strange (s) quark does to find an anti-down ($\bar{d}$) to form a K^-, allowing the creation of more K^+ than K^-. This effect is called *kaon distillation.* In contrast, a baryon-free central region, at higher energies, should have equal numbers of quark and anti-quark pairs, leading to identical K^+/π^+ and K^-/π^- ratios. Thus, an enhancement of the K^+/π^+ ratio relative to the K^-/π^- ratio is likely to be indicative of complete nuclear stopping.

Additionally, it is predicted that the K^+/π^+ ratio can yield a definite signature for the phase transition to a quark-gluon plasma. While the theoretical interpretations of an enhanced K^+/π^+ ratio vary [3, 7, 12, 13, 16, 18, 29], its significance can be argued on very simple grounds. Due to the quark structure of the K^+ ($u\bar{s}$) and π^+ ($u\bar{d}$), the K^+/π^+ ratio is essentially measuring the ratio of $\bar{s}$ to $\bar{d}$ quarks, both of which are not present at ground state energies as valence quarks and must have been products of the reaction. In proton-proton collisions, the ratio of strange quarks to light sea quarks of any one kind ($q = \bar{u}$, u, $\bar{d}$, d), R(s/q), is $\simeq$ 3% [6]. For a QGP, the same ratio is theoretically given by [6],

$$R_{QGP}(s/q) = \frac{2}{3} e^{(\mu - m_s)/T} \qquad (II-1)$$

where μ is the chemical potential for the light quark, typically ranging from essentially zero in the central region to over 300 MeV in the baryon-rich fragmentation region, $m_s \simeq$ 150 MeV is the mass of the strange quark, and T is the temperature, $\simeq$ 200 MeV. This ratio is much higher ($\simeq$ 30% for $\mu = 0$) than the pp result, where the hadronic matter is composed primarily of pions. It has also been argued [6], that the K^+/π^+ ratio should measure the initial ratio between strange and light quarks in a hot hadronic phase in which no QGP was created, providing that reabsorption processes do not significantly alter the measured K^+/π^+ value. If so, the K^+/π^+ ratio is a valuable experimental tool for determining the character of the early phases of a high energy, heavy-ion collision.

II–3 Pion-Nucleon Resonances

Because rescattering processes can affect the degree to which the K^+/π^+ ratio is informative, it is important that these effects be quantified. The process most likely to alter the K^+/π^+ ratio is that of pion-nucleon resonance formation. These resonant states, as excited composite particles. are the Δ's (J = 3/2) and the N^*'s (J = 1/2). While many pion-nucleon resonances exist at high energies, only 13 N^* and 10 Δ resonances have been experimentally tabulated [2]. Pions exhibit a relatively large interaction cross-section for reactions with nucleons which lead to the formation of these resonant states. Thus, secondary pions traveling through the nuclear medium can form hadron resonances and decay into products that change the K^+/π^+ ratio, thereby lessening the information it carries from the earliest stages of the reaction.

Fundamentally, the understanding of resonance formation requires the partial wave, quantum mechanical treatment of hadron scattering (c.f. [24]). The wave function, ψ, of an incoming pion scattering off of a fixed nucleon can initially be given by a plane wave,

$$\psi_i = e^{ikz} \qquad \text{(II} - 2)$$

where k is $1/\lambda$, the DeBroglie wavelength of the particle, and the time dependence has been explicitly ignored. This plane wave can in turn be expressed as a superposition of spherical waves,

$$\psi_i = \frac{i}{2kr}\sum_l (2l+1)\left[(-1)^l e^{-ikr} - e^{ikr}\right] P_l(Cos\theta) \qquad \text{(II} - 3)$$

in which the first exponential corresponds to a plane wave approaching the origin (for $z < 0$) while the second exponential represents a plane wave leaving the origin ($z > 0$). Each term in l corresponds to a distinct angular momentum state of quantum number l. a *partial wave*. The scattering center at $z = 0$ cannot affect the incoming wave. but it can change the phase of the l^{th} outgoing partial wave by $2\delta_l$ and the amplitude by η_l, where $0 < \eta_l < 1$. Applying this criteria, it can be determined [24] that the *scattered* wave is of the form,

$$\psi_{scat} = \frac{e^{ikr}}{r} F(\theta) \qquad (II-4)$$

where the scattering amplitude, $F(\theta)$ depends upon δ_l and η_l as,

$$F(\theta) \propto \sum_l f(l) = \sum_l \frac{\eta_l e^{2i\delta_l} - 1}{2i} \qquad (II-5)$$

The quantity $f(l)$ is the scattering amplitude of the l^{th} partial wave, and is key to the understanding of resonance phenomena. If the amplitude $f(l)$ passes through maximum for some value of l and $\lambda\!\!\!^{-}$, the incident particle and the scattering center are said to resonate. It can be shown (c.f. [24]) that the cross-section for such an resonance is proportional to the square of $f(l)$,

$$\sigma = 4\pi \lambda\!\!\!^{-2} \sum_l (2l+1) \left| f(l) \right|^2 \qquad (II-6)$$

From the functional form of $f(l)$, it is clear that such a resonance occurs for $\delta_l = \pi/2$. In such a resonant state, the incident and scattering particle are spoken of as a composite system of angular momentum $J = l$, which can subsequently decay.

From the partial wave scattering amplitudes, it is possible to derive the functional form of the cross-section for resonant state formation. Assuming that the phase shift, δ_l, is a function of energy, and that no internal excitations occur ($\eta_l = 1$), we drop the subscript l for brevity and express $f(E)$ as,

$$f(E) = \frac{e^{i\delta(E)}\left(e^{i\delta(E)} - e^{-i\delta(E)}\right)}{2i} = e^{i\delta(E)}\sin\delta(E) = \frac{1}{\cot\delta(E) - i} \qquad (\mathrm{II}-7)$$

Near the resonance, $\delta(E) \simeq \pi/2$ and $\cot\delta \simeq 0$, so the expression for $\cot\delta$ can be expanded about the resonance energy, E_0,

$$\cot\delta(E) = \cot\delta(E_0) + (E - E_0)\left[\frac{d}{dE}\cot\delta(E)\right]_{E=E_0} + \cdots \qquad (\mathrm{II}-8)$$

Defining $2/\Gamma = -\left[\frac{d}{dE}\cot\delta(E)\right]_{E=E_0}$, and assuming that we can neglect higher order terms in the expansion, Equation II-8 can be simplified to,

$$\cot\delta(E) \simeq -(E - E_0)\frac{2}{\Gamma} \qquad (\mathrm{II}-9)$$

or,

$$f(E) \simeq \frac{\Gamma/2}{(E_0 - E) - i\Gamma/2} \qquad (\mathrm{II}-10)$$

Thus, using Equation II-6, the cross-section can be expressed as a function of energy and the width parameter Γ,

$$\sigma = 4\pi \lambda\!\!\!^{-2}(2J+1)\frac{\Gamma^2/4}{(E-E_0)^2+\Gamma^2/4} \qquad (II-11)$$

Equation II-11 is the energy dependent Breit-Wigner cross-section for resonant states. Several adjustments must be made, however, to express it in its most general form. First, Γ, in general, equals the sum of Γ_{el}, the width of the elastic channel ($\pi N \rightarrow \Delta/N^* \rightarrow \pi N$), and Γ_r, the width of the inelastic decay channels. For reasons beyond this discussion (c.f. [24]), this modification, the Γ^2 in the numerator of II-11 should be, in general, replaced by $\Gamma_{el}\Gamma_r$. In addition, II-11 weights the cross-section only for the spin multiplicity of the final state resonance, but not by the initial pion and nucleon spin states. Thus the generalized Breit-Wigner formula for the cross-section of a pion-nucleon resonance is given by,

$$\sigma(E) = \frac{2J+1}{(2s_\pi+1)(2s_N+1)}\pi \lambda\!\!\!^{-2}\frac{\Gamma_{el}\Gamma_r}{\frac{\Gamma^2}{4}+(E-E_0)^2} \qquad (II-12)$$

It is this energy dependent cross-section which can be used to determine the magnitude of pion rescattering within the nuclear media, and the subsequent effect it may have upon the K^+/π^+ ratio.

II–4 The E-802 Experiment

The E-802 collaboration at the Brookhaven National Laboratory has been using the new heavy-ion capability of the Alternating Gradient Synchrotron (AGS) to study a variety of heavy-ion collisions at projectile energies of 14.5 A·GeV. Systems of silicon (^{28}Si) and oxygen (^{16}O) beams on gold (Au), silver (Ag), copper (Cu), and aluminum (Al) targets have been studied. While data for all of these systems exists, only analysis of the Si-Au data has been made [20].

The primary goal in the design of the E-802 spectrometer [1], shown in Figure II–3, as well as in its use, has been to measure inclusive particle spectra with good particle identification over a large range of rapidity and transverse momentum. The single-arm magnetic spectrometer covers a solid angle of approximately 25 msr at an adjustable angle of 5° to 58° from the beam axis, providing good particle identification for momenta from $\simeq$ 0.5 to 5.0 GeV/c. Additional detectors provide event characterization measurements such as charged particle multiplicity, neutral energy flow, and zero-degree calorimetry. The entire apparatus was designed to function properly for event multiplicities through the magnet of up to approximately 15. At this multiplicity, the detectors are designed to have a > 95% chance of having at *most* one double hit per event. As a result of these design criteria, the spectrometer should be capable of measuring produced hadron momentum spectra at a resolution of $\Delta p/p \leq 0.005p$, as well as providing accurate particle identification at all spectrometer settings.

Figure II–4 (from [20]) shows a sample spectrum for TOF versus inverse momentum at a spectrometer angle of $\theta_{lab} = 14°$–$28°$ for Si+Au at 14.5 A·GeV. This data demonstrates the excellent particle separation attainable from the E-802 spectrometer at all but the highest particle momenta. Perhaps the most intriguing result from E-802 to date has been the enhanced K^+/π^+ ratio observed in some kinematic regions. Table II–1

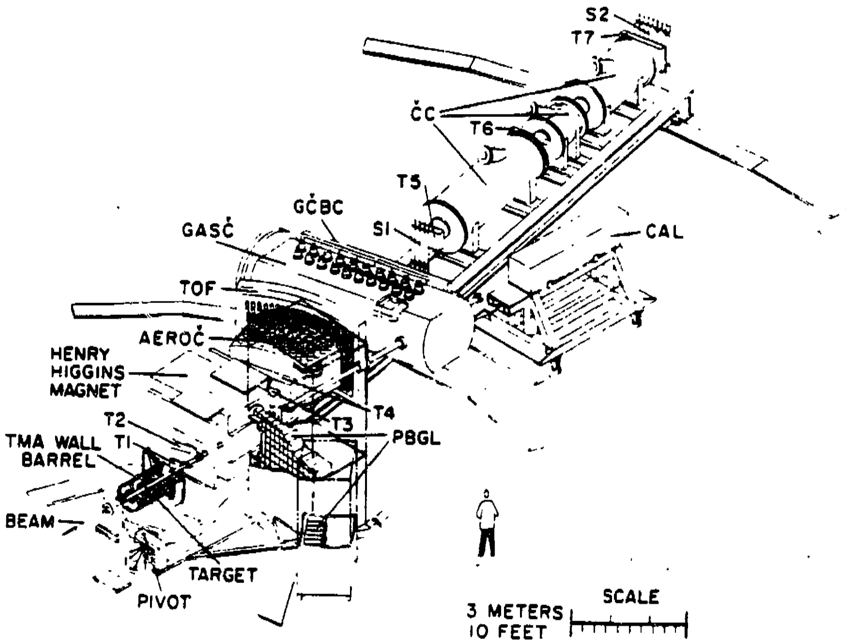


Figure II–3: The E-802 Spectrometer (from [1]).

(from [20]) summarizes the measured K^+/π^+ and K^-/π^- ratios for 14.5 A·GeV/c Si+Au collisions. The MIN BIAS trigger selects events for which any nucleus-nucleus reaction has occurred, whereas the central trigger selectively includes only those events with low impact parameter (as determined from the charged particle multiplicity). Within the experimental error, the K^+/π^+ ratio is 4-5 times the K^-/π^- ratio. Additionally, the K^+/π^+ ratio is 3-4 times greater than that found in *pp* and *pA* reactions [20]. Interpreting the

Table II–1: The observed K^+/π^+ ratio for 14.5 A·GeV/c Si+Au (E-802)

	MIN BIAS trigger	*CENTRAL trigger*
K^+/π^+	$19 \pm 5\%$	$24 \pm 5\%$
K^-/π^-	$6 \pm 5\%$	$4^{+4\%}_{-2\%}$

meaning of this K^+/π^+ ratio enhancement, relative to both the K^-/π^- ratio and the

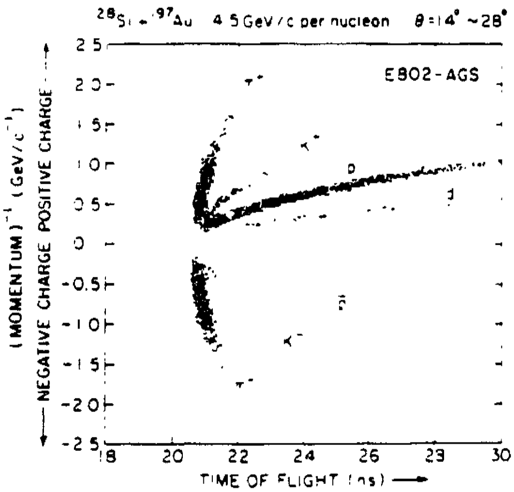


Figure II–4: A sample spectrum from the E-802 spectrometer (from [20]).

pp and pA K^+/π^+ values, requires an understanding of the effect of pion rescattering. Should this and other absorptive processes not significantly alter the initial information content of the K^+/π^+ ratio, the measured value may be of considerable experimental significance. The remainder of this work seeks to quantify the effect of pion rescattering on the final state particle distributions within the context of the E-802 experiment and its observations.

Chapter III

The PiNRP Model

A code to model the formation and decay of pion-nucleon resonances, `PiNRP` (Pion-Nucleon Resonance Production), has been developed to study the basic properties of these reaction channels. A structure chart for the code is given in Figure III–1. Written in `FORTRAN`, `PiNRP` calculates the momentum-dependent cross-sections and decay modes for 13 N^* and 10 Δ resonances. Particle distributions from this simulation provide a simple, yet informative, measure of the degree to which strangeness production from secondary interactions may be attributed to resonance mechanisms. Clearly, a large (or rising) cross-section for K^+ production at a given pion momentum is a possible indication of an enhanced (or increasing) K^+/π^+ particle ratio. The model also retains additional significance within this investigation through its use in the design, interpretation, and verification of the more physically complete simulation, `NNEveSim`, detailed in Chapter IV. A complete understanding of the `PiNRP` model, its philosophy as well as its results, is thus as important to the strength of later conclusions as it is to the knowledge of pion-nucleon resonance formation and decay.

III–1 The Model

The PiNRP model is a simple simulation of N* and Δ resonance formation and decay. Given an incident pion of known momentum in the rest frame of a target nucleon, the model assumes that an interaction has occurred (as might be predetermined in a more sophisticated model, such as NNEveSim). A Monte Carlo calculation based upon the relative cross-sections of those resonances allowed from isospin conservation is used to determine which resonance is actually produced. The decay products from this channel are likewise calculated and returned to the user. The VAX FORTRAN implementation of this procedure is contained in Appendix B. The specifics of the model, as well as the assumptions and limitations implicit in its inception, are best discussed within the context of the naturally occurring division between the calculation of cross-sections and the determination of individual resonance decay products.

III–1.1 Pion-Nucleon Resonance Cross-Section Calculation

The PiNRP model limits itself to the consideration of only 13 N* (through N(2600)) and 10 Δ (through Δ(2420)) pion-nucleon resonances. The justification is quite simple—these are the only resonances on which there exists sufficient data. In fact, several of the more massive resonances have not been completely characterized experimentally, and have been included in the PiNRP model with the caveat that the model fail, i.e. return an

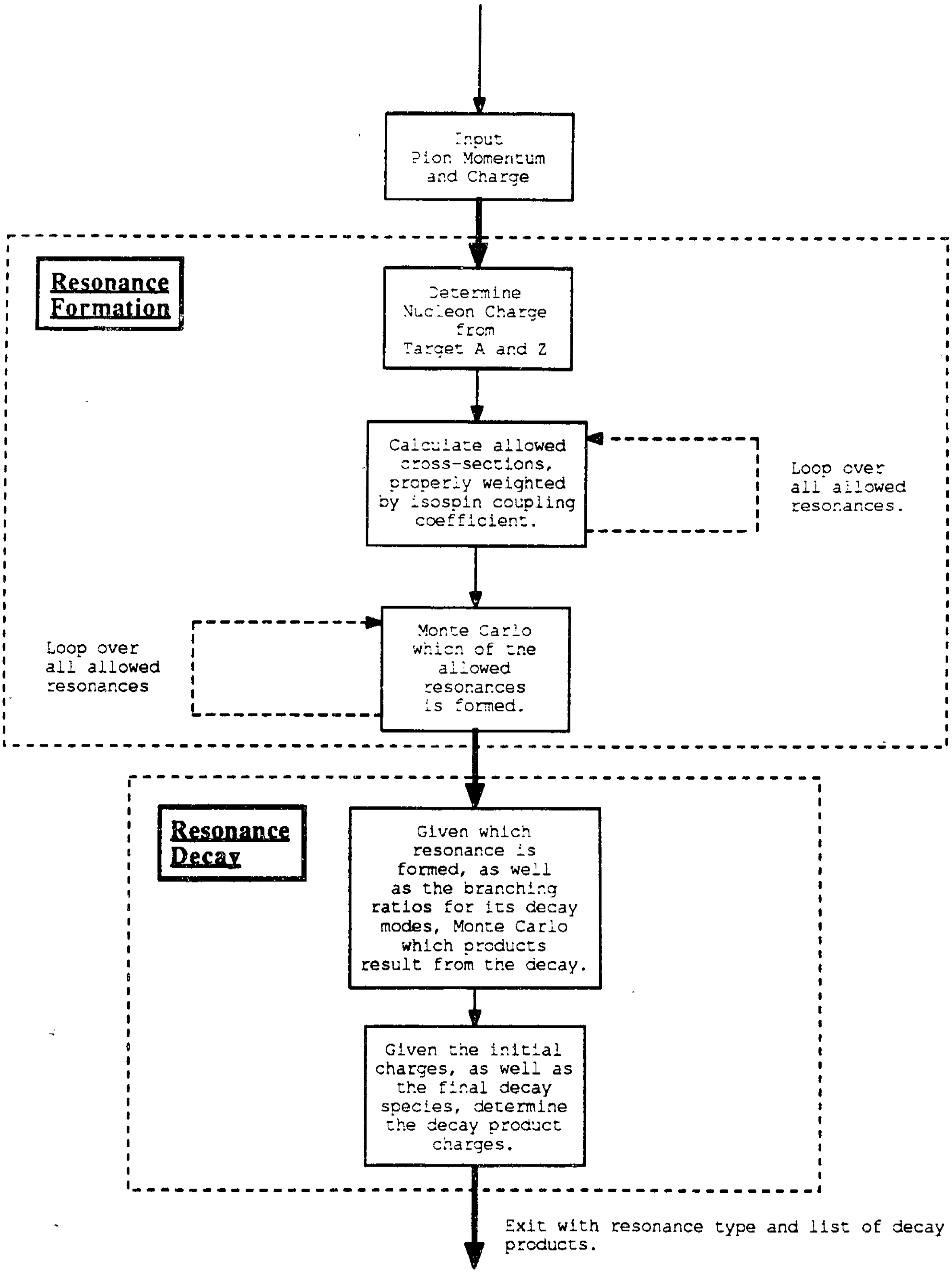


Figure III–1: Structure of the PiNRP Model error condition and no decay particles, when an ill-determined resonance decay results. Therefore, the PiNRP model, particularly the calculation of cross-sections, is of limited utility for pion momenta greater than roughly 1.2–1.8 GeV/c, and is not recommended at all for momenta exceeding 2.0 GeV/c. The pion-nucleon cross-section above these momenta is primarily determined by a continuum of closely spaced resonances, none of which are well-known experimentally. All resonance data used for the PiNRP model have been taken from the 1986 *Review of Particle Properties* [2], and represent rough averages of the data available at that time.

An underlying premise of the PiNRP model is the assumption that an interaction between a pion and nucleon has occurred. The mechanism by which this event happened is irrelevant to the model, the only goal of which is to determine the resulting resonance and decay products. Given the charge of both the pion and nucleon, the cross-sections of formation for only those resonances allowed by isospin (i.e., charge) conservation need be considered. Baryons of isospin 1/2, the N^* resonances can only form N^+ and N^0 variants, whereas the isospin 3/2 Δ's can form Δ^{++}, Δ^+, Δ^0, or Δ^-. Therefore, the model must carefully consider only the cross-sections of those resonances that can be formed and from these determine which channel is actually created.

The procedure by which the individual cross-sections are determined is straightforward. The cross-section for the formation of a single resonance, as a function of the pion energy or momentum, is assumed to be of the general Breit-Wigner form, as derived in Section II–3. Specifically, the model presumes that the cross-section for the β channel of the ith resonance is given by:

$$\sigma_{\pi+N \to \beta}(E_\pi, \beta) = \frac{2J+1}{(2s_\pi+1)(2s_N+1)} \pi \lambda\!\!\!^{-2} \frac{\Gamma_{el}\Gamma_\beta}{\frac{\Gamma^2}{4} + (E_\pi - E_0)^2} \qquad \text{(III} - 1)$$

where $\lambda\!\!\!^{-}$ is the reduced DeBroglie wavelength of the incident pion ($= \hbar/p_\pi$), Γ_{el} is the partial width of the elastic channel. Γ_β is the partial width of the β decay channel, Γ is the full width of the resonance. J and E_0 are the total spin and energy (mass) of the resonant system. and s_π and s_N are the total spin of the pion and nucleon, respectively. Summing over the various decay channels, β, Equation III-1 becomes:

$$\sigma_i(E_\pi) = \sum_\beta \sigma_{\pi+N \to \beta}(E_\pi, \beta) = \frac{2J+1}{6} \pi \lambda\!\!\!^{-2} \frac{\Gamma_{el}\Gamma}{\frac{\Gamma^2}{4} + (E_\pi - E_0)^2} \qquad \text{(III} - 2)$$

where the spin multiplicity factors for the pion and nucleon have been explicitly evaluated. The specific data used for each resonance is documented in Appendix A and in the code in Appendix B.

Given the analytic form, Equation III-2, for the total cross-section, the PiNRP model calculates these values at the incident pion momentum for all allowed resonances, properly weighting them by the Clebsch-Gordan isospin coupling coefficients. The probability of an individual resonance being formed in an interaction is then defined as the ratio of the cross-section for that particular resonance divided by the sum of all cross-sections, i.e.,

$$p_i \equiv \frac{\sigma_i}{\sum_i \sigma_i} \qquad \text{(III} - 3)$$

A Monte Carlo on this set of probabilities yields the resonance formed. This information is returned to the user to facilitate the study of specific resonance channels.

For example, consider a $\pi^+ n$ interaction. Knowing the charges of the pion and nucleon, PiNRP determines which resonances can be formed, and what Clebsch-Gordan coefficients are used to weight their relative cross-sections. As the π^+ can be represented by an isospin state vector $|1\ 1>$, and the neutron by $|\frac{1}{2}\ -\frac{1}{2}>$, the possible final isospin states are given by $|\frac{3}{2}\ \frac{1}{2}>$ and $|\frac{1}{2}\ \frac{1}{2}>$, a Δ^+ and N^{*+}, respectively. The Clebsch-Gordan coupling coefficients for these final states are $\sqrt{\frac{1}{3}}$ for $|\frac{3}{2}\ \frac{1}{2}>$ and $\sqrt{\frac{2}{3}}$ for $|\frac{1}{2}\ \frac{1}{2}>$. For each of the 13 N^* and 10 Δ resonances, PiNRP calculates the Breit-Wigner cross-section as a function of the pion momentum and weights it by the square of the appropriate Clebsch-Gordan coefficient; in this example, $\frac{2}{3}$ for the N^*'s and $\frac{1}{3}$ for the Δ's. Each of these individual cross-sections, as well as their sum, is retained. A random number between 0 and 1 is then multiplied by the total cross-section and the result compared with the list of individual cross-sections. If the cross-section for the first resonance on the list is less than the given fraction of the total cross-section, the cross-section for the next resonance is added to the first and the comparison is made again. When the cumulative sum of cross-sections exceeds or equals the given fraction of the total cross-section, the last resonance added to the sum is chosen as the resonance that is formed. This Monte Carlo method of choosing which resonance is created is independent of the ordering of the individual cross-sections in the list, so long as the random number is uniformly distributed between 0 and 1. In the case of $\pi^+ n$, the N^* resonances are effectively weighted by a factor of two over the Δ's, and at a given momentum the above procedure might result in the production of the N(1650) resonance. With this information, the PiNRP model is ready to determine the resulting decay products.

III–1.2 The Determination of Decay Products

Given a resonance and the charges of the initial pion and nucleon, the model determines, via a Monte Carlo calculation, what decay products are formed in a specific event. Two, three, and, in one instance, four body decays can result. Table III–1 summarizes which decay products are possible.

Table III–1: Annotated List of Resonance Decay Products

Decay Products	*Branching Ratios*	*Notes*
$N\pi$	5–94%	The elastic channel, including charge exchange.
$N\eta$	0.10–50%	The η decayed separately.
ΛK	0.10–15%	These are the only sources of kaons in
ΣK	1–12%	N* and Δ resonances.
$\Delta\pi$	2–60%	Form a Δ, and decay it as well.
$N\rho$	2–50%	These are both treated as $N\pi\pi$, as
$N(\pi\pi)_{symmetric}$	2–45%	the ρ decays primarily in this fashion.
$N\gamma$	0.03–1%	Rare.

Note that among the products, N is generally a proton or neutron, not an excited N* resonance, whereas Δ resonances are common decay products, which are themselves allowed to decay. The sole exception to this rule occurs with the decay of the Δ(1910), which explicitly decays into the N(1440) resonance plus a single pion. The charges of the individual products are determined from the initial charges of the pion and nucleon. Once these decay products have been determined from the accepted branching ratios documented for each resonance in Appendix A, identification codes corresponding to the

GEANT standard [5] are returned to the user for further analysis. In addition, the model returns an error code whenever the decay products are uncertain. For several of the more massive resonances, only 5–10% of the decay channels are known.

III–1.3 PiNRP Code Implementation

The PiNRP model has been implemented in VAX FORTRAN. The code itself is given in Appendix B. The entire model is contained in a large subroutine, designed to be called from an external shell that passes specific pion-nucleon events and analyzes the results returned by PiNRP.

The main subroutine is structured so as to be both clear and modular. It is divided into two distinct sections, one to determine which resonance is formed and another to resolve which decay products are created. Figure III–1 schematically details this structure. It was required that the decay section be reentrant as many of the N^*'s and Δ's decay into lower energy resonances. Separate subroutines exist for each decay channel to ensure the proper conservation of charge for that specific mode. In a sacrifice for improved readability, the code has not been optimized for execution speed. Subroutine calls are used where ever possible to increase modularity, and loops, branches, and conditionals are constructed to highlight their use and significance in the code. Nominally, 10 msec of CPU time is required to process and write out a single pion-nucleon event. To create the canonical data sets that are later discussed, nearly 9 hours of CPU time was required to simulate 1,000,000 pion-nucleon events at 14 different momenta for a given pion species. Three of such data sets were generated.

III–2 Results and Discussion

The PiNRP model provides the simplest means of observing strangeness production via pion-nucleon resonance formation. To facilitate these investigations, a shell program, PiNBIN, was written to provide a useful interface to the PiNRP subroutine. The user can select the number of events to simulate, which pion species are used, what momenta they have, and the charge distribution of nucleons they are incident upon. For each event, the PiNRP model is called and the resulting decay products are binned by particle type and incident pion momentum. In addition, to ensure that PiNRP was a reasonable model and simulation, the total πN interaction cross-section, as calculated within PiNRP was determined and compared with existing data.

III–2.1 Verification of the Pion-Nucleon Cross-Sections

Figures III–2—III–7 plot the total interaction cross-sections as calculated by PiNRP. Comparison is made with experiment, where possible. Experimental data is only available for π^+p and π^-p, due to the difficulties in studying systems involving neutral particles. Because of the isospin invariance of cross-section, however, this data should allow for the determination of the π^+n and π^-n cross-sections as well.

As can be seen in Figures III–2 and III–3, the PiNRP model approximates the total interaction cross-sections well for pion momenta between 0.25–1.5 GeV/c. For reasons mentioned earlier, the calculated cross-section drops unphysically for momenta exceeding 1.2–2.0 GeV/c. Deviations from the experimental data are best explained by

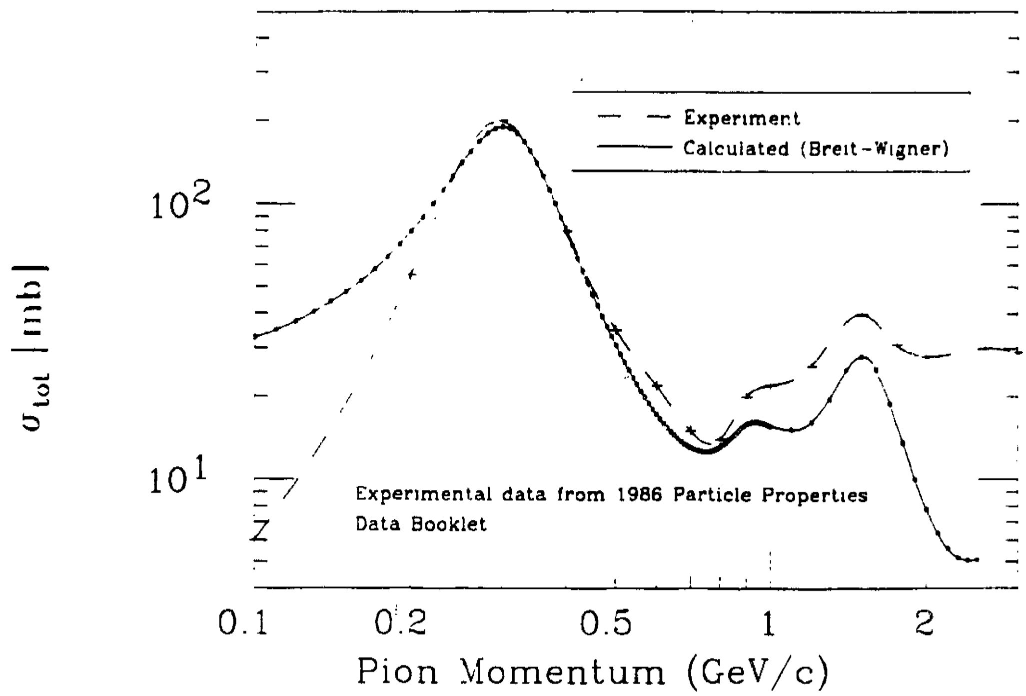


Figure III-2: The total π^+p interaction cross-section (calculated and experimental).

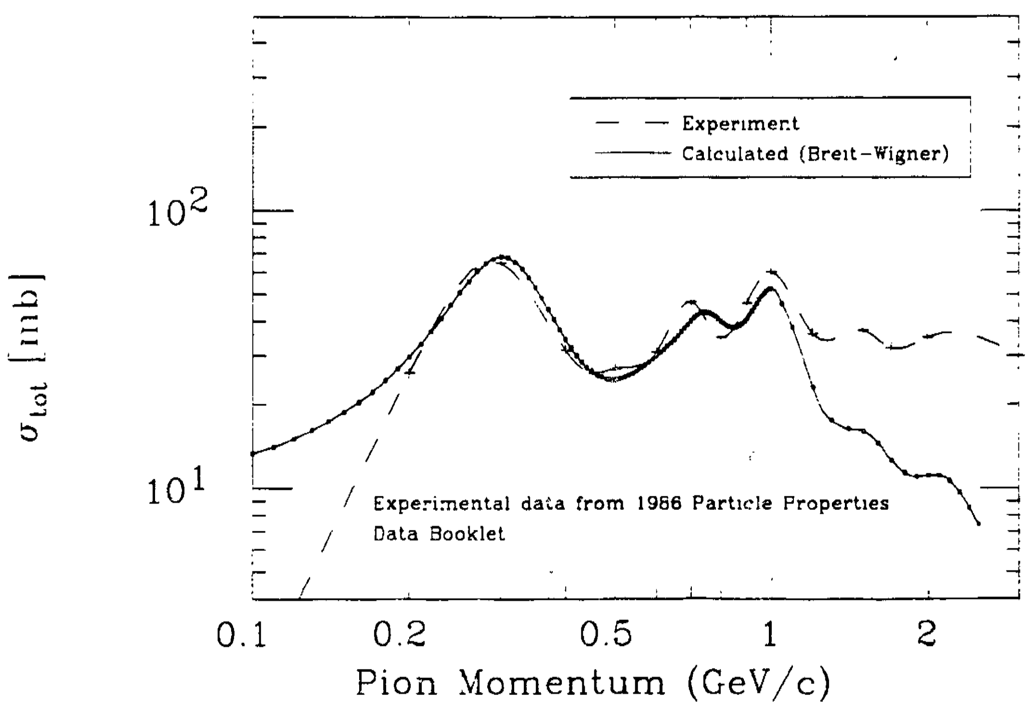


Figure III-3: The total π^-p interaction cross-section (calculated and experimental).

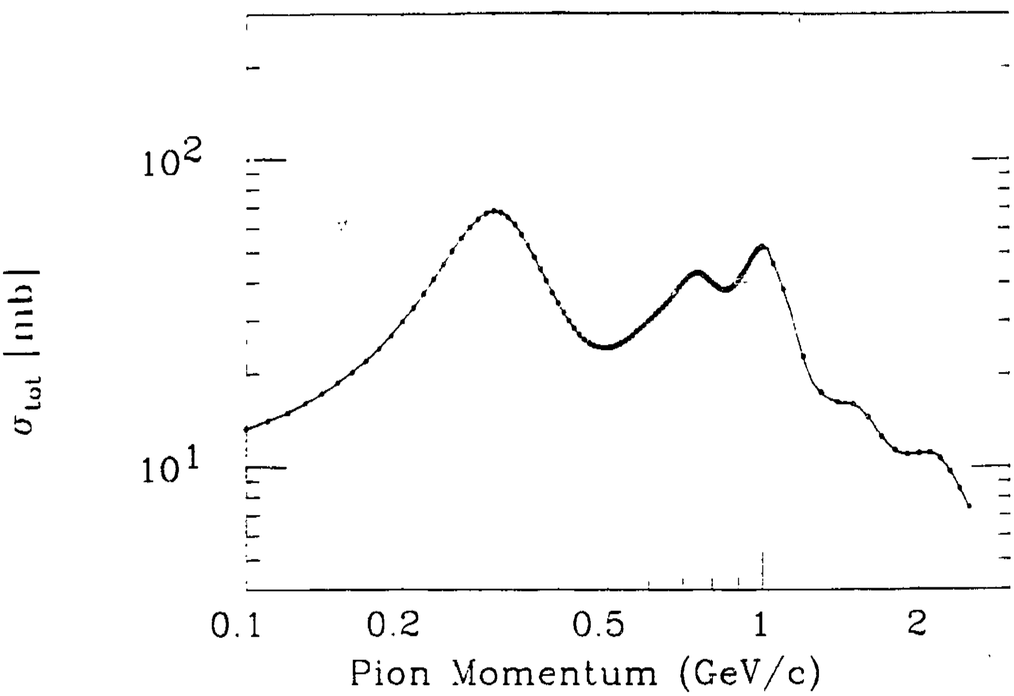


Figure III–4: The total calculated $\pi^+ n$ interaction cross-section.

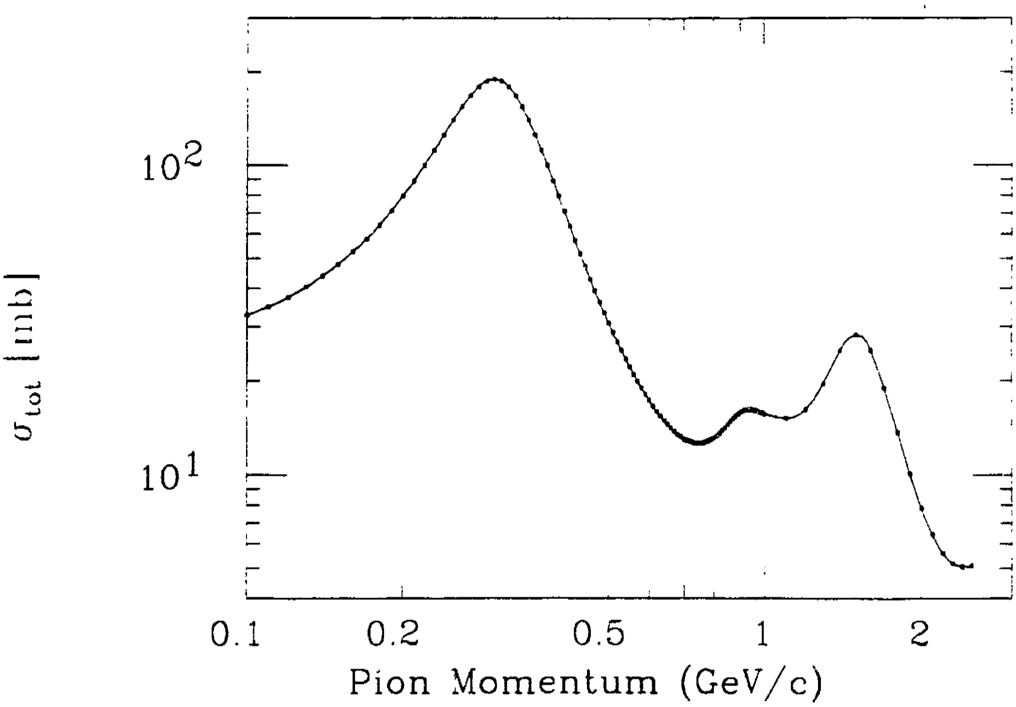


Figure III–5: The total calculated $\pi^- n$ interaction cross-section.

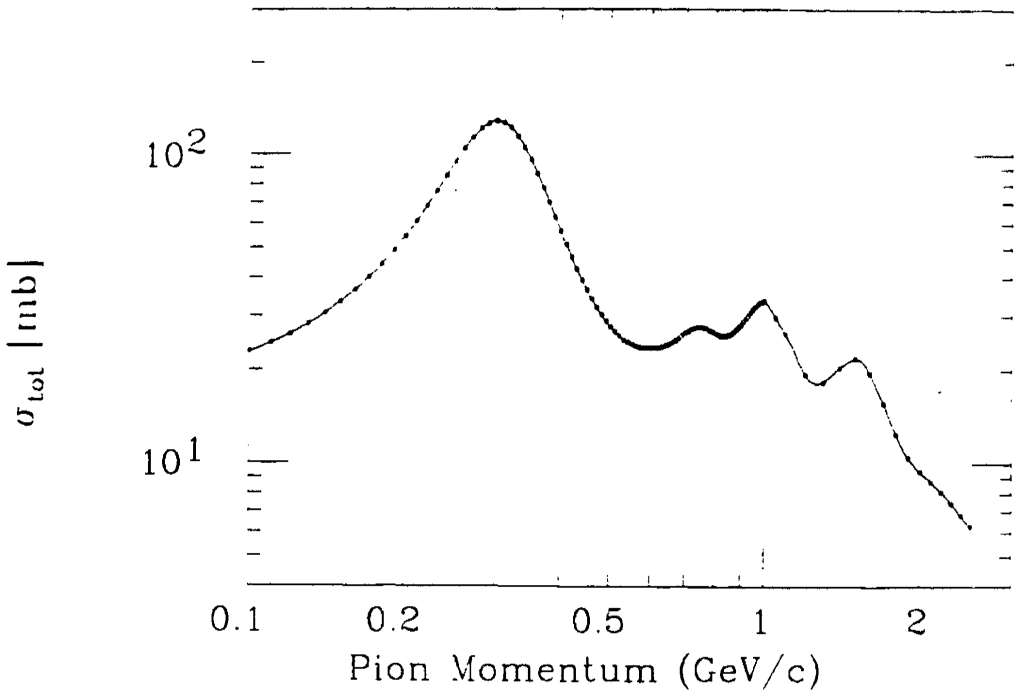


Figure III-6: The total calculated $\pi^0 p$ interaction cross-section.

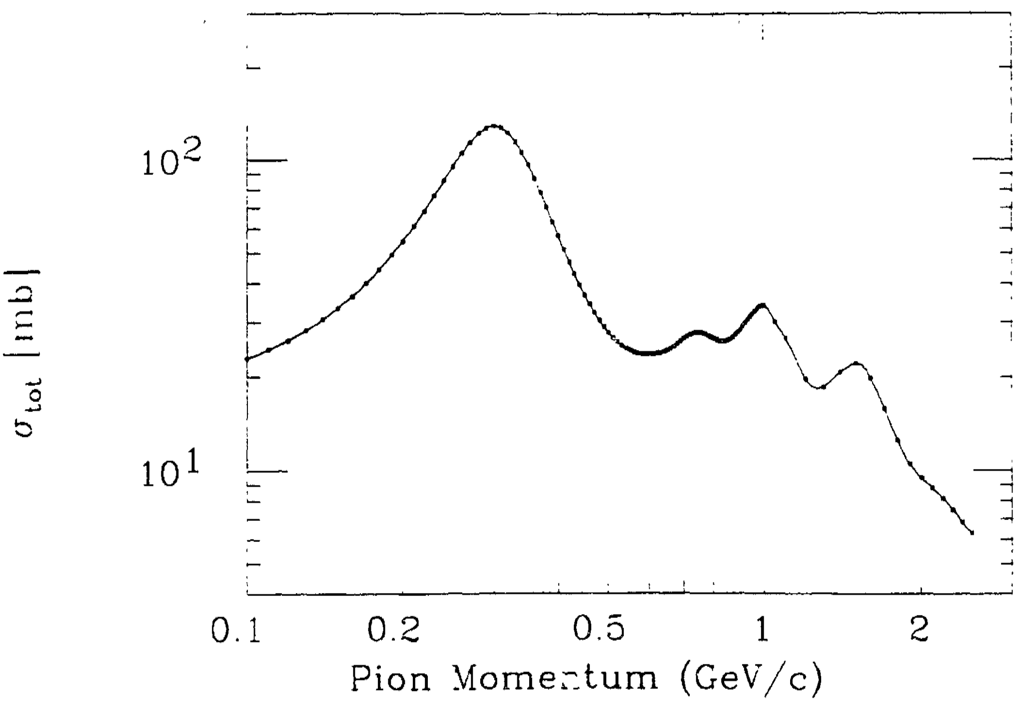


Figure III-7: The total calculated $\pi^0 n$ interaction cross-section.

the uncertainties in the decay parameters, E_0 and Γ: the Breit-Wigner formula appears to adequately duplicate the data. For several reasons, no attempt was made to perform a χ^2 fit of the calculated data to the experimental results, using E_0 and Γ as free parameters. First, the experimental data on these resonances, for both the total cross-section as well as the decay parameters, is sketchy and incomplete. It is not clear whether favoring the validity of the measured cross-section over the experimentally determined values of E_0 and Γ is justified. Secondly, and most importantly, adjustments of such a nature should not drastically effect the results of the PiNRP model or any simulation based upon it. With such arguments, the calculation of the π^+p and π^-p cross-sections, as well as their isospin-symmetric counterparts, π^-n and π^+n, within the PiNRP model is believed to be acceptable. The remaining cross-sections for π^0p and π^0n, shown in Figures III–6 and III–7, have been indirectly verified through a consideration of the resonances they form and their connection to the π^+p and π^-p cross-sections. They do not appear unrealistic, and demonstrate maxima where expected.

III–2.2 Results of the PiNRP Model

To explore the PiNRP predictions for pion-nucleon resonance formation, a canonical set of data was generated via the PiNBIN shell. In 0.1 GeV/c momentum bins over the pion momentum range of interest (0.7–2.0 GeV/c), data were separately generated for positive, negative, and neutral pions incident upon a distribution of equally numbered protons and neutrons. One million events were simulated for each pion species at each incident momentum, leading to 42 files summarizing the total final state particle distributions.

All semi-stable decay products (the η's, ρ's, and Δ's were all decayed prior to binning) mentioned in Table III-1 were counted and are included in the data which follow.

Figures III-8 through III-11 demonstrate the general characteristics of the model as a function of both pion species and incident momentum. The reliance of the model upon incomplete experimental data is vividly shown in Figure III-8, plotting the number of ill-determined decays occurring per event. This quantity climbs from less than 2% to nearly 50% within only a factor of three in momentum. The data for π^- and π^+ are effectively superimposed, while the π^0 data differ slightly from the others. To compensate for this large effect, all data in subsequent multiplicity distributions have been normalized with respect to valid decay events only. As a result, the counting error in quantities determined at large momenta, effectively proportional to $1/\sqrt{N}$, is a factor of $\sqrt{2}$ greater, although still amounting to less than 1%.

Figures III-9—III-11 show total particle multiplicity yields for incident positive, negative, and neutral pions, respectively. In each Figure, the average multiplicity of all charged and neutral products is given over the range of incident pion momenta spanning 0.7 to 2.0 GeV/c. The total multiplicity appears to be essentially independent of the incident pion species, predominantly flat while tapering off at low momenta. The charged particle multiplicities vary with both incident pion momentum and species, consistant with the requirement of charge conservation. These distributions are not surprising, given the assumptions of the model. Because an interaction is assumed to have occurred, the multiplicity is not expected to be proportional to the total pion-nucleon cross-section, or vary appreciably with momentum. In addition, most of the resonances decay with roughly

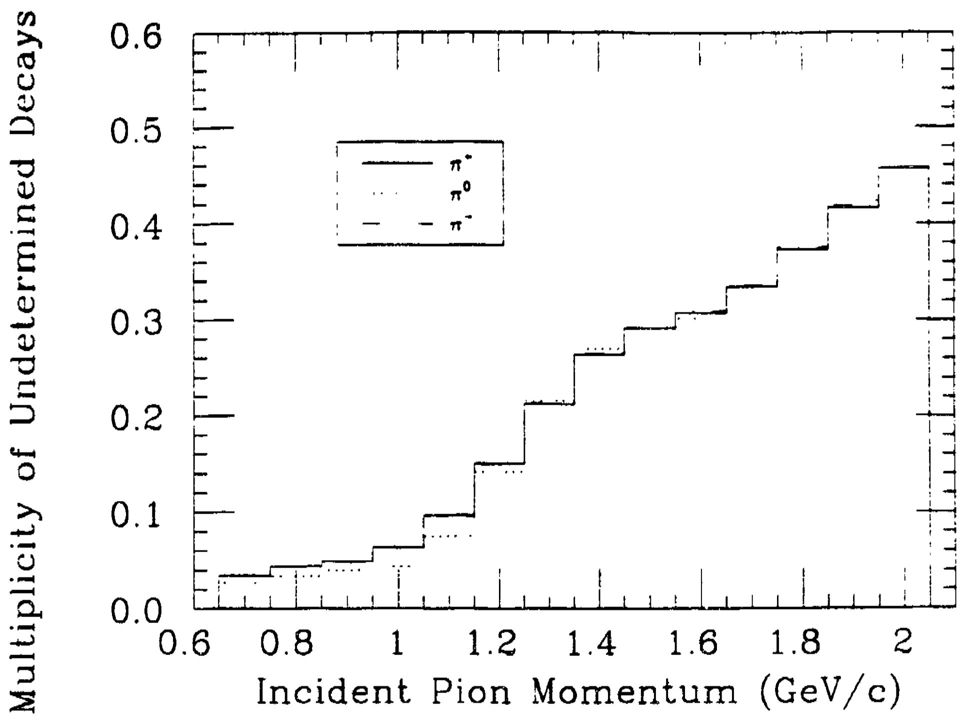


Figure III–8: The average multiplicity of ill-determined decays as a function of incident pion momentum and charge.

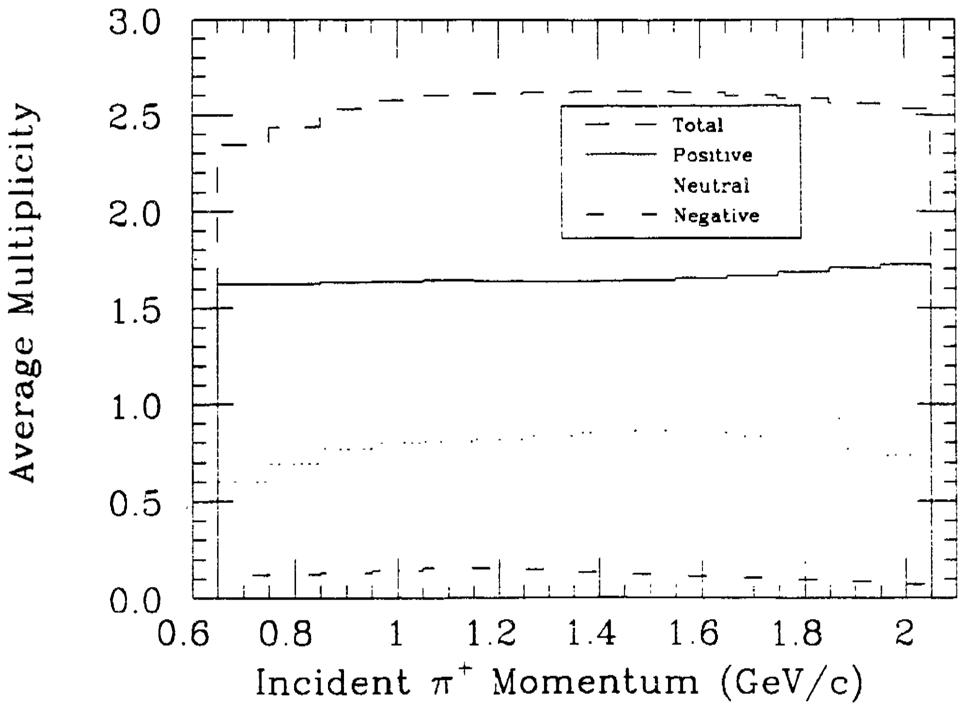


Figure III–9: Multiplicity of decay products for incident π^+ in the PiNRP model

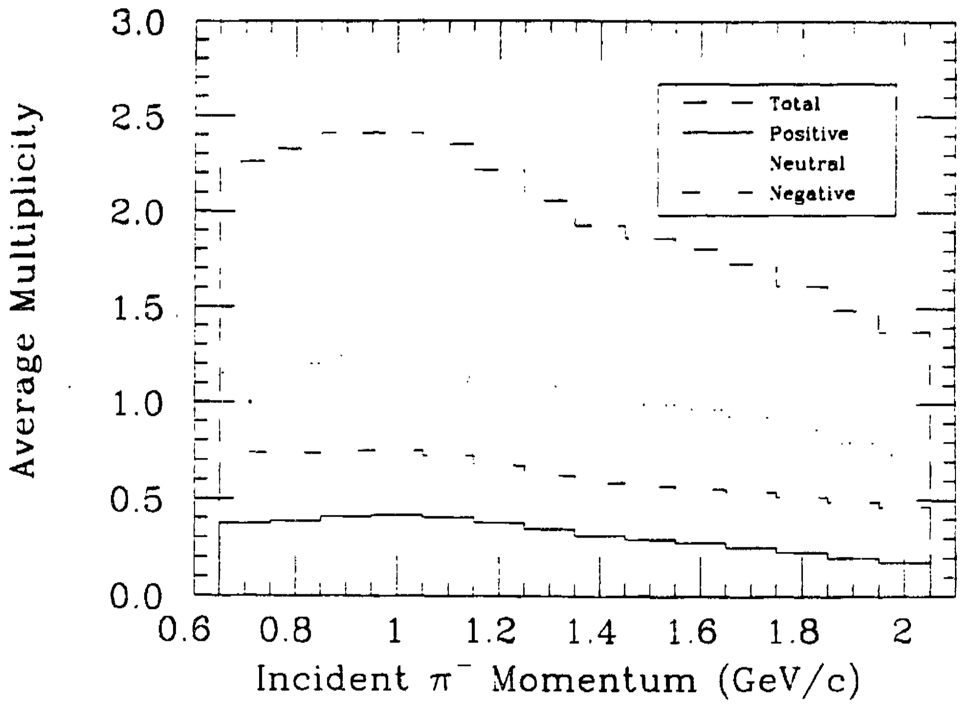


Figure III–10: Multiplicity of decay products for incident π^- in the PiNRP model

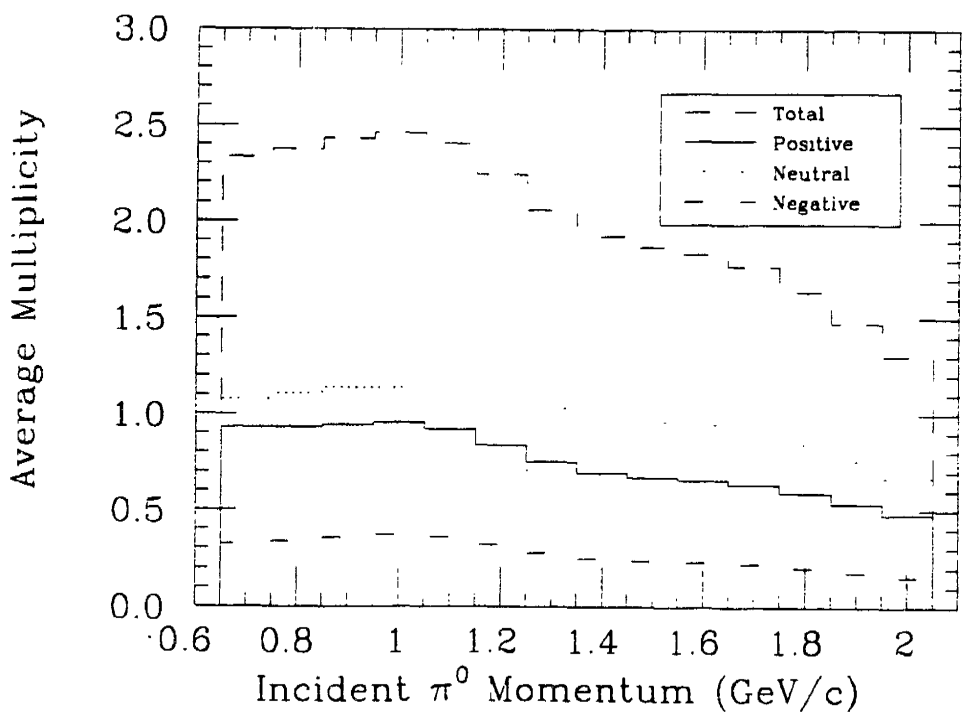


Figure III–11: Multiplicity of decay products for incident π^0 in the PiNRP model equal probability into either two or three-body channels. making an average multiplicity around 2.5 plausible. The lower total multiplicity at low momentum is a result of the fact that the prominant $\Delta(1232)$ resonance decays only via two-body channels. These results are more diagnostic than they are otherwise informative, but they do demonstrate that the model is capable of duplicating the expected aspects of the decay process.

The most physically revealing results obtained from the canonical PiNRP data are summarized in Figures III–12—III–17. As a set, this data characterizes the PiNRP predictions for strangeness production and enhancement through pion-nucleon resonance formation. Shown in Figures III–12—III–14 are the pion multiplicities as functions of the incident pion species and momentum. Unlike the previous figures, III–9 through III–11, which show the total particle multiplicity yield, Figures III–12—III–14 indicate solely the multiplicity of pions created through the elastic and multi-pion decay channels of the N^* and Δ resonances. As such, they measure the degree to which either the total pion multiplicity is lessoned or the π^+ decay mode is reduced in favor of negative or neutral channels. Both of these effects have important consequences upon the observed K^+/π^+particle ratio.

The information content of Figures III–12, III–13, and III–14 is similar, indicating that there is little dependence of the pion multiplicity upon the species of the incident pion. This is expected due to the assumed isospin invariance of the cross-section, and the reciprocal nature of the π^+N and π^-N systems. All of the multiplicity distributions show some momentum dependence, although the effect for produced π^0's is the greatest. The produced π^0 multiplicity distributions show a clear maximum near 1.5 GeV/c, while

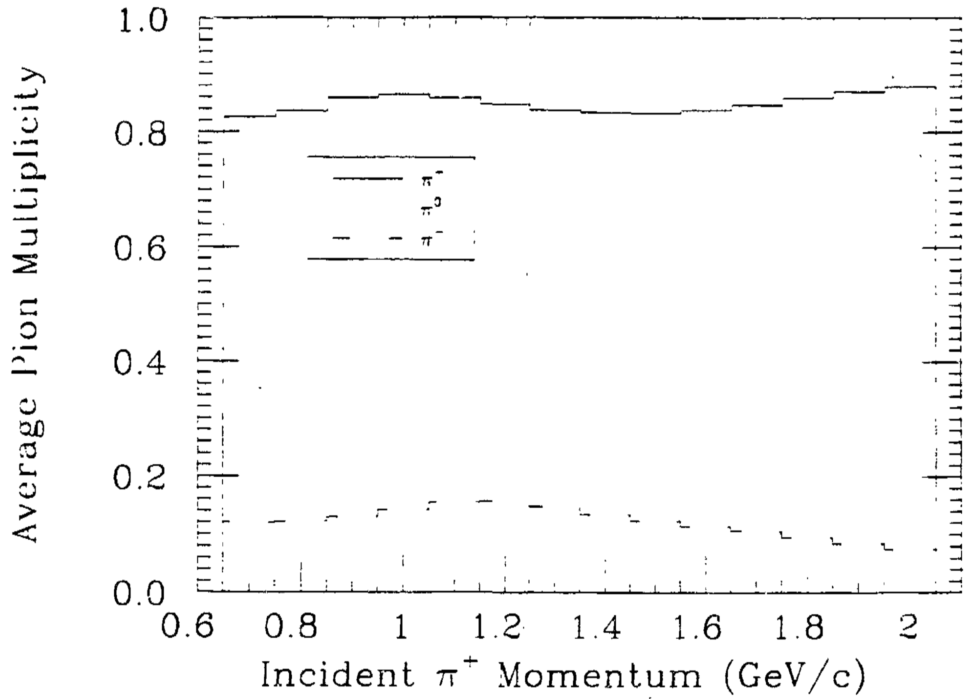


Figure III–12: Multiplicity of produced pions for incident π^+ in the PiNRP model

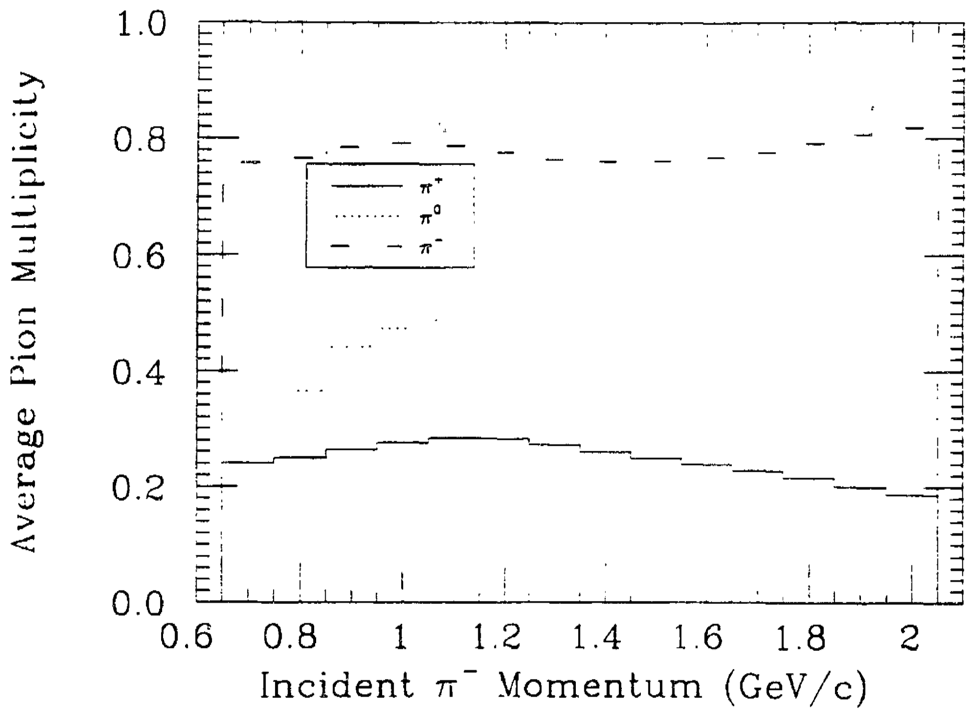


Figure III–13: Multiplicity of produced pions for incident π^- in the PiNRP model

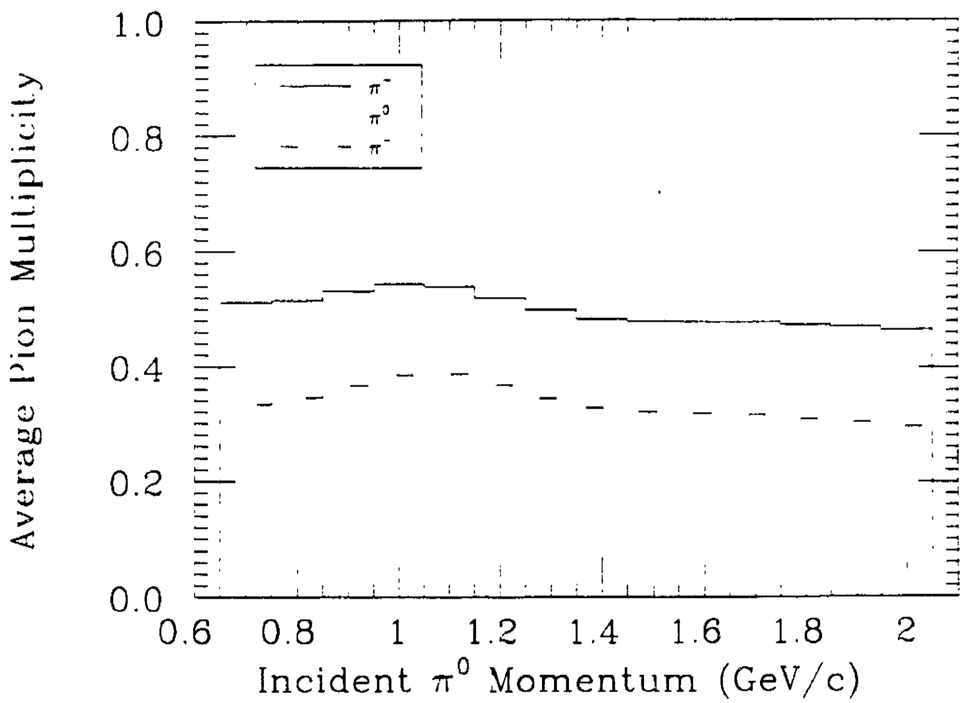


Figure III–14: Multiplicity of produced pions for incident π^0 in the PiNRP model

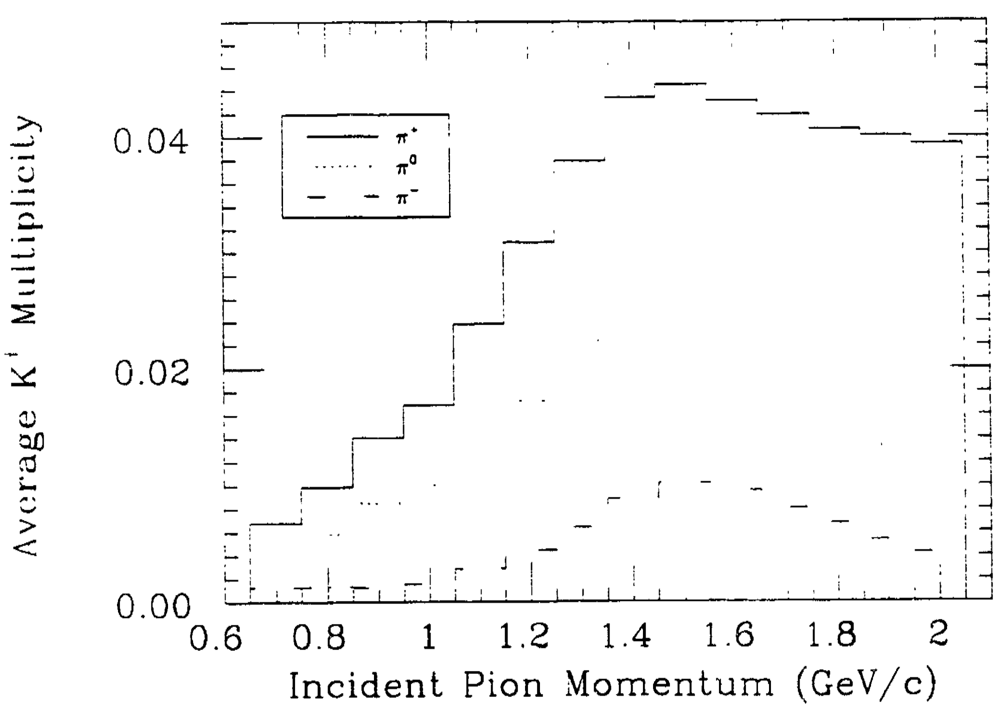


Figure III–15: Multiplicity of produced K^+ in the PiNRP model the same distributions for π^+ and π^- show only a weak dependence upon momentum and a slight maximum near 1.1 GeV/c. As can be seen in comparison with Figure III–15, the π^0 multiplicity distributions have some correlation with the general shape of the K^+ multiplicity. No *a priori* reason can be given, however, to link their individual production. What can be determined from these distributions, however, is that there is no clear reduction in either the positive or total pion multiplicities over the momentum range studied. Thus, any change in the K^+/π^+ ratio must be solely due to an increase in the number of produced kaons, and not to an overall decrease in the number of positive pions.

As already mentioned, Figure III–15 details the K^+ multiplicity dependence upon the momentum and charge of the incident pion. A few observations are immediately evident from this data. First, pion-nucleon resonance production does result in significantly greater K^+ multiplicities for pion momenta exceeding 1.0 GeV/c. The production of kaons increases sixfold in a doubling of momentum, peaking at 1.5 GeV/C and only gradually lessoning for greater momenta, although continuum resonances could this trend to change. While the greatest branching ratio for the production of kaons occurs in the decay of N(1710) at an incident pion momentum of 1.10 GeV/c, the density of kaon-producing resonances increases with momentum, causing the multiplicity to peak at a significantly higher 1.5 GeV/c. Figure III–16 shows the total cross-section for K^+ production, as compared with the total π^+p cross-section. In essence, this kaon cross-section is the branching ratio determined from Figure III–15 multiplied by the total cross-section at a given momentum. Values are not shown for cross-sections less than 100 μbarns.

Lastly, Figure III–17 shows the model-predicted K^+/π^+ ratio. Because the pion multiplicity distributions were essentially flat. the data in Figure III–17 is to a great degree just a rescaling of Figure III–15. The shape of the K^+ multiplicity distribution clearly dominates the resulting K^+/π^+ ratio as a function of momentum. For all incident pion species, the K^+/π^+ ratio appears to peak at 1.5 GeV/c, with a maximum value of nearly 6% for neutral pions.

The entire shape of this and the other multiplicity distributions is, however, not well determined for $p > 1.8$ GeV/c. Due to the poor data existing for the more massive pion-nucleon resonances, the distributions derived from PiNRP may deviate significantly from the true physical reality, especially where nearly 50% of the decay products are unknown. Because only 5-10% of the decay channels for some resonances are known, and the branching ratios for kaon production are generally low for the known resonances in this region, the renormalization of the results for well-determined decays to 100% has the effect of enhancing the relative importance of the kaon channels.

For example, the average branching ratio for decay channels producing kaons at momenta greater than 1.4 GeV/c is roughly 0.3%. If, however. only 10% of the decay channels are known, and the results are normalized to 100%. kaons have an *effective* branching ratio of 3%, an order of magnitude greater than it, in reality, may be. The effect is greatest for kaons because the remaining 90% of the decay channels unknown to the model most likely include pion products, and thus the renormalization should introduce only a small relative error into the total pion branching ratio. Because the proportion of known kaon decays is small, and it is unlikely that they exist in large

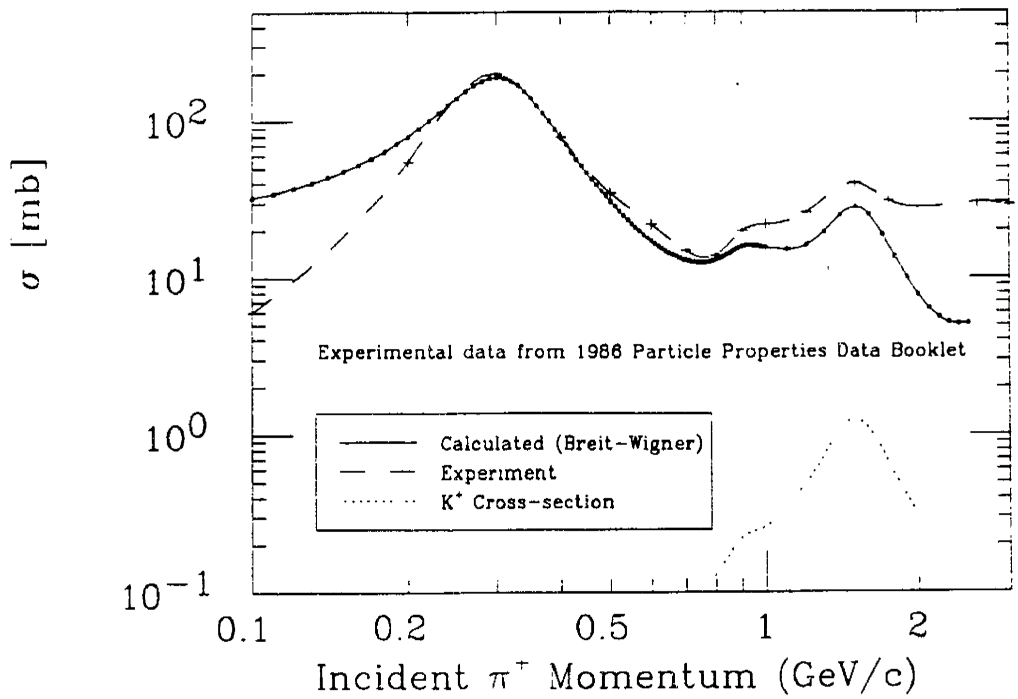


Figure III–16: Total cross-section for K^+ formation in the PiNRP model

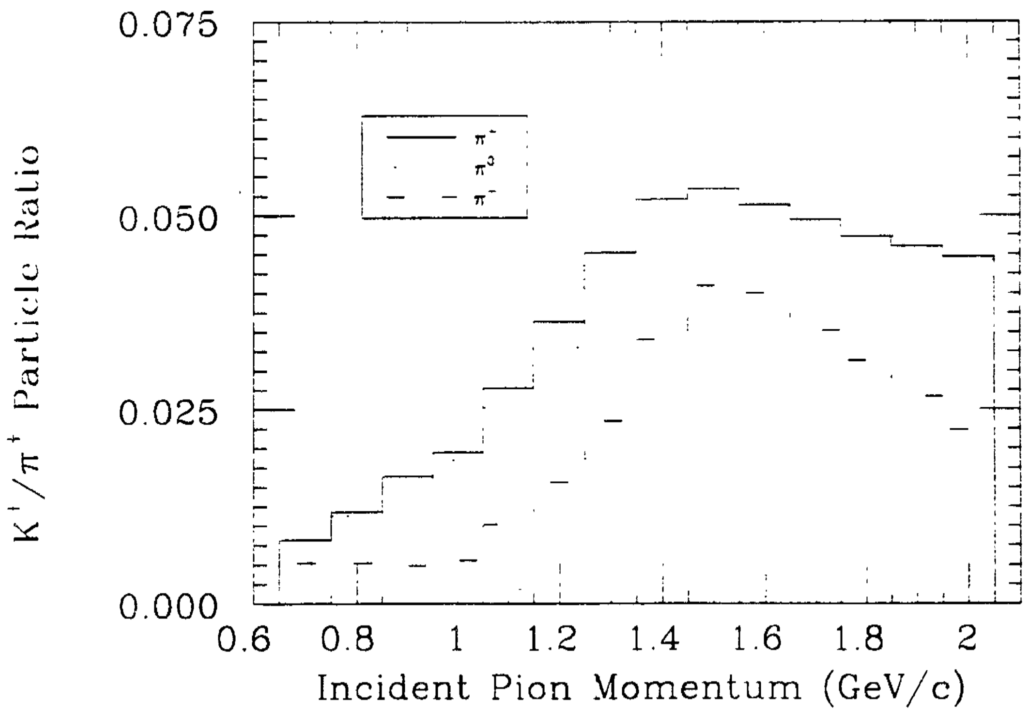


Figure III–17: K^+/π^+ ratio predicted by the PiNRP model proportion, relative to pions, in the unknown decay modes, the renormalization thus results in the unphysical enhancement of the produced kaon multiplicity.

It is unfortunate that the effect of undetermined decays should be so great, but the choices available for dealing with uncertainties of this magnitude are equally unattractive. Normalizing with respect to the total number of events, good or bad, yields the unphysical result of multiplicity distributions that tail steadily downwards with increasing momentum. As kaons are more sensitive to any normalization effect, because their branching ratios are low and thus subject to a greater relative error, it would seem that much of the significance in their data has been lost. The data presented, therefore, probably represent an upper bound to the K^+ multiplicity. Only with more complete data on the massive pion-nucleon resonances can the PiNRP model yield truly informative predictions.

III–3 Conclusions

Due to the overwhelming uncertainties discussed in Section III–2.2, it is difficult to interpret the predictions of the PiNRP model. Clearly, the model is capable of producing results physically consistent with intuition for a limited range of momenta, but the fraction of undetermined decays becomes significant enough at high momentum to wash away most, if not all, of the information content in the data. As discussed earlier, the K^+ multiplicity is perhaps most sensitive to these problems, while the pion multiplicities and the K^+/π^+ ratio data are undoubtedly effected to a lesser degree. Unfortunately, the magnitude of these errors cannot be adequately quantified. The K^+/π^+ ratio shown in Figure III–17 represents what is likely an upper bound on the actual value, simply because it is more likely that the unresolved decays produced additional pions rather than kaons.

The available resonance data is clearly inadequate for use by the model, insomuch as it represents an accurate summary of existing experimental measurements. The single greatest improvement that can be made to the PiNRP model would be the inclusion of more accurate and extensive pion-nucleon resonance data. Despite this deficiency, however, the model does provide some valuable information about resonance formation, and yields what is, arguably, an upper bound to the total K^+/π^+ ratio attainable via pion-nucleon resonance production and decay.

Chapter IV

The NNEveSim Model

To perform direct comparisons with the existing E-802 data, the program `NNEveSim` was designed to apply the `PiNRP` model in a simulation of nuclear collisions. The basic framework of the program is given in Figure IV–1. `NNEveSim` models a general nuclear collision as a series of incident nucleons individually propagating through a one-dimensional tube of nuclear matter. When an interaction is found to have occurred, through a Monte Carlo of the nucleon's mean free path, a single secondary particle is taken from an external list and allowed to interact with the target nucleons before leaving the nucleus. While making simplistic assumptions on an event by event basis, `NNEveSim` should return the proper final state particle distributions when averaged over a large number of events.

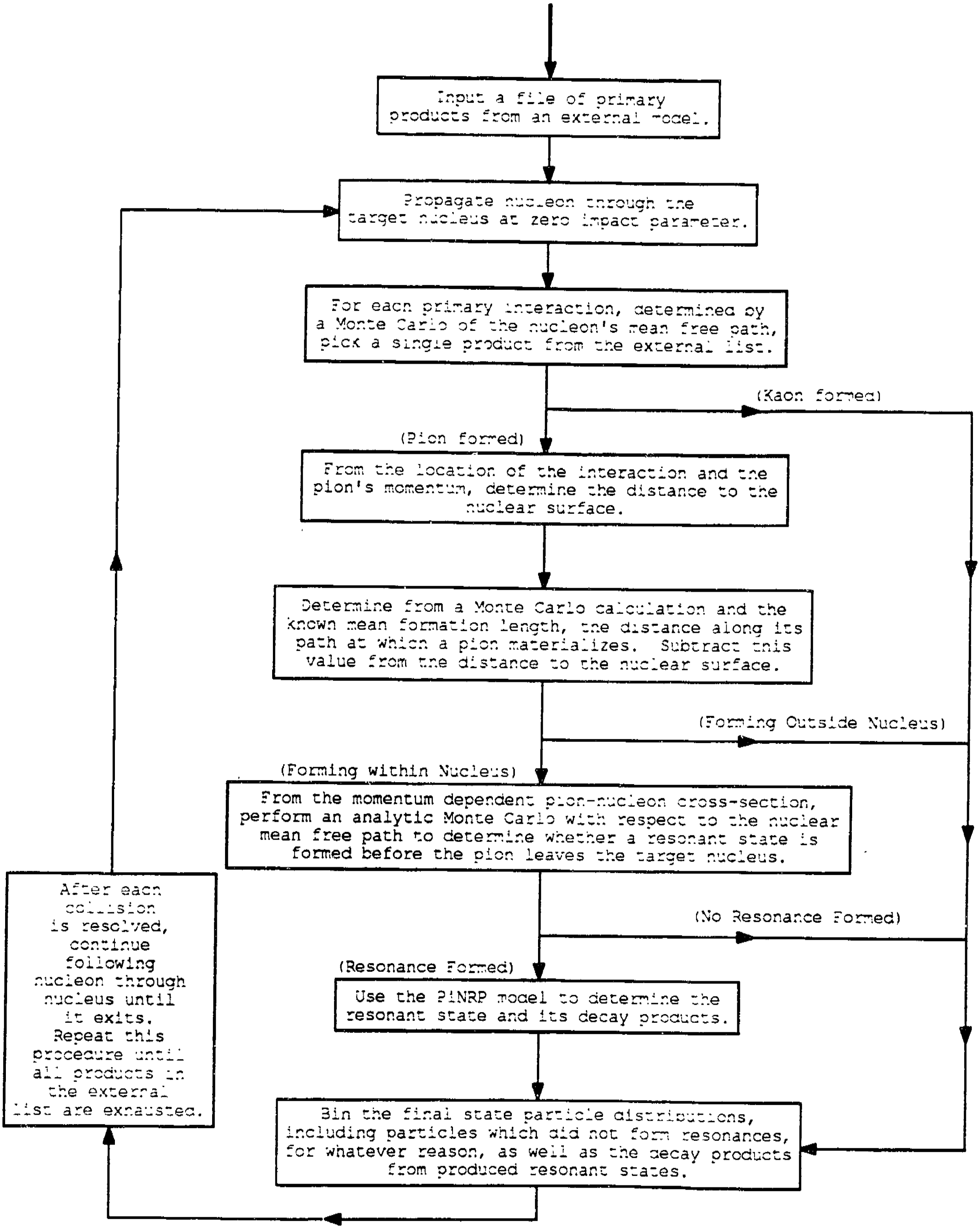


Figure IV–1: Structure of the NNEveSim Model

IV–1 The Simulation of a Typical High-Energy *AB* Collision

The NNEveSim simulation provides a simple model of pion-nucleon resonance formation in nuclear collisions. The model will work for any general high-energy nucleus-nucleus reaction, although the data presented herein was generated with parameters designed to correspond with the 14.5 A·GeV/c Si-Au reactions studied by E-802. The NNEveSim model breaks down a single, highly asymmetric, *AB* collision into a series of nucleon-nucleus interactions, all at zero impact parameter. In addition to simplifying the many-body processes, this scheme should provide the greatest probability for secondary pion rescattering, and thus should yield results that represent an upper bound to the effect of pion-nucleon resonance production on final state particle distributions. For each projectile nucleon, a Monte Carlo calculation simulates propagation through nuclear matter of uniform density. When a collision occurs, a single secondary particle is taken from a list of particles created by the LUND [4] event generator. Both the incident nucleon, as well as the produced particle, are allowed to further interact with the nucleus before they exit. If a pion-nucleon resonance is formed, the PiNRP model is used to determine which resonances and decay products result.

IV–1.1 Primary Reactions and Their Products

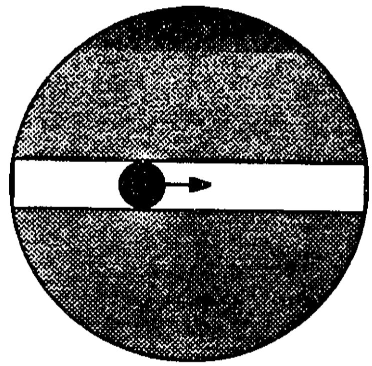
Figure IV–2 shows a schematic of the nuclear reaction process, as simulated by NNEveSim. In essence, the projectile nucleus as a whole is never considered by the NNEveSim model.

As seen in Figure IV–2a, a single nucleon is traced through the target at zero impact parameter. Interactions of this nucleon with nucleons from the target are the sole means of primary product creation within the model. Assuming that the target is of a uniform, ground state baryon density of 0.17 baryon/fm^3, and an inelastic nucleon-nucleon interaction cross-section of 32 mb, a mean free path of 1.95 fm is used to Monte Carlo the distance traveled between interaction. The probability of the nucleon traveling a distance l is given by,

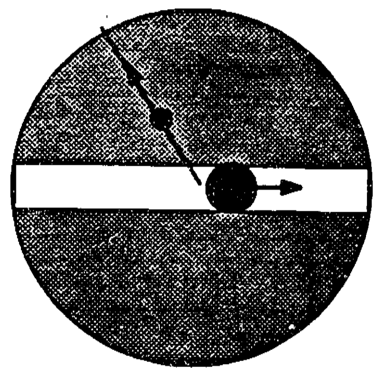
$$p(l) = e^{-l/l_0} \qquad (IV-1)$$

where l_0 is the mean free path. The model selects a random number between 0 and 1 for the value of $p(l)$ and inverts the expression in Equation IV-1 to solve for l. If this added distance does not place the nucleon outside of a one dimensional tube (shown in Figure IV–2a) having a length equal to that of the target's diameter, the nucleon's position is updated, and a primary collision is assumed to have occurred.

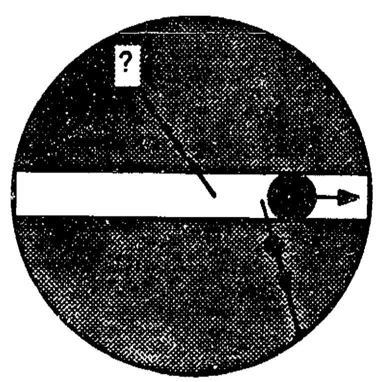
When a primary collision occurs, as shown in Figure IV–2b, a single secondary particle is formed and both it and the incident nucleon propagate through the target until they exit the nucleus. The product is taken from a database created by the LUND [4] event generator. Rather than complicate the NNEveSim model unduly, the simulation was designed for an external model to provide the results of the primary collisions. Although the LUND model, as a standard in the E-802 collaboration, has been used in this capacity, it is conceivable that other models may serve the same purpose. The LUND model was used to simulate 2,500 zero impact parameter Si-Au collisions at an incident projectile



a) Given projectile nucleon is propagated through a 1—D tube of nuclear matter at zero impact parameter..



b) The nucleon's mean free path in uniform nuclear matter is used to determine where an interaction occurs. Products are chosen from a precalculated list, one by one, and allowed to propagate through the target nucleus, possibly forming resonant states.



c) The incident nucleon continues its path, possibly creating one or more additional particles.

Figure IV–2: Sketch of the Propagation of a Projectile Nucleon in the NNEveSim model.

momentum of 14.5 A·GeV/c. From the resulting output, a compact list of all of the produced kaons and pions was used as the secondary particle database, containing the particle identification, its total momentum and energy, and the three components of momentum with respect to the beam axis. The momentum spectra for LUND generated pions is shown in Figures IV–3—IV–5.

One advantage of this approach, besides divorcing the `NNEveSim` model from any specific theory of heavy ion collisions, is the resulting speed which results from having all of the primary products precomputed. Unfortunately, because LUND does not give spatial coordinates for the production point of a produced particle, a one-to-one event correspondence could not be made within the `NNEveSim` model. Thus, for each primary collision, a single secondary particle is used. An obvious objection to this procedure is the lack of energy and momentum conservation at each primary collision. The projectile nucleon continues its forward trajectory uninhibited (The momentum of the projectile nucleon is irrelevent as it is always assumed to have a nucleon-nucleon cross-section of 32 mb, independent of momentum and the number of collisions. This is true for all accessable momenta greater than 1.0 GeV/c.), while the product leaves at an arbitrary angle, energy, and momentum (Figure IV–2c). The procedure is best not thought of as a valid visualization of the reaction dynamics, but rather as a consistent and physical means of generating a production site for each particle in the list. The event by event nature of the collision is not preserved, although the general physics should emerge on the average. Specifically, because only particle *ratios* are required, and they are well-defined only in the limit of a large number of events, the simplifications in the `NNEveSim` model

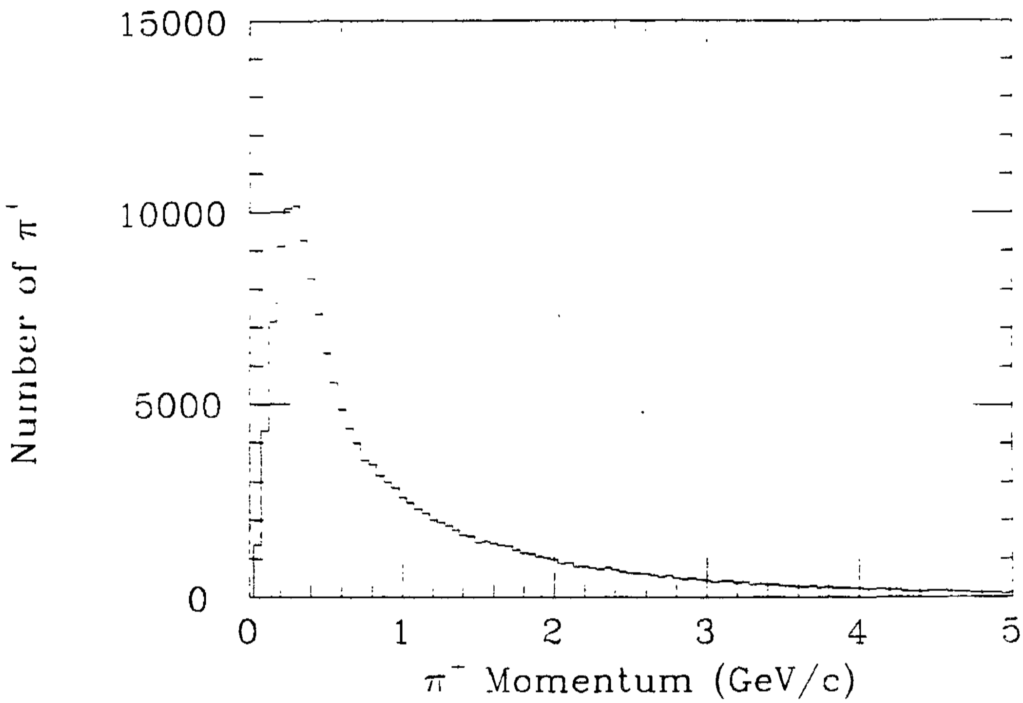


Figure IV-3: LUND π^+ Momentum Distribution

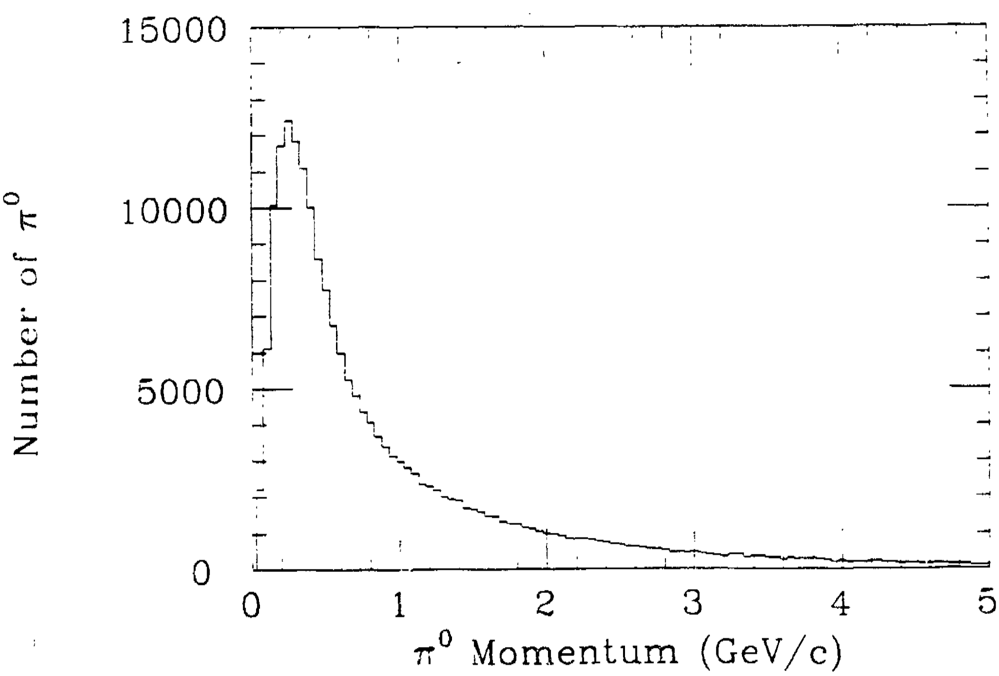


Figure IV-4: LUND π^0 Momentum Distribution

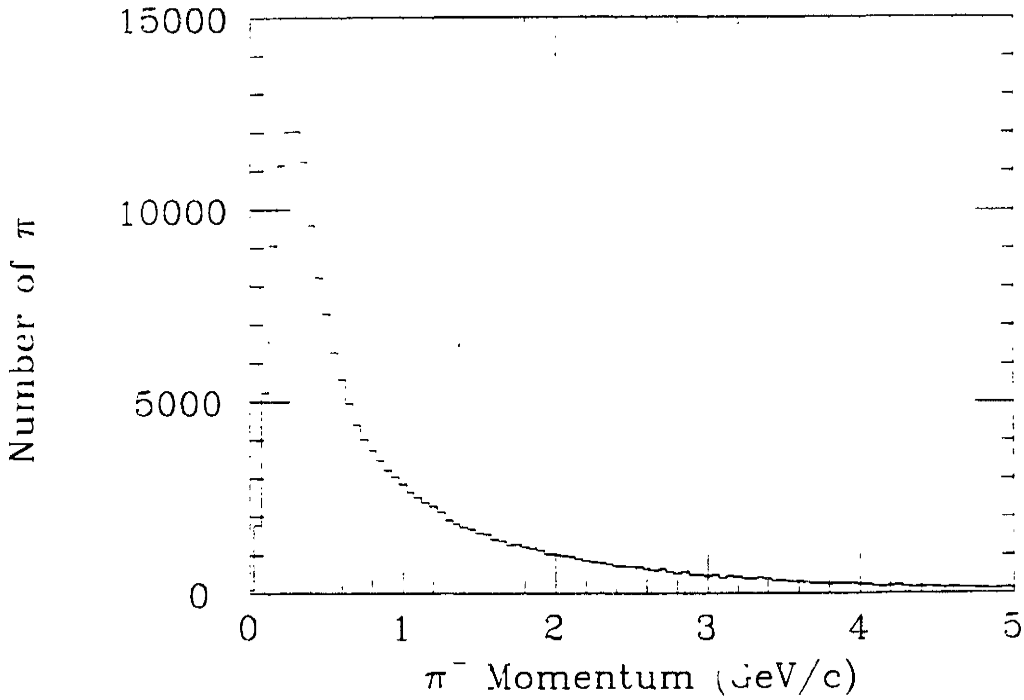


Figure IV–5: LUND π^- Momentum Distribution

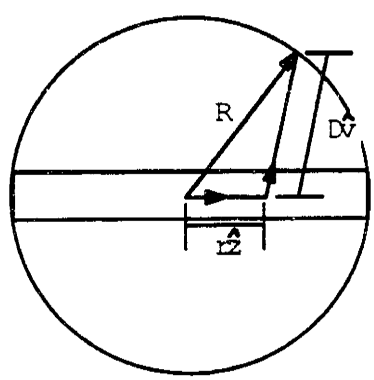


Figure IV–6: Geometry for Distance-to-Surface Calculation do not invalidate its results.

Once a product has been formed, the components of its momentum are used to determine the distance that it must travel before leaving the nucleus. Figure IV–6 depicts the quantities that are used in this calculation. The vectors $\vec{R}$ and $\hat{z}$ are defined from the center of the nucleus: $\vec{R}$ to the point on the nuclear surface where the produced particle will exit, if no resonances are produced, and $\hat{z}$ along the beam direction which the incident nucleon propagated. The vector $\hat{v}$ is a unit vector in the direction of the particle's velocity vector. It is thus a consequence of geometry and vector algebra that,

$$\left|\vec{R}\right| = |r\hat{z} + D\hat{v}| \qquad \text{(IV}-2\text{)}$$

where r is the radial distance (positive or negative) to the site of the particle's production, and D is the desired distance to the nuclear surface. Taking the magnitude of Equation IV-2,

$$R^2 = r^2 + 2rD(\hat{z}\cdot\hat{v}) + D^2 \qquad \text{(IV}-3\text{)}$$

But, we can express $\hat{v}$ in terms of the known components of momentum,

$$\hat{v} = \left(\frac{P_x}{P}\right)\hat{x} + \left(\frac{P_y}{P}\right)\hat{y} + \left(\frac{P_z}{P}\right)\hat{z} \qquad \text{(IV}-4\text{)}$$

Thus, Equation IV-3 reduces to the following,

$$D^2 + \frac{2rP_z}{P}D + (r^2 - R^2) = 0 \qquad \text{(IV}-5\text{)}$$

Equation IV-5 is a quadratic equation in D, which for $r \leq R$ always yields a single positive real root. Given the solution for D, the distance to the nuclear surface, the problem of propagating the produced particle through the nucleus reduces to a one dimensional Monte Carlo of the particle's mean free path.

IV–1.2 Secondary Pion Rescattering

Once a secondary particle has been produced and the distance it must travel to the nuclear surface has been calculated, two additional steps are necessary to determine whether a resonance state is formed. Only produced pions are considered by the `NNEveSim` model in the determination of secondary interactions. As the need to conserve strangeness limits kaon-nucleon reactions at typical produced kaon energies to primarily elastic rescattering and K^- absorption, positive kaons produced in the LUND simulation are assumed to contribute to the final state particle distributions unaffected by intermediate interactions. Produced pions, however, undergo two Monte Carlo calculations to determine if they interact to form pion-nucleon resonances.

First, the `NNEveSim` model determines the formation length necessary for the pion to materialize along its flight path. The formation time, for the model's purposes, is defined from the quantum mechanical uncertainty relation for time and energy,

$$\Delta E \Delta t \geq \hbar \qquad \text{(IV} - 6\text{)}$$

If one considers ΔE the rest mass energy of a produced particle, Δt is defined as the time it takes for that particle to identifiably appear along its trajectory. Assuming that the particle is traveling at the speed of light, which introduces neglible error in the model for particles of high β, an equivalent distance can be assigned to this uncertainty. The formation time and length, of course, are most naturally given in the rest frame of the product; they must be multiplied by γ to be valid in the laboratory frame. The probability of a particle, of formation length l_{form}, materializing at a distance of l along its path is given by,

$$p(l) = e^{-l/l_{form}} \qquad \text{(IV}-7)$$

For the distance l in which the composite particle does not truly exist, it will not interact as a composite particle of known cross-section with the nuclear matter through which it is traveling. Once a pion identifiably materializes, it then propagates with a momentum dependent cross-section through any remaining portion of the nucleus. The formation length thus is of profound consequence to the secondary production of pion-nucleon resonances. For high values of γ, produced pions may form entirely outside the nucleus, having traveled the distance between their point of creation and formation as if in free space. Even for moderate values of γ, the distance of nuclear matter through which a pion must propagate with a chance of interaction may be significantly reduced.

If the produced pion has any further distance to travel within the nucleus after it materializes, the `NNEveSim` model relies upon the `PiNRP` model to determine the momentum dependent cross-section for resonance formation. The version of `PiNRP` used in `NNEveSim` differs slightly from that described in Chapter III. The modified version of

PiNRP used in NNEveSim does a Monte Carlo calculation of the distance traveled by a pion, once the interaction cross-section has been determined, and compares it with the total distance to the nuclear surface. If greater, no interaction occurs, and the pion is added to the list of final state particles. If the distance traveled is less than the distance to the edge of the nucleus, a resonance is formed and the PiNRP model returns the decay products. Three sets of data were generated from the LUND input file, two calculated with the full simulation and one with the option to form resonances disabled. The two full simulation data sets differ in their treatment of the target nucleons' momenta. Whereas in one set, the momentum of the target nucleons was ignored, the second added a Monte Carlo calculation to include a Fermi momentum distribution. Using the ground state, absolute zero, Fermi momentum distribution for the target nucleons,

$$\mathrm{p}(p) = \begin{cases} \frac{1}{p_f} = \frac{1}{28\mathrm{MeV}} & \text{for } p < p_f \\ 0 & p > p_f \end{cases} \qquad \text{(IV} - 8)$$

an arbitrary momentum was chosen and projected on, and added to, the pion's momentum. The resulting relative momentum was used in the computation of the pion-nucleon cross-sections. As indicated in Chapter III, the PiNRP model does not adequately represent the pion-nucleon cross-section for momenta much greater than 1.0 GeV/c. This should not introduce much error in the NNEveSim results, however, as pions of this energy have a relativistic γ of such magnitude to cause a vast majority to form outside of the nuclear volume.

If a resonance is formed, the decay products are not allowed to reinteract with the target nucleus, but are instead immediately binned in the final state particle histograms. For this reason, no attempt was made to determine the energy or momenta of decay products. The model thus returns the total, angle integrated, number of kaons and pions produced in the 2,500 LUND simulated Si-Au collisions processed by `NNEveSim`.

IV–1.3 `NNEveSim` Code Implementation

The `NNEveSim` model was written in VAX `FORTRAN` and is contained in Appendix C. The code is rather simple, with the majority of the work being provided by either the LUND or `PiNRP` models. The model loops through nucleons, propagating them through target nuclei until the produced particle list provided by LUND is exhausted. All noninteracting primary products and those secondary products produced via resonance decays are binned in the final state particle distributions. While over nine hours of CPU time were required to generate the 2,500 LUND events, just under one hour of CPU was required for `NNEveSim` to process them.

IV–2 Results and Discussion

Table IV–1 shows the raw particle distribution generated by LUND without processing with `NNEveSim`. A total of 573,726 pions and kaons resulted from the zero impact parameter LUND Si-Au simulation at 14.5 A·GeV/c. A K^+/π^+ ratio of $2.75 \pm 0.04\%$ results from this simulation. The data for `NNEveSim` is found in Tables IV–2 and IV–3, with and without the Fermi momentum, respectively. The inclusion of the nucleons' Fermi momentum appears to have no discernable affect upon the final state particle distributions, and both NNEveSim results yield a K^+/π^+ ratio of $2.57 \pm 0.04\%$, actually *lower* than the raw LUND value. An examination of the data in Tables IV–2 and IV–3, and a consideration of the results shown in Chapter III, demonstrate that this low K^+/π^+ result is consistent with both the PiNRP and NNEveSim models.

Because NNEveSim yields similar results both with and without an added Fermi momentum distribution, the data in Table IV–2 alone will be discussed for consistency. In addition to the total final state particle distribution predicted by NNEveSim the Table contains the relative contributions from secondary resonance production as well as particles originating directly from the primary nuclear collision. Column 2 gives the total number of pions that were rescattered, while column 3 shows the resulting particle yields. For pions not forming resonances, columns 4 and 5 show the number materializing inside and outside of the nuclear volume, indicating whether the failure to form a resonance was primarily due to the formation length of the particle or to its interaction cross-section. As kaons were not allowed to interact, they do not contribute to columns 2, 4, or 5. For pions, the sum of columns 3, 4, and 5 equals the final state value given in column 1, and the sum of columns 2, 4, and 5 yields the values in Table IV–1. For kaons, the value in column 3 indicates the excess number produced above the values given in Table IV–1. With this information, the data summarized in Table IV–4 has been generated.

In the first column of Table IV–4, the percentage of pions that materialized outside of the nuclear volume is given. The second column indicates the percentage of those

Table IV–1: Raw LUND Particle Distributions for 14.5 A·GeV/c Si+Au

Particle Type	*Number Produced*
π^+	167,699
π^0	197,081
π^-	189,801
K^+	4,620
K^-	1,204
K^0_{long}	6,613
K^0_{short}	6,708

Table IV–2: NNEveSim Results With Fermi Momentum

Particle Type	*Total Final State Distribution*	*Pion Rescatters* Number of *Interacting Particles*	*Resulting Products*	*Pion Does Not Rescatter With Pion Materializing* *Inside Nucleus*	*Outside Nucleus*
π^+	198,465	60,669	93,847	26,811	77,807
π^0	185,553	75,310	66,132	28,487	90,934
π^-	187,901	71,373	72,026	29,951	85,924
K^+	5,103	–	483	–	–
K^-	1,204	–	–	–	–
K^0_{long}	6,820	–	262	–	–
K^0_{short}	7,008	–	245	–	–

Table IV–3: NNEveSim Results Without Fermi Momentum

Particle Type	*Total Final State Distribution*	*Pion Rescatters* Number of *Interacting Particles*	*Resulting Products*	*Pion Does Not Rescatter With Pion Materializing* *Inside Nucleus*	*Outside Nucleus*
π^+	198,970	60,408	94,088	26,806	78,076
π^0	185,229	75,467	65,950	28,175	91,104
π^-	187,793	71,337	71,874	30.098	85,821
K^+	5,115	–	495	–	–
K^-	1,204	–	–	–	–
K^0_{long}	6,990	–	269	–	–
K^0_{short}	6,872	–	272	–	–

Table IV–4: Reaction Mechanism Summary for Primary Pions

Particle Type	*% of Pions Forming Outside of the Nucleus*	*% of Pions Forming Inside the Nucleus that Rescatter*	*Net % Change in Particle Number*
π^+	46.4	69.4	+54.7
π^0	46.1	72.6	-12.2
π^-	45.3	70.4	+0.9

pions materializing within the nucleus that also formed resonant states. Both of these values appear rather constant between the different charge species, although they are not entirely independent. Because the proportion of pions forming outside the nucleus is considerable, it is evident that the relativistic enhancement of the formation length is sufficient to substantially reduce the remaining path length of those pions that form within the nucleus. The formation length of the pions thus seems to limit any effect that pion-nucleon resonance formation may have upon the final state particle distributions. From the third column of Table IV–4, however, it is clear that the resonances that are formed do have a profound effect on the number of produced pions. In particular, there is an enhancement of over 50% in the number of π^+ that are formed. From Table IV–2, we see that only approximately 10% more K^+ are produced by resonance formation and decay, and thus the excess production of pions is actually responsible for the NNEveSim prediction of K^+/π^+ *suppression* from the secondary formation of resonant states. This is consistent with the $N\pi\pi$ decay channels dominating over strange particle production.

IV–3 Conclusions

Given the results presented in Section IV–2, and the previous data obtained by the PiNRP model (*c.f.* Chapter III), it is apparent that these models do not support the hypothesis that secondary pion-nucleon resonance formation can explain the K^+/π^+ ratio enhancement seen by the E-802 collaboration [20]. The formation length of typical produced pions simulated by the LUND particle generator actually suppresses any effect that may be attributed to resonance formation by roughly a factor of two. In addition, the existence of $N\pi\pi$ decay channels not only nullifies the additional production of kaons, but is actually of sufficient magnitude to *lower* the K^+/π^+ ratio. The raw LUND data gave a K^+/π^+ ratio of $2.75 \pm 0.04\%$, while the NNEveSim predicts $2.57 \pm 0.04\%$. Clearly, the absolute value that NNEveSim yields is highly dependent upon the LUND model; it is the relative effect which is important. While the PiNRP and NNEveSim models themselves are simplistic, and thus cannot be expected to predict the physics with great accuracy, they nonetheless include the essentials requisite to simulate resonance production. With a reasonable degree of confidence, therefore, they give results that differ so vastly from the observed data given in Section II–4 that it appears unlikely that pion-nucleon resonance production can explain strangeness enhancement at E-802 energies.

Several enhancements to the NNEveSim model are possible, although it is unlikely that they should alter the previous results significantly. First, providing the energy and momenta of resonance decay products would allow for examination of the K^+/π^+ ratio in distinct sections of phase space. Comparison with data might shed more light upon the applicability of the **NNEveSim** model and alter the credibility of its predictions. A second improvement that was considered, although never implemented, would be an alteration of the model to accept a more physical description of the reaction process. Explicitly, the addition of a relativistic co-moving frame to the model of pion propagation, as well as a reaction volume more appropriate than an isolated, ground state target nucleus, would add considerable complication to the model, but would lesson the limitations which the current model retains. The last change that can be made, most easily, is the use of an alterate model as the front-end generator of primary particles. These enhancements might alter the results somewhat, although it is doubtful that they will completely reverse the previous conclusions.

Chapter V

Summary

The PiNRP and NNEveSim models do not support the claim that the enhanced K^+/π^+ ratio seen in the E-802 14.5 A·GeV/c Si-Au data can be attributed to pion rescattering in the target nucleus. The PiNRP model predicts a maximum K^+/π^+ ratio, in the limit of infinite cross-section, of $\simeq$ 6% at a pion momentum of 1.5 GeV/c. This result is subject to a certain, though unquantified, error due to the poor data available for the pion-nucleon resonances at and above 1.8 GeV/c. Since the undocumented decay modes should primarily create pions, and very few additional kaons, the K^+/π^+ ratio can be expected to be lower than the reported values, especially at high momentum ($\geq$ 1.8 GeV/c). From a full simulation of 14.5 A·GeV/c Si-Au collisions, NNEveSim predicts a net, albeit small, *decrease* in the K^+/π^+ ratio due to the effects of pion rescattering. A value of $2.57 \pm 0.04\%$ was found for K^+/π^+, integrated over all angles, in comparison with the value of $2.75 \pm 0.04\%$ from the raw LUND data. Just fewer than 50% of the pions were found to materialize outside of the nuclear volume, the relativistic time dilation of highly energetic pions thus suppressing the effect of pion rescattering by roughly a factor of two. Because, however, the mean free path of those pions forming within the nucleus is on the order of only 0.50 fm, nearly 70% of these rescattered to produce a highly altered final state particle distribution. A net increase of 55% was observed in the number of produced π^+'s, overshadowing the 10% gain in the total number of K^+. Although pion rescattering does affect the final state particle distributions, it does not explain the 19–24% K^+/π^+ ratio observed experimentally [20].

The interpretation of the E-802 data in light of the `PiNRP` and `NNEveSim` results is uncertain. Although it seems that pion-nucleon resonance production does not explain the enhanced K^+/π^+ ratio, it has been advanced [13] that pion-pion interactions may result in a similar effect. If resonance production of any form is disproven to explain the experimental data, it is most likely that the high K^+/π^+ ratio, especially relative to the K^-/π^- ratio, is indicative of extremely high baryon densities at the time of particle production. Thus, it would seem that AGS energy collisions yield nearly maximum nuclear stopping, and highly compressed and dense states of nuclear matter. What ever the resulting explanation, K^+/π^+ enhancement is a signature of exotic, and heretofore unseen, phenomena that should excite interest and promote further experimental and theoretical exploration at the frontier of nuclear physics.

Appendix A

The PiNRP Branching Ratios

The following details the branching ratios used within the `PiNRP` model for the decay of pion-nucleon resonant states. The values given represent a consistent approximation to the widely varying data given in the *Review of Particle Properties* [2]. Note as well that many of the more massive resonances have not been sufficiently studied to determine the all of their decay channels. Herein, these undetermined decays are notated with a particle type of *unknown*.

Table A–1: $\Delta(1232)$ Branching Ratios

Decay Product	*Branching Ratio*
$N\pi$	99.40%
$N\gamma$	0.60%

Table A–2: $\Delta(1620)$ Branching Ratios

Decay Product	*Branching Ratio*
$N\pi$	30.00%
$\Delta\pi$	59.97%
$N\rho$	10.00%
$N\gamma$	0.03%

Table A–3: $\Delta(1700)$ Branching Ratios

Decay Product	*Branching Ratio*
$N\pi$	15.00%
$\Delta\pi$	54.75%
$N\rho$	30.00%
$N\gamma$	0.25%

Table A–4: $\Delta(1900)$ Branching Ratios

Decay Product	*Branching Ratio*
$N\pi$	10.00%
ΣK	10.00%
Unknown	80.00%

Table A–5: $\Delta(1905)$ Branching Ratios

Decay Product	*Branching Ratio*
$N\pi$	10.00%
ΣK	3.00%
$\Delta\pi$	25.00%
$N\rho$	50.00%
Unknown	12.00%
$N\gamma$	0.05%

Table A–6: $\Delta(1910)$ Branching Ratios

Decay Product	*Branching Ratio*
$N\pi$	20.00%
ΣK	12.00%
$\Delta\pi$	6.00%
$N\rho$	7.00%
$N(1440)\pi$	55.00%

Table A–7: $\Delta(1920)$ Branching Ratios

Decay Product	*Branching Ratio*
$N\pi$	17.00%
ΣK	5.00%
Unknown	78.00%

Table A–8: $\Delta(1930)$ Branching Ratios

Decay Product	*Branching Ratio*
$N\pi$	17.00%
ΣK	10.00%
Unknown	73.00%

Table A–9: $\Delta(1950)$ Branching Ratios

Decay Product	*Branching Ratio*
$N\pi$	40.00%
ΣK	1.00%
$\Delta\pi$	30.00%
$N\rho$	10.00%
Unknown	18.85%
$N\gamma$	0.15%

Table A–10: $\Delta(2420)$ Branching Ratios

Decay Product	*Branching Ratio*
$N\pi$	10.00%
Unknown	90.00%

Table A–11: N(1440) Branching Ratios

Decay Product	*Branching Ratio*
$N\pi$	60.00%
$\Delta\pi$	15.00%
$N\rho$	15.00%
$N\pi\pi$	9.87%
$N\gamma$	0.13%

Table A–12: N(1520) Branching Ratios

Decay Product	*Branching Ratio*
$N\pi$	55.00%
$N\eta$	0.10%
$\Delta\pi$	25.00%
$N\rho$	15.00%
$N\pi\pi$	4.00%
$N\gamma$	0.90%

Table A–13: N(1535) Branching Ratios

Decay Product	*Branching Ratio*
$N\pi$	42.00%
$N\eta$	50.00%
$\Delta\pi$	2.00%
$N\rho$	2.05%
$N\pi\pi$	3.00%
$N\gamma$	0.95%

Table A–14: N(1650) Branching Ratios

Decay Product	*Branching Ratio*
$N\pi$	60.00%
$N\eta$	1.50%
ΛK	8.00%
$\Delta\pi$	5.30%
$N\rho$	15.00%
$N\pi\pi$	10.00%
$N\gamma$	0.20%

Table A–15: N(1675) Branching Ratios

Decay Product	*Branching Ratio*
$N\pi$	37.00%
$N\eta$	1.00%
ΛK	0.10%
$\Delta\pi$	54.79%
$N\rho$	5.00%
$N\pi\pi$	2.00%
$N\gamma$	0.11%

Table A–16: N(1680) Branching Ratios

Decay Product	*Branching Ratio*
$N\pi$	60.00%
$N\eta$	0.70%
$\Delta\pi$	10.00%
$N\rho$	12.00%
$N\pi\pi$	17.00%
$N\gamma$	0.30%

Table A–17: N(1700) Branching Ratios

Decay Product	*Branching Ratio*
$N\pi$	10.00%
$N\eta$	4.79%
ΛK	0.20%
$\Delta\pi$	30.00%
$N\rho$	10.00%
$N\pi\pi$	45.00%
$N\gamma$	0.01%

Table A–18: N(1710) Branching Ratios

Decay Product	*Branching Ratio*
$N\pi$	15.00%
$N\eta$	20.00%
ΛK	15.00%
ΣK	10.00%
$\Delta\pi$	15.00%
$N\rho$	12.50%
$N\pi\pi$	12.50%

Table A–19: N(1720) Branching Ratios

Decay Product	*Branching Ratio*
$N\pi$	15.00%
$N\eta$	3.50%
ΛK	5.00%
ΣK	5.00%
$\Delta\pi$	10.00%
$N\rho$	50.00%
$N\pi\pi$	11.50%

Table A–20: N(2190) Branching Ratios

Decay Product	*Branching Ratio*
$N\pi$	14.00%
$N\eta$	3.00%
ΛK	0.30%
Unknown	82.70%

Table A–21: N(2220) Branching Ratios

Decay Product	*Branching Ratio*
$N\pi$	18.00%
$N\eta$	0.50%
ΛK	0.20%
Unknown	81.30%

Table A–22: N(2250) Branching Ratios

Decay Product	*Branching Ratio*
$N\pi$	10.00%
$N\eta$	2.00%
ΛK	0.30%
Unknown	87.70%

Table A–23: N(2600) Branching Ratios

Decay Product	*Branching Ratio*
$N\pi$	5.00%
Unknown	95.00%

Appendix B

The PiNRP code

The following is the complete VAX `FORTRAN` implementation of the `PiNRP` model. The code is fairly well documented, and should serve to lead the reader through its structure. A slightly altered version of this code was used in the `NNEveSim` code as well, with modifications to determine the distance traveled in the nucleus within the `PiNRP` subroutine. Unfortunately, this routine does not at all correspond to the `FORTRAN 77` standard, relying heavily upon the DEC VAX extensions. Also included are the common block used in both `PiNRP` and `NNEveSim`, as well as the `PiNBIN` shell used to control the `PiNRP` model.

```
c
c     PISCAT common block for particle data.  See below comments for details.
c
c     File:             UDISK:[RMARQ.PISCAT]COMMON.FOR
c     By:               Ron Marquardt
c     Created:          September 22, 1988
c     Modified:         ???
c
c     Project:          PISCAT program for thesis research (pion rescattering)
c
c     Variables:  (as of 9/22/88)
c
c     PARTID (Which, What) -- INTEGER*2 array
c             Which: 1 - 250 (particle #)
c             What:  = 1, ID# of type
c                    = 2, charge
c             NOTES:  PARTID holds the integer data on produced particles.
c
c     PARTDAT (Which, What) -- REAL*4 array
c             Which: 1 - 250 (particle #)
c             What:  = 1-3, vertex location (X, Y, Z) [fm]
c                    = 4-6, 3-D momentum (X, Y, Z) [GeV/c]
c                    = 7,   energy [GeV]
c                    = 8,   total momentum [GeV/c]
c             NOTES:  PARTDAT holds the real data (positions and momenta)
c                     of produced particles.
c
c     NEWPART (Which, What) -- REAL*4 array
c             Which: 1 - 10 (new particle #)
c             What:  = 1, ID# for type --> stored as REAL, treat as INT
c                    = 2, charge       --> stored as REAL, treat as INT
c                    = 3, initial energy (total)
c                    = 4-6, initial momentum [GeV/c]
c             NOTES:  NEWPART holds the preliminary data on produced
c                     particles before they are assigned momenta.  This
c                     allows the initial cross-section stuff to be dealt
c                     with independent of the energy/momentum conservation.
c
c     PARTFLAG (Which) -- LOGICAL array
c             Which: 1 - 250 (particle #)
c             NOTES:  A logical assigned to each particle to keep track
c                     of validity.  Originally thought to be needed,
c                     although it may now be useless.
c
c     PARTPNT -- INTEGER*2 pointing to the current particle in the list.
c
c     ENERGY  -- REAL*4 giving the mean energy/u for the projectile
c
c     A_PROJ, A_TARG, Z_PROJ, Z_TARG -- INTEGER*2s giving the A and Z
c                                       for both target and projectile.
c
c     NRADIUS -- REAL*4 giving the nuclear radius of the target based
c                upon A_TARG.
c
c     NBAD -- Stores the number of unresolved decays.
c

      INTEGER*4         NBAD
      INTEGER*2         PARTID (1:250, 1:2), PARTPNT,
     1                  A_PROJ, A_TARG, Z_PROJ, Z_TARG
      REAL*4            PARTDAT (1:250, 1:8), NEWPART (1:10, 1:6), ENERGY,
     1                  NRADIUS
      LOGICAL           PARTFLAG (1:250)

      COMMON  /PARTCOM/         PARTID, PARTDAT, NEWPART, PARTFLAG,
     1                          PARTPNT, ENERGY, A_PROJ,
     2                          A_TARG, Z_PROJ, Z_TARG, NRADIUS, NBAD

c     END OF FILE
```

```
      PROGRAM PINBIN

c     PINBIN program to generate pi/N events and bin them for later
c     analysis.  Calls PIN_COLLIDE for all collision info.
c
c     File:           UDISK:[RMARQ.PISCAT.SOURCE]PINBIN.FOR
c     By:             Ron Marquardt
c     Created:        November 2, 1988
c     Modified:       ???
c
c     Project:        PISCAT program for thesis research (pion rescattering)
c
c     Variables:  (as of 11/2/88)

c     Global

      INCLUDE 'DISK$USER:[RMARQ.PISCAT.SOURCE]COMMON.FOR'

c     Local

      INTEGER*4       RND, SEED, BIN (0:19), NSTAR, DELTA, NEVENTS, I
      INTEGER*2       CHARGE, NUMPART, PARTTYPES (1:20), RES
      INTEGER*2       PITYPE, P_DIST, UNAMELEN, J
      INTEGER*2       TLEN (3), FNAMELEN, BINCODE (29), WHICH
      REAL*4          P_PI, PROB, SECNDS, P_PI_LOW, P_PI_HIGH, TEMP
      LOGICAL         PIONABS, USEED
      CHARACTER*70    UNAME, TEXT (3), FILE
      CHARACTER*10    BINNAME (0:14)

      PARAMETER M_PROTON = 938.3
      PARAMETER M_NEUTRON = 939.6
      PARAMETER M_PI = 139.57

      COMMON  /MONTE/         PROB, RND
      DATA    BINCODE         /1, 0, 0, 0, 0, 0, 2, 3, 4, 5, 6, 0, 7, 8,
     1                        0, 9, 10, 11, 12, 13, 14, 0, 0, 0, 0, 0,
     1                        0, 0, 0/

      DATA    BINNAME         /'(Bad      )', '(Gamma    )', '(Pi 0     )',
     1                         '(Pi +     )', '(Pi -     )', '(K0 Long )',
     1                         '(K+       )', '(n        )', '(p        )',
     1                         '(K0 Short)', '(Eta      )', '(Lambda   )',
     1                         '(Sigma + )', '(Sigma 0 )', '(Sigma - )'/

c     OK, this is what we do...  We ask for the following,
c
c             1) Number of events.
c             2) Types of pions (Pi+, Pi-, or both)
c             3) Target nucleus (A and Z)
c             4) Pion absorbtion (Y or N)
c             5) Random seed or user-specified
c             6) Fixed incident momentum, constant over range,
c                or arbitary momentum distribution.
c
c     Then we do our initializations and call PIN_COLLIDE for every
c     event requested.  Keep track of bad events.  Bin created
c     particles and keep track of particlo ratios.
c
c     Later... Add position and momentum space and allow for a cut
c     based upon detector performance.  Need to get position and
c     momenta of produced particles in PIN_COLLIDE, or some other
c     routine.
c

      PRINT *, 'PINBIN Pi/N resonance production simulator'
      PRINT *, ' '
      PRINT *, ' '

      CALL INPUT_TEXT ('Enter user name:  \', UNAME, UNAMELEN)
      PRINT *, ' '
      PRINT *, 'Enter up to 3 lines of notes for this simulation run'
      CALL INPUT_TEXT ('1: \', TEXT (1), TLEN (1))
      CALL INPUT_TEXT ('2: \', TEXT (2), TLEN (2))
      CALL INPUT_TEXT ('3: \', TEXT (3), TLEN (3))
      PRINT *, ' '
      CALL INPUT_TEXT ('Enter name for output file:  \', FILE, FNAMELEN)
      OPEN (UNIT=10, FILE=FILE(1:FNAMELEN), STATUS='UNKNOWN')
      I = UNAMELEN
      IF (I .NE. 0) WRITE (10, 100) UNAME (1:UNAMELEN)
      I = TLEN (1)
      IF (I .NE. 0) WRITE (10, 100) TEXT (1) (1:TLEN(1))
```

```
      I = TLEN (2)
      IF (I .NE. 0) WRITE (10, 100) TEXT (2) (1:TLEN(2))
      I = TLEN (3)
      IF (I .NE. 0) WRITE (10, 100) TEXT (3) (1:TLEN(3))
100   FORMAT (1X, '( ', A<I>, ' )')
      PRINT *, ' '
      PRINT *, ' '

      CALL INPUT_INT ('Enter number of Pi/N events to simulate: \',
     1                NEVENTS, 1000)
      WRITE (10, 110) NEVENTS
110   FORMAT (1X, '( ', I8, ' Pi/N events. )')
      PRINT *, ' '

      PRINT *, 'Enter incident pion type (1-3):  '
      PRINT *, '         1) Pi+ only'
      PRINT *, '         2) Pi- only'
      PRINT *, '         3) Pi0 only'
115   PRINT *, '         4) Both Pi+ and Pi-'
105   CALL INPUT_INT ('Pion type (1-3):  \', PITYPE, 3)
      IF (PITYPE .EQ. 1) THEN
              WRITE (10, 120) 'Pi+ only         '
      ELSE IF (PITYPE .EQ. 2) THEN
              WRITE (10, 120) 'Pi- only         '
      ELSE IF (PITYPE .EQ. 3) THEN
              WRITE (10, 120) 'Pi0 only         '
      ELSE IF (PITYPE .EQ. 4) THEN
              WRITE (10, 120) 'All Pi types     '
      ELSE
              GOTO 105
      ENDIF
120     FORMAT (1X, '( ', A16, ' )')
      PRINT *, ' '

      PRINT *, 'Enter target nucleus data:  '
      CALL INPUT_INT ('Target mass number:  \', A_TARG, 1)
      CALL INPUT_INT ('Target charge:       \', Z_TARG, 1)
      WRITE (10, 130) A_TARG, Z_TARG
130   FORMAT (1X, '( Target A:  ', I4, ', Target Z:  ', I4, ' )')
      PRINT *, ' '

      CALL INPUT_YN ('Include nuclear pion absorption?  \', PIONABS, 'Y')
      IF (PIONABS) THEN
              WRITE (10, 140) 'No Pion absorbtion.     '
      ELSE
              WRITE (10, 140) 'Without pion absorbtion.'
      ENDIF
140   FORMAT (1X, '( ', A24, ' )')
      PRINT *, ' '

      CALL INPUT_YN ('User specified random number seed? \', USEED, 'N')
      IF (USEED) THEN
              CALL INPUT_INT ('Random number seed:  \', SEED, 7777777)
              IF ((FLOAT(SEED)/2.0) .EQ. NINT(FLOAT(SEED)/2.0))
     1        SEED = SEED + 1
              RND = SEED
      ELSE
              RND = NINT (SECNDS (0.0) * 2000.0) + 1
      ENDIF
      WRITE (10, 150) RND
150   FORMAT (1X, '( Seed = ', I12, ' )')
155   PROB = RAN(RND)
      PRINT *, ' '

      PRINT *, 'Enter pion momentum distribution type (1-3):  '
      PRINT *, '         1) Constant incident momentum'
      PRINT *, '         2) Equal probability over user-specified range'
      PRINT *, '         3) Parameterized experimental momentum distribution'
      CALL INPUT_INT ('Pion momentum distribution (1-3):  \', P_DIST, 3)
      PRINT *, ' '
      IF (P_DIST .EQ. 1) THEN
              CALL INPUT_REAL ('Enter incident pion momentum (GeV):  \',
     1                         P_PI_LOW, 1.0)
              WRITE (10, 160) P_PI_LOW
160   FORMAT (1X, '( Constant Pion momentum = ', F12.7, ' GeV/c )')
      ELSE IF (P_DIST .EQ. 2) THEN
              CALL INPUT_REAL ('Enter low pion momentum (GeV/c):  \',
     1                         P_PI_LOW, 0.2)
              CALL INPUT_REAL ('Enter high pion momentum (GeV/c):  \',
     1                         P_PI_HIGH, 1.7)
              WRITE (10, 170) P_PI_LOW, P_PI_HIGH
```

```
 170  FORMAT (1X, '( Constant Probability over momentum interval from ',
     1        F12.7, ' to ', F12.7, ' Gev/c )')
      ELSE IF (P_DIST .EQ. 3) THEN
              WRITE (10, 180)
 180  FORMAT (1X, '( Parameterized experimental momentum distribution )')
      ELSE
              GOTO 155
      ENDIF

C     Now we can do it...

      PRINT *, 'Calculating ', NEVENTS, ' Pi/N collisions.'

      DELTA = 0
      NSTAR = 0
      NBAD = 0
      DO J = 1, 19
              BIN (J) = 0
      ENDDO

      DO I = 1, NEVENTS

              IF (PITYPE .EQ. 4) THEN
                      PROB = RAN(RND)
                      IF (PROB .LE. 0.50000) THEN
                              CHARGE = 1
                      ELSE
                              CHARGE = -1
                      ENDIF
              ELSE IF (PITYPE .EQ. 1) THEN
                      CHARGE = 1
              ELSE IF (PITYPE .EQ. 2) THEN
                      CHARGE = -1
              ELSE IF (PITYPE .EQ. 3) THEN
                      CHARGE = 0
              ELSE
                      PRINT *, 'ERROR -- Invalid PITYPE.'
              ENDIF

              IF (P_DIST .EQ. 1) THEN
                      P_PI = P_PI_LOW
              ELSE IF (P_DIST .EQ. 2) THEN
                      PROB = RAN(RND)
                      P_PI = (P_PI_HIGH - P_PI_LOW)*PROB + P_PI_LOW
C             ELSE IF (P_DIST .EQ. 3) THEN
C                     Add momentum distribution...
              ELSE
                      PRINT *, 'ERROR -- Invalid P_DIST.'
              ENDIF

              DO J = 1, 20
                      PARTTYPES (J) = 0
              ENDDO
              NUMPART = 0

              TEMP = 100000000.00
              CALL PIN_COLLIDE (CHARGE, P_PI, TEMP, RES, NUMPART, PARTTYPES)

              DO J = 1, NUMPART
                      WHICH = BINCODE(PARTTYPES(J))
                      BIN (WHICH) = BIN (WHICH) + 1
              ENDDO

              IF (RES .GT. 90) THEN
                      DELTA = DELTA + 1
              ELSE
                      NSTAR = NSTAR + 1
              ENDIF

      ENDDO

      BIN (0) = BIN (0) + NBAD                  ! could be changed = Nbad

      DO J = 0, 14
              WRITE (10, 200) J, BIN (J), BINNAME (J)
 200  FORMAT (1X, I2, 3X, I20, 15X, A10)
      ENDDO

      PRINT *, 'Completed and exiting...'
      CLOSE (10)
```

END

```
      SUBROUTINE PIN_COLLIDE (CHARGE, P_PI, RES, NUMPART, PARTTYPES)

C
C     A routine designed to yield particle distributions from pi-nucleon
C     collisions.  At the moment, all that should be passed is an incident
C     momentum and charge (for a pion).  Returned are the Geant particle
C     code numbers for the produced particles, as well as the ID given
C     the resonance formed, the number of particles created, and the
C     number of bad decays that have occured (incremented by one within
C     this subroutine).
C
C     File:             UDI3K:[RMARQ.PISCAT.SOURCE]PIN_COLLIDE.FOR
C     By:               Ron Marquardt
C     Created:          October 6, 1988
C     Modified:         November 1, 1988
C
C     Project:          PISCAT program for thesis research (pion rescattering)
C
C     Variables:  (as of 11/1/88)

C     Global

      INCLUDE 'DISK$USER:[RMARQ.PISCAT.SOURCE]COMMON.FOR'

C     Local

      INTEGER*4         RND
      INTEGER*2         CHARGE, NUMPART, PARTTYPES (1:20),RES,RES_LEVEL
      INTEGER*2         I, J, INDEX
      REAL*4            P_PI, CS (1:40), CS_TOTAL, PROB, E_CM, SECNDS
      LOGICAL           ISAPROTON
      PARAMETER M_PROTON = 938.3
      PARAMETER M_NEUTRON = 939.6
      PARAMETER M_PI = 139.57

      COMMON  /MONTE/           PROB, RND

C     RES: if >90, then a delta of charge RES-100, else
C          a N* of charge RES.

C     Code

C     The random number generator has previously been seeded, either
C     randomly, or by a user-given constant.  The associated variables
C     are to be found in the common block MONTE -- RND is the seed and
C     PROB is the random number (0--1).

C     We assume that before this routine, a Monte-Carlo of mean free path
C     yielded a collision with a nucleon.  Knowing that a collision has
C     occured, we figure out the type of nucleon struck and the relative
C     chances for the various resonances to be formed.  Thus, we first
C     determine whether a neutron or proton was hit, based upon the A
C     and Z of the target nucleus.

      IF (FLOAT(A_TARG)*RAN(RND) .GT. Z_TARG) THEN
              ISAPROTON = .FALSE.
              E_CM = SQRT(M_PROTON**2 + M_PI**2 + 2*SQRT(M_PI**2 +
     1                   (1000.0*P_PI)**2)*M_PROTON)
      ELSE
              ISAPROTON = .TRUE.
              E_CM = SQRT(M_NEUTRON**2 + M_PI**2 + 2*SQRT(M_PI**2 +
     1                   (1000.0*P_PI)**2)*M_NEUTRON)
      ENDIF

      IF (ISAPROTON) THEN                      ! Switch on status of target

              IF (CHARGE .EQ. 1) THEN          ! PI+

C     [Pi+/P --> Delta++]

                      RES = 100 + 2            ! Delta with +2 charge

                      CS_TOTAL = 0.0
                      DO I = 1, 10
                              CS (I) = CROSS_SECTION (1, I, E_CM)
                              CS_TOTAL = CS_TOTAL + CS (I)
                      ENDDO

                      PROB = CS_TOTAL*RAN(RND)
                      I = 1
                      CS_TOTAL = CS (1)
```

```
                    DO WHILE (PROB .GT. CS_TOTAL)
                            I = I + 1
                            CS_TOTAL = CS_TOTAL + CS (I)
                    ENDDO

                    RES_LEVEL = I

            ELSE IF (CHARGE .EQ. -1) THEN    ! PI-

C      [Pi-/P --> N0 or Delta0]

                    CS_TOTAL = 0.0
                    DO I = 1, 10
                            CS (I) = CROSS_SECTION (1, I, E_CM)/3.0
                            CS_TOTAL = CS_TOTAL + CS (I)
                    ENDDO

                    DO I = 1, 13
                            CS (I+10) = CROSS_SECTION (2, I, E_CM)/1.5
                            CS_TOTAL = CS_TOTAL + CS (I)
                    ENDDO

                    PROB = CS_TOTAL*RAN(RND)
                    I = 1
                    CS_TOTAL = CS (1)
                    DO WHILE (PROB .GT. CS_TOTAL)
                            I = I + 1
                            CS_TOTAL = CS_TOTAL + CS (I)
                    ENDDO

                    IF (I .LE. 10) THEN
                            RES = 100 + 0             ! DELTA-0
                            RES_LEVEL = I
                    ELSE
                            RES = 0                   ! N-0
                            RES_LEVEL = I - 10
                    ENDIF

            ELSE                                     ! PI0

C      [Pi0/P --> N+ or Delta+]

                    CS_TOTAL = 0.0
                    DO I = 1, 10
                            CS (I) = CROSS_SECTION (1, I, E_CM)/1.5
                            CS_TOTAL = CS_TOTAL + CS (I)
                    ENDDO

                    DO I = 1, 13
                            CS (I+23) = CROSS_SECTION (2, I, E_CM)/1.5
                            CS_TOTAL = CS_TOTAL + CS (I)
                    ENDDO

                    PROB = CS_TOTAL*RAN(RND)
                    I = 1
                    CS_TOTAL = CS (1)
                    DO WHILE (PROB .GT. CS_TOTAL)
                            I = I + 1
                            CS_TOTAL = CS_TOTAL + CS (I)
                    ENDDO

                    IF (I .LE. 10) THEN
                            RES = 100 + 1             ! DELTA+
                            RES_LEVEL = I
                    ELSE
                            RES = 1                   ! N+
                            RES_LEVEL = I - 10
                    ENDIF

            ENDIF

      ELSE

            IF (CHARGE .EQ. 1) THEN          ! PI+

C      [Pi+/n --> N+ or Delta+]

                    CS_TOTAL = 0.0
                    DO I = 1, 10
                            CS (I) = CROSS_SECTION (1, I, E_CM)/3.0
                            CS_TOTAL = CS_TOTAL + CS (I)
```

```
                  ENDDO

                  DO I = 1, 13
                          CS (I+23) = CROSS_SECTION (2, I, E_CM)/1.5
                          CS_TOTAL = CS_TOTAL + CS (I)
                  ENDDO

                  PROB = CS_TOTAL*RAN(RND)
                  I = 1
                  CS_TOTAL = CS (1)
                  DO WHILE (PROB .GT. CS_TOTAL)
                          I = I + 1
                          CS_TOTAL = CS_TOTAL + CS (I)
                  ENDDO

                  IF (I .LE. 10) THEN
                          RES = 100 + 1              ! DELTA+
                          RES_LEVEL = I
                  ELSE
                          RES = 1                    ! N+
                          RES_LEVEL = I - 10
                  ENDIF

          ELSE IF (CHARGE .EQ. -1) THEN    ! PI-

C     [Pi-/n --> Delta-]

                  RES = 100 - 1              ! Delta with -1 charge

                  CS_TOTAL = 0.0
                  DO I = 1, 10
                          CS (I) = CROSS_SECTION (1, I, E_CM)
                          CS_TOTAL = CS_TOTAL + CS (I)
                  ENDDO

                  PROB = CS_TOTAL*RAN(RND)
                  I = 1
                  CS_TOTAL = CS (1)
                  DO WHILE (PROB .GT. CS_TOTAL)
                          I = I + 1
                          CS_TOTAL = CS_TOTAL + CS (I)
                  ENDDO

                  RES_LEVEL = I

          ELSE                               ! PI0

C     [Pi0/n --> N0 or Delta0]

                  CS_TOTAL = 0.0
                  DO I = 1, 10
                          CS (I) = CROSS_SECTION (1, I, E_CM)/1.5
                          CS_TOTAL = CS_TOTAL + CS (I)
                  ENDDO

                  DO I = 1, 13
                          CS (I+10) = CROSS_SECTION (2, I, E_CM)/3.0
                          CS_TOTAL = CS_TOTAL + CS (I)
                  ENDDO

                  PROB = CS_TOTAL*RAN(RND)
                  I = 1
                  CS_TOTAL = CS (1)
                  DO WHILE (PROB .GT. CS_TOTAL)
                          I = I + 1
                          CS_TOTAL = CS_TOTAL + CS (I)
                  ENDDO

                  IF (I .LE. 10) THEN
                          RES = 100 + 0              ! DELTA-0
                          RES_LEVEL = I
                  ELSE
                          RES = 0                    ! N-0
                          RES_LEVEL = I - 10
                  ENDIF

          ENDIF

      ENDIF

C     We know the resonance and the charge.  Now we must return the
```

```
c     decay products.

c     We generate a random number once.  All of the conditionals below
c     will require it.  We do it now to save space.  Note that each
c     conditional thus is written to >>expect<< a valid new value for
c     PROB.

      PROB = RAN(RND)
      NUMPART = 0
      INDEX = 1

c     The following code must be reentrant, as several of the particles
c     decay into other resonances.  To this end, index points to the
c     most current slot avaliable in the PARTTYPES array.

c     We first make a conditional on RES.  If RES .GT. 90, then the
c     resonance is a delta, with a charge of RES - 100.  With this
c     knowledge, we decay based upon the value of RES_LEVEL.

      IF (RES .GT. 90) THEN                        ! A delta

c     Calculate the charge.

              CHARGE = RES - 100

              IF (RES_LEVEL .EQ. 1) THEN

c     --- Delta (1232) ---
c     Probabilities:  N/Pi            99.40%
c                     N/Gamma          0.60%

 100                  IF (PROB .LE. 0.994) THEN

                                      NUMPART = NUMPART + 2
                              PROB = RAN(RND)
                              CALL NPI (PARTTYPES, CHARGE, INDEX)
                              INDEX = NUMPART + 1

                      ELSE

                              NUMPART = NUMPART + 2
                              CALL NGAMMA (PARTTYPES, CHARGE, INDEX)
                              INDEX = NUMPART + 1

                      ENDIF

              ELSE IF (RES_LEVEL .EQ. 2) THEN

c     --- Delta (1620) ---
c     Probabilities:  N/Pi            30.00%
c                     Delta/Pi        59.97%
c                     N/Rho           10.00%
c                     N/Gamma          0.03%

 200                  IF (PROB .LE. 0.0003) THEN

                              NUMPART = NUMPART + 2
                              CALL NGAMMA (PARTTYPES, CHARGE, INDEX)
                              INDEX = NUMPART + 1

                      ELSE IF (PROB .LE. 0.3003) THEN

                              NUMPART = NUMPART + 2
                              PROB = RAN(RND)
                              CALL NPI (PARTTYPES, CHARGE, INDEX)
                              INDEX = NUMPART + 1

                      ELSE IF (PROB .LE. 0.4003) THEN

                              NUMPART = NUMPART + 3
                              CALL NRHO (PARTTYPES, CHARGE, INDEX)
                              INDEX = NUMPART + 1

                      ELSE

                              NUMPART = NUMPART + 1
                              PROB = RAN(RND)
                              CALL DELTAPI (PARTTYPES, CHARGE, INDEX)
                              PROB = RAN(RND)
                              INDEX = NUMPART + 1
                              GOTO 100
```

```
                    ENDIF

              ELSE IF (RES_LEVEL .EQ. 3) THEN

C     --- Delta (1700) ---
C     Probabilities:  N/Pi            15.00%
C                     Delta/Pi        54.75%
C                     N/Rho           30.00%
C                     N/Gamma          0.25%

 300                  IF (PROB .LE. 0.0025) THEN

                              NUMPART = NUMPART + 2
                              CALL NGAMMA (PARTTYPES, CHARGE, INDEX)
                              INDEX = NUMPART + 1

                      ELSE IF (PROB .LE. 0.1525) THEN

                              NUMPART = NUMPART + 2
                              PROB = RAN(RND)
                              CALL NPI (PARTTYPES, CHARGE, INDEX)
                              INDEX = NUMPART + 1

                      ELSE IF (PROB .LE. 0.4525) THEN

                              NUMPART = NUMPART + 3
                              CALL NRHO (PARTTYPES, CHARGE, INDEX)
                              INDEX = NUMPART + 1

                      ELSE

                              NUMPART = NUMPART + 1
                              PROB = RAN(RND)
                              CALL DELTAPI (PARTTYPES, CHARGE, INDEX)
                              PROB = RAN(RND)
                              INDEX = NUMPART + 1
                              GOTO 100

                      ENDIF

              ELSE IF (RES_LEVEL .EQ. 4) THEN

C     --- Delta (1900) ---
C     Probabilities:  N/Pi            10.00%
C                     Sigma/K         10.00%
C                     NOTE:  80% of decay unaccounted for.

400                   IF (PROB .LE. 0.10) THEN

                              NUMPART = NUMPART + 2
                              PROB = RAN(RND)
                              CALL NPI (PARTTYPES, CHARGE, INDEX)
                              INDEX = NUMPART + 1

                      ELSE IF (PROB .LE. 0.20) THEN

                              NUMPART = NUMPART + 2
                              PROB = RAN(RND)
                              CALL SIGMAK (PARTTYPES, CHARGE, INDEX)
                              INDEX = NUMPART + 1

                      ELSE

                              NBAD = NBAD + 1

                      ENDIF

              ELSE IF (RES_LEVEL .EQ. 5) THEN

C     --- Delta (1905) ---
C     Probabilities:  N/Pi            10.00%
C                     Sigma/K          3.00%
C                     Delta/Pi        25.00%
C                     N/Rho           50.00%
C                     N/Gamma          0.05%
C                     NOTE:  12% of decay unaccounted for.

                      IF (PROB .LE. 0.0005) THEN

                              NUMPART = NUMPART + 2
```

```
                        CALL NGAMMA (PARTTYPES, CHARGE, INDEX)
                        INDEX = NUMPART + 1

                ELSE IF (PROB .LE. 0.5005) THEN

                        NUMPART = NUMPART + 3
                        CALL NRHO (PARTTYPES, CHARGE, INDEX)
                        INDEX = NUMPART + 1

                ELSE IF (PROB .LE. 0.6005) THEN

                        NUMPART = NUMPART + 2
                        PROB = RAN(RND)
                        CALL NPI (PARTTYPES, CHARGE, INDEX)
                        INDEX = NUMPART + 1

                ELSE IF (PROB .LE. 0.8505) THEN

                        NUMPART = NUMPART + 1
                        PROB = RAN(RND)
                        CALL DELTAPI (PARTTYPES, CHARGE, INDEX)
                        PROB = RAN(RND)
                        INDEX = NUMPART + 1
                        IF (PROB .LE. 0.333333) THEN
                                PROB = RAN(RND)
                                GOTO 100
                        ELSE IF (PROB .LE. 0.666667) THEN
                                PROB = RAN(RND)
                                GOTO 200
                        ELSE
                                PROB = RAN(RND)
                                GOTO 300
                        ENDIF

                ELSE IF (PROB .LE. 0.8805) THEN

                        NUMPART = NUMPART + 2
                        PROB = RAN(RND)
                        CALL SIGMAK (PARTTYPES, CHARGE, INDEX)
                        INDEX = NUMPART + 1

                ELSE

                        NBAD = NBAD + 1

                ENDIF

            ELSE IF (RES_LEVEL .EQ. 6) THEN

C       --- Delta (1910) ---
C       Probabilities:  N/Pi            20.00%
C                       Sigma/K         12.00%
C                       Delta/Pi         6.00%
C                       N/Rho            7.00%
C                       N(1440)/Pi      55.00%
C                       NOTE:  Resonance decays into an N*.

 750            IF (PROB .LE. 0.20) THEN

                        NUMPART = NUMPART + 2
                        PROB = RAN(RND)
                        CALL NPI (PARTTYPES, CHARGE, INDEX)
                        INDEX = NUMPART + 1

                ELSE IF (PROB .LE. 0.32) THEN

                        NUMPART = NUMPART + 2
                        PROB = RAN(RND)
                        CALL SIGMAK (PARTTYPES, CHARGE, INDEX)
                        INDEX = NUMPART + 1

                ELSE IF (PROB .LE. 0.38) THEN

                        NUMPART = NUMPART + 1
                        PROB = RAN(RND)
                        CALL DELTAPI (PARTTYPES, CHARGE, INDEX)
                        PROB = RAN(RND)
                        INDEX = NUMPART + 1
                        IF (PROB .LE. 0.333333) THEN
                                PROB = RAN(RND)
                                GOTO 100
```

```
                        ELSE IF (PROB .LE. 0.666667) THEN
                                PROB = RAN(RND)
                                GOTO 200
                        ELSE
                                PROB = RAN(RND)
                                GOTO 300
                        ENDIF

                ELSE IF (PROB .LE. 0.45) THEN

                        NUMPART = NUMPART + 3
                        CALL NRHO (PARTTYPES, CHARGE, INDEX)
                        INDEX = NUMPART + 1

                ELSE

C       Special case:  only one to decay into N* type resonance.

                        NUMPART = NUMPART + 1
                        PROB = RAN(RND)
                        IF (CHARGE .EQ. 2) THEN
                                PARTTYPES (INDEX) = 8
                                CHARGE = 1
                        ELSE IF (CHARGE .EQ. 1) THEN
                                IF (PROB .LE. 0.5000) THEN
                                        PARTTYPES (INDEX) =
     1                                          7
                                        CHARGE = 1
                                ELSE
                                        PARTTYPES (INDEX) =
     1                                          8
                                        CHARGE = 0
                                ENDIF
                        ELSE IF (CHARGE .EQ. 0) THEN
                                IF (PROB .LE. 0.5000) THEN
                                        PARTTYPES (INDEX) =
     1                                          7
                                        CHARGE = 0
                                ELSE
                                        PARTTYPES (INDEX) =
     1                                          9
                                        CHARGE = 1
                                ENDIF
                        ELSE IF (CHARGE .EQ. -1) THEN
                                PARTTYPES (INDEX) = 9
                                CHARGE = 0
                        ELSE
                                PRINT *, 'Charge error -- N*'
                        ENDIF
                        PROB = RAN(RND)
                        INDEX = NUMPART + 1
                        GOTO 500

                ENDIF

            ELSE IF (RES_LEVEL .EQ. 7) THEN

C       --- Delta (1920) ---
C       Probabilities:  N/Pi            17.00%
C                       Sigma/K          5.00%
C                       NOTE:  78% of decay unaccounted for.

                IF (PROB .LE. 0.17) THEN

                        NUMPART = NUMPART + 2
                        PROB = RAN(RND)
                        CALL NPI (PARTTYPES, CHARGE, INDEX)
                        INDEX = NUMPART + 1

                ELSE IF (PROB .LE. 0.22) THEN

                        NUMPART = NUMPART + 2
                        PROB = RAN(RND)
                        CALL SIGMAK (PARTTYPES, CHARGE, INDEX)
                        INDEX = NUMPART + 1

                ELSE

                        NBAD = NBAD + 1

                ENDIF
```

```
            ELSE IF (RES_LEVEL .EQ. 8) THEN

C     --- Delta (1930) ---
C     Probabilities:  N/Pi          17.00%
C                     Sigma/K       10.00%
C                     N/Pi-Pi       Not Seen
C                     NOTE:  73% of decay unaccounted for.

                      IF (PROB .LE. 0.17) THEN

                              NUMPART = NUMPART + 2
                              PROB = RAN(RND)
                              CALL NPI (PARTTYPES, CHARGE, INDEX)
                              INDEX = NUMPART + 1

                      ELSE IF (PROB .LE. 0.27) THEN

                              NUMPART = NUMPART + 2
                              PROB = RAN(RND)
                              CALL SIGMAK (PARTTYPES, CHARGE, INDEX)
                              INDEX = NUMPART + 1

                      ELSE

                              NBAD = NBAD + 1

                      ENDIF

            ELSE IF (RES_LEVEL .EQ. 9) THEN

C     --- Delta (1950) ---
C     Probabilities:  N/Pi          40.00%
C                     Sigma/K        1.00%
C                     Delta/Pi      30.00%
C                     N/Rho         10.00%
C                     N/Gamma        0.15%
C                     NOTE:  18.85% of decay unaccounted for.

                      IF (PROB .LE. 0.0015) THEN

                              NUMPART = NUMPART + 2
                              CALL NGAMMA (PARTTYPES, CHARGE, INDEX)
                              INDEX = NUMPART + 1

                      ELSE IF (PROB .LE. 0.1015) THEN

                              NUMPART = NUMPART + 3
                              CALL NRHO (PARTTYPES, CHARGE, INDEX)
                              INDEX = NUMPART + 1

                      ELSE IF (PROB .LE. 0.4015) THEN

                              NUMPART = NUMPART + 1
                              PROB = RAN(RND)
                              CALL DELTAPI (PARTTYPES, CHARGE, INDEX)
                              PROB = RAN(RND)
                              INDEX = NUMPART + 1
                              IF (PROB .LE. 0.333333) THEN
                                      PROB = RAN(RND)
                                      GOTO 100
                              ELSE IF (PROB .LE. 0.666667) THEN
                                      PROB = RAN(RND)
                                      GOTO 200
                              ELSE
                                      PROB = RAN(RND)
                                      GOTO 300
                              ENDIF

                      ELSE IF (PROB .LE. 0.8015) THEN

                              NUMPART = NUMPART + 2
                              PROB = RAN(RND)
                              CALL NPI (PARTTYPES, CHARGE, INDEX)
                              INDEX = NUMPART + 1

                      ELSE IF (PROB .LE. 0.8315) THEN

                              NUMPART = NUMPART + 2
                              PROB = RAN(RND)
                              CALL SIGMAK (PARTTYPES, CHARGE, INDEX)
```

```
                          INDEX = NUMPART + 1

                    ELSE

                          NBAD = NBAD + 1

                    ENDIF

              ELSE IF (RES_LEVEL .EQ. 10) THEN

C     --- Delta (2420) ---
C     Probabilities:  N/Pi            10.00%
C                     NOTE:  90% of decay unaccounted for.

                    IF (PROB .LE. 0.10) THEN

                          NUMPART = NUMPART + 2
                          PROB = RAN(RND)
                          CALL NPI (PARTTYPES, CHARGE, INDEX)
                          INDEX = NUMPART + 1

                    ELSE

                          NBAD = NBAD + 1

                    ENDIF

              ENDIF

C     The alternate portion of the conditional on RES.  The resonance
C     must be the N*, with total charge = RES.

      ELSE                              ! N* resonances

C     Set the total charge.

              CHARGE = RES

              IF (RES_LEVEL .EQ. 1 ) THEN

C     --- N* (1440) ---
C     Probabilities:  N/Pi            60.00%
C                     Delta/Pi        15.00%
C                     N/Rho           15.00%
C                     N/Pi-Pi          9.87%
C                     N/Gamma          0.13%

 500                IF (PROB .LE. 0.60) THEN

                          NUMPART = NUMPART + 2
                          PROB = RAN(RND)
                          CALL NPI (PARTTYPES, CHARGE, INDEX)
                          INDEX = NUMPART + 1

                    ELSE IF (PROB .LE. 0.75) THEN

                          NUMPART = NUMPART + 1
                          PROB = RAN(RND)
                          CALL DELTAPI (PARTTYPES, CHARGE, INDEX)
                          PROB = RAN(RND)
                          INDEX = NUMPART + 1
                          IF (PROB .LE. 0.50) THEN
                                PROB = RAN(RND)
                                GOTO 100
                          ELSE
                                PROB = RAN(RND)
                                GOTO 200
                          ENDIF

                    ELSE IF (PROB .LE. 0.90) THEN

                          NUMPART = NUMPART + 3
                          CALL NRHO (PARTTYPES, CHARGE, INDEX)
                          INDEX = NUMPART + 1

                    ELSE IF (PROB .LE. 0.9013) THEN

                          NUMPART = NUMPART + 2
                          CALL NGAMMA (PARTTYPES, CHARGE, INDEX)
                          INDEX = NUMPART + 1
```

```
                        ELSE

                                NUMPART = NUMPART + 3
                                PROB = RAN(RND)
                                CALL NPIPI (PARTTYPES, CHARGE, INDEX)
                                INDEX = NUMPART + 1

                        ENDIF

                ELSE IF (RES_LEVEL .EQ. 2) THEN

C       --- N* (1520) ---
C       Probabilities:  N/Pi            55.00%
C                       N/Eta            0.10%
C                       Delta/Pi        25.00%
C                       N/Rho           15.00%
C                       N/Pi-Pi          4.00%
C                       N/Gamma          0.90%

                        IF (PROB .LE. 0.55) THEN

                                NUMPART = NUMPART + 2
                                PROB = RAN(RND)
                                CALL NPI (PARTTYPES, CHARGE, INDEX)
                                INDEX = NUMPART + 1

                        ELSE IF (PROB .LE. 0.5510) THEN

                                NUMPART = NUMPART + 2
                                CALL NETA (PARTTYPES, CHARGE, INDEX)
                                INDEX = NUMPART + 1

                        ELSE IF (PROB .LE. 0.5910) THEN

                                NUMPART = NUMPART + 3
                                PROB = RAN(RND)
                                CALL NPIPI (PARTTYPES, CHARGE, INDEX)
                                INDEX = NUMPART + 1

                        ELSE IF (PROB .LE. 0.8410) THEN

                                NUMPART = NUMPART + 1
                                PROB = RAN(RND)
                                CALL DELTAPI (PARTTYPES, CHARGE, INDEX)
                                PROB = RAN(RND)
                                INDEX = NUMPART + 1
                                IF (PROB .LE. 0.333333) THEN
                                        PROB = RAN(RND)
                                        GOTO 100
                                ELSE IF (PROB .LE. 0.666667) THEN
                                        PROB = RAN(RND)
                                        GOTO 200
                                ELSE
                                        PROB = RAN(RND)
                                        GOTO 300
                                ENDIF

                        ELSE IF (PROB .LE. 0.9910) THEN

                                NUMPART = NUMPART + 3
                                CALL NRHO (PARTTYPES, CHARGE, INDEX)
                                INDEX = NUMPART + 1

                        ELSE

                                NUMPART = NUMPART + 2
                                PROB = RAN(RND)
                                CALL SIGMAK (PARTTYPES, CHARGE, INDEX)
                                INDEX = NUMPART + 1

                        ENDIF

                ELSE IF (RES_LEVEL .EQ. 3) THEN

C       --- N* (1535) ---
C       Probabilities:  N/Pi            42.00%
C                       N/Eta           50.00%
C                       Delta/Pi         2.00%
C                       N/Rho            2.05%
C                       N/Pi-Pi          3.00%
C                       N/Gamma          0.95%
```

```
                    IF (PROB .LE. 0.42) THEN

                            NUMPART = NUMPART + 2
                            PROB = RAN(RND)
                            CALL NPI (PARTTYPES, CHARGE, INDEX)
                            INDEX = NUMPART + 1

                    ELSE IF (PROB .LE. 0.92) THEN

                            NUMPART = NUMPART + 2
                            CALL NETA (PARTTYPES, CHARGE, INDEX)
                            INDEX = NUMPART + 1

                    ELSE IF (PROB .LE. 0.94) THEN

                            NUMPART = NUMPART + 1
                            PROB = RAN(RND)
                            CALL DELTAPI (PARTTYPES, CAHRGE, INDEX)
                            PROB = RAN(RND)
                            INDEX = NUMPART + 1
                            IF (PROB .LE. 0.333333) THEN
                                    PROB = RAN(RND)
                                    GOTO 100
                            ELSE IF (PROB .LE. 0.666667) THEN
                                    PROB = RAN(RND)
                                    GOTO 200
                            ELSE
                                    PROB = RAN(RND)
                                    GOTO 300
                            ENDIF

                    ELSE IF (PROB .LE. 0.95) THEN

                            NUMPART = NUMPART + 3
                            PROB = RAN(RND)
                            CALL NPIPI (PARTTYPES, CHARGE, INDEX)
                            INDEX = NUMPART + 1

                    ELSE IF (PROB .LE. 0.9595) THEN

                            NUMPART = NUMPART + 2
                            CALL NGAMMA (PARTTYPES, CHARGE, INDEX)
                            INDEX = NUMPART + 1

                    ELSE

                            NUMPART = NUMPART + 3
                            CALL NRHO (PARTTYPES, CHARGE, INDEX)
                            INDEX = NUMPART + 1

                    ENDIF

            ELSE IF (RES_LEVEL .EQ. 4) THEN

C     --- N* (1650) ---
C     Probabilities:  N/Pi              60.00%
C                     N/Eta              1.50%
C                     Lambda/K           8.00%
C                     Delta/Pi           5.30%
C                     N/Rho             15.00%
C                     N/Pi-Pi           10.00%
C                     N/Gamma            0.20%

                    IF (PROB .LE. 0.60) THEN

                            NUMPART = NUMPART + 2
                            PROB = RAN(RND)
                            CALL NPI (PARTTYPES, CHARGE, INDEX)
                            INDEX = NUMPART + 1

                    ELSE IF (PROB .LE. 0.70) THEN

                            NUMPART = NUMPART + 3
                            PROB = RAN(RND)
                            CALL NPIPI (PARTTYPES, CHARGE, INDEX)
                            INDEX = NUMPART + 1

                    ELSE IF (PROB .LE. 0.78) THEN

                            NUMPART = NUMPART + 2
```

```
                        PROB = RAN(RND)
                        CALL LAMBDAK (PARTTYPES, CHARGE, INDEX)
                        INDEX = NUMPART + 1

                ELSE IF (PROB .LE. 0.93) THEN

                        NUMPART = NUMPART + 3
                        CALL NRHO (PARTTYPES, CHARGE, INDEX)
                        INDEX = NUMPART + 1

                ELSE IF (PROB .LE. 0.945) THEN

                        NUMPART = NUMPART + 2
                        CALL NETA (PARTTYPES, CHARGE, INDEX)
                        INDEX = NUMPART + 1

                ELSE IF (PROB .LE. 0.947) THEN

                        NUMPART = NUMPART + 2
                        CALL NGAMMA (PARTTYPES, CHARGE, INDEX)
                        INDEX = NUMPART + 1

                ELSE

                        NUMPART = NUMPART + 1
                        PROB = RAN(RND)
                        CALL DELTAPI (PARTTYPES, CHARGE, INDEX)
                        PROB = RAN(RND)
                        INDEX = NUMPART + 1
                        IF (PROB .LE. 0.333333) THEN
                                PROB = RAN(RND)
                                GOTO 100
                        ELSE IF (PROB .LE. 0.666667) THEN
                                PROB = RAN(RND)
                                GOTO 200
                        ELSE
                                PROB = RAN(RND)
                                GOTO 300
                        ENDIF

                ENDIF

        ELSE IF (RES_LEVEL .EQ. 5) THEN

C     --- N* (1675) ---
C     Probabilities:  N/Pi            37.00%
C                     N/Eta            1.00%
C                     Lambda/K         0.10%
C                     Delta/Pi        54.79%
C                     N/Rho            5.00%
C                     N/Pi-Pi          2.00%
C                     N/Gamma          0.11%

                IF (PROB .LE. 0.37) THEN

                        NUMPART = NUMPART + 2
                        PROB = RAN(RND)
                        CALL NPI (PARTTYPES, CHARGE, INDEX)
                        INDEX = NUMPART + 1

                ELSE IF (PROB .LE. 0.42) THEN

                        NUMPART = NUMPART + 3
                        CALL NRHO (PARTTYPES, CHARGE, INDEX)
                        INDEX = NUMPART + 1

                ELSE IF (PROB .LE. 0.44) THEN

                        NUMPART = NUMPART + 3
                        PROB = RAN(RND)
                        CALL NPIPI (PARTTYPES, CHARGE, INDEX)
                        INDEX = NUMPART + 1

                ELSE IF (PROB .LE. 0.4411) THEN

                        NUMPART = NUMPART + 2
                        CALL NGAMMA (PARTTYPES, CHARGE, INDEX)
                        INDEX = NUMPART + 1

                ELSE IF (PROB .LE. 0.4511) THEN
```

```
                        NUMPART = NUMPART + 2
                        CALL NETA (PARTTYPES, CHARGE, INDEX)
                        INDEX = NUMPART + 1

                ELSE IF (PROB .LE. 0.4521) THEN

                        NUMPART = NUMPART + 2
                        PROB = RAN(RND)
                        CALL LAMBDAK (PARTTYPES, CHARGE, INDEX)
                        INDEX = NUMPART + 1

                ELSE

                        NUMPART = NUMPART + 1
                        PROB = RAN(RND)
                        CALL DELTAPI (PARTTYPES, CHARGE, INDEX)
                        PROB = RAN(RND)
                        INDEX = NUMPART + 1
                        IF (PROB .LE. 0.333333) THEN
                                PROB = RAN(RND)
                                GOTO 100
                        ELSE IF (PROB .LE. 0.666667) THEN
                                PROB = RAN(RND)
                                GOTO 200
                        ELSE
                                PROB = RAN(RND)
                                GOTO 300
                        ENDIF

                ENDIF

        ELSE IF (RES_LEVEL .EQ. 6) THEN

C     --- N* (1680) ---
C     Probabilities:  N/Pi           60.00%
C                     N/Eta           0.70%
C                     Lambda/K       Not Seen
C                     Delta/Pi       10.00%
C                     N/Rho          12.00%
C                     N/Pi-Pi        17.00%
C                     N/Gamma         0.30%

                IF (PROB .LE. 0.60) THEN

                        NUMPART = NUMPART + 2
                        PROB = RAN(RND)
                        CALL NPI (PARTTYPES, CHARGE, INDEX)
                        INDEX = NUMPART + 1

                ELSE IF (PROB .LE. 0.70) THEN

                        NUMPART = NUMPART + 1
                        PROB = RAN(RND)
                        CALL DELTAPI (PARTTYPES, CHARGE, INDEX)
                        PROB = RAN(RND)
                        INDEX = NUMPART + 1
                        IF (PROB .LE. 0.5000) THEN
                                PROB = RAN(RND)
                                GOTO 100
                        ELSE
                                PROB = RAN(RND)
                                GOTO 200
                        ENDIF

                ELSE IF (PROB .LE. 0.82) THEN

                        NUMPART = NUMPART + 3
                        CALL NRHO (PARTTYPES, CHARGE, INDEX)
                        INDEX = NUMPART + 1

                ELSE IF (PROB .LE. 0.99) THEN

                        NUMPART = NUMPART + 3
                        PROB = RAN(RND)
                        CALL NPIPI (PARTTYPES, CHARGE, INDEX)
                        INDEX = NUMPART + 1

                ELSE IF (PROB .LE. 0.997) THEN

                        NUMPART = NUMPART + 2
                        CALL NETA (PARTTYPES, CHARGE, INDEX)
```

```
                        INDEX = NUMPART + 1

                ELSE

                        NUMPART = NUMPART + 2
                        CALL NGAMMA (PARTTYPES, CHARGE, INDEX)
                        INDEX = NUMPART + 1

                ENDIF

            ELSE IF (RES_LEVEL .EQ. 7) THEN

C     --- N* (1700) ---
C     Probabilities:  N/Pi           10.00%
C                     N/Eta           4.79%
C                     Lambda/K        0.20%
C                     Delta/Pi       30.00%
C                     N/Rho          10.00%
C                     N/Pi-Pi        45.00%
C                     N/Gamma         0.01%

                IF (PROB .LE. 0.10) THEN

                        NUMPART = NUMPART + 2
                        PROB = RAN(RND)
                        CALL NPI (PARTTYPES, CHARGE, INDEX)
                        INDEX = NUMPART + 1

                ELSE IF (PROB .LE. 0.40) THEN

                        NUMPART = NUMPART + 1
                        PROB = RAN(RND)
                        CALL DELTAPI (PARTTYPES, CHARGE, INDEX)
                        PROB = RAN(RND)
                        INDEX = NUMPART + 1
                        IF (PROB .LE. 0.5000) THEN
                                PROB = RAN(RND)
                                GOTO 100
                        ELSE
                                PROB = RAN(RND)
                                GOTO 200
                        ENDIF

                ELSE IF (PROB .LE. 0.50) THEN

                        NUMPART = NUMPART + 2
                        CALL NETA (PARTTYPES, CHARGE, INDEX)
                        INDEX = NUMPART + 1

                ELSE IF (PROB .LE. 0.95) THEN

                        NUMPART = NUMPART + 3
                        CALL NRHO (PARTTYPES, CHARGE, INDEX)
                        INDEX = NUMPART + 1

                ELSE IF (PROB .LE. 0.9979) THEN

                        NUMPART = NUMPART + 2
                        CALL NETA (PARTTYPES, CHARGE, INDEX)
                        INDEX = NUMPART + 1

                ELSE IF (PROB .LE. 0.9999) THEN

                        NUMPART = NUMPART + 2
                        PROB = RAN(RND)
                        CALL LAMBDAK (PARTTYPES, CHARGE, INDEX)
                        INDEX = NUMPART + 1

                ELSE

                        NUMPART = NUMPART + 2
                        CALL NGAMMA (PARTTYPES, CHARGE, INDEX)
                        INDEX = NUMPART + 1

                ENDIF

            ELSE IF (RES_LEVEL .EQ. 8) THEN

C     --- N* (1710) ---
C     Probabilities:  N/Pi           15.00%
C                     N/Eta          20.00%
```

```
c                         Lambda/K        15.00%
c                         Sigma/K         10.00%
c                         Delta/Pi        15.00%
c                         N/Rho           12.50%
c                         N/Pi-Pi         12.50%

                          IF (PROB .LE.  0.15) THEN

                                  NUMPART = NUMPART + 2
                                  PROB = RAN(RND)
                                  CALL NPI (PARTTYPES, CHARGE, INDEX)
                                  INDEX = NUMPART + 1

                          ELSE IF (PROB .LE. 0.35) THEN

                                  NUMPART = NUMPART + 2
                                  CALL NETA (PARTTYPES, CHARGE, INDEX)
                                  INDEX = NUMPART + 1

                          ELSE IF (PROB .LE. 0.50) THEN

                                  NUMPART = NUMPART + 2
                                  PROB = RAN(RND)
                                  CALL LAMBDAK (PARTTYPES, CHARGE, INDEX)
                                  INDEX = NUPART + 1

                          ELSE IF (PROB .LE. 0.60) THEN

                                  NUMPART = NUMPART + 2
                                  PROB = RAN(RND)
                                  CALL SIGMAK (PARTTYPES, CHARGE, INDEX)
                                  INDEX = NUMPART + 1

                          ELSE IF (PROB .LE. 0.75) THEN

                                  NUMPART = NUMPART + 1
                                  PROB = RAN(RND)
                                  CALL DELTAPI (PARTTYPES, CHARGE, INDEX)
                                  PROB = RAN(RND)
                                  INDEX = NUMPART + 1

                                  IF (PROB .LE. 0.5000) THEN
                                          PROB = RAN(RND)
                                          GOTO 100
                                  ELSE
                                          PROB = RAN(RND)
                                          GOTO 200
                                  ENDIF

                          ELSE IF (PROB .LE. 0.875) THEN

                                  NUMPART = NUMPART + 3
                                  CALL NRHO (PARTTYPES, CHARGE, INDEX)
                                  INDEX = NUMPART + 1

                          ELSE

                                  NUMPART = NUMPART + 3
                                  PROB = RAN(RND)
                                  CALL NPIPI (PARTTYPES, CHARGE, INDEX)
                                  INDEX = NUMPART + 1

                          ENDIF

                  ELSE IF (RES_LEVEL .EQ. 9) THEN

c       --- N* (1720) ---
c       Probabilities:    N/Pi            15.00%
c                         N/Eta            3.50%
c                         Lambda/K         5.00%
c                         Sigma/K          5.00%
c                         Delta/Pi        10.00%
c                         N/Rho           50.00%
c                         N/Pi-Pi         11.50%

                          IF (PROB .LE. 0.15) THEN

                                          NUMPART = NUMPART + 2
                                  PROB = RAN(RND)
                                  CALL NPI (PARTTYPES, CHARGE, INDEX)
                                  INDEX = NUMPART + 1
```

```
            ELSE IF (PROB .LE. 0.185) THEN

                    NUMPART = NUMPART + 2
                    CALL NETA (PARTTYPES, CHARGE, INDEX)
                    INDEX = NUMPART + 1

            ELSE IF (PROB .LE. 0.235) THEN

                    NUMPART = NUMPART + 2
                    PROB = RAN(RND)
                    CALL LAMBDAK (PARTTYPES, CHARGE, INDEX)
                    INDEX = NUMPART + 1

            ELSE IF (PROB .LE. 0.285) THEN

                            NUMPART = NUMPART + 2
                    PROB = RAN(RND)
                    CALL SIGMAK (PARTTYPES, CHARGE, INDEX)
                    INDEX = NUMPART + 1

            ELSE IF (PROB .LE. 0.385) THEN

                    NUMPART = NUMPART + 1
                    PROB = RAN(RND)
                    CALL DELTAPI (PARTTYPES, CHARGE, INDEX)
                    PROB = RAN(RND)
                    INDEX = NUMPART + 1
                    IF (PROB .LE. 0.50000) THEN
                            PROB = RAN(RND)
                            GOTO 100
                    ELSE
                            PROB = RAN(RND)
                            GOTO 200
                    ENDIF

            ELSE IF (PROB .LE. 0.50) THEN

                            NUMPART = NUMPART + 3
                    PROB = RAN(RND)
                    CALL NPIPI (PARTTYPES, CHARGE, INDEX)
                    INDEX = NUMPART + 1

            ELSE

                    NUMPART = NUMPART + 3
                    CALL NRHO (PARTTYPES, CHARGE, INDEX)
                    INDEX = NUMPART + 1

            ENDIF

        ELSE IF (RES_LEVEL .EQ. 10) THEN

C     --- N* (2190) ---
C     Probabilities:  N/Pi            14.00%
C                     N/Eta            3.00%
C                     Lambda/K         0.30%
C                     NOTE:  62.7% of decay unaccounted for.

            IF (PROB .LE. 0.14) THEN

                            NUMPART = NUMPART + 2
                    PROB = RAN(RND)
                    CALL NPI (PARTTYPES, CHARGE, INDEX)
                    INDEX = NUMPART + 1

            ELSE IF (PROB .LE. 0.17) THEN

                    NUMPART = NUMPART + 2
                    CALL NETA (PARTTYPES, CHARGE, INDEX)
                    INDEX = NUMPART + 1

            ELSE IF (PROB .LE. 0.173) THEN

                    NUMPART = NUMPART + 2
                    PROB = RAN(RND)
                    CALL LAMBDAK (PARTTYPES, CHARGE, INDEX)
                    INDEX = NUMPART + 1

            ELSE
```

```
                          NBAD = NBAD + 1

                    ENDIF

              ELSE IF (RES_LEVEL .EQ. 11) THEN

C      --- N* (2220) ---
C      Probabilities:  N/Pi             18.00%
C                      N/Eta             0.50%
C                      Lambda/K          0.20%
C                      NOTE:  81.3% of decay unaccounted for.

                    IF (PROB .LE. 0.18) THEN

                                NUMPART = NUMPART + 2
                          PROB = RAN(RND)
                          CALL NPI (PARTTYPES, CHARGE, INDEX)
                          INDEX = NUMPART + 1

                    ELSE IF (PROB .LE. 0.185) THEN

                          NUMPART = NUMPART + 2
                          CALL NETA (PARTTYPES, CHARGE, INDEX)
                          INDEX = NUMPART + 1

                    ELSE IF (PROB .LE. 0.187) THEN

                          NUMPART = NUMPART + 2
                          PROB = RAN(RND)
                          CALL LAMBDAK (PARTTYPES, CHARGE, INDEX)
                          INDEX = NUMPART + 1

                    ELSE

                          NBAD = NBAD + 1

                    ENDIF

              ELSE IF (RES_LEVEL .EQ. 12) THEN

C      --- N* (2250) ---
C      Probabilities:  N/Pi             10.00%
C                      N/Eta             2.00%
C                      Lambda/K          0.30%
C                      NOTE:  87.7% of decay unaccounted for.

                    IF (PROB .LE. 0.10) THEN

                                NUMPART = NUMPART + 2
                          PROB = RAN(RND)
                          CALL NPI (PARTTYPES, CHARGE, INDEX)
                          INDEX = NUMPART + 1

                    ELSE IF (PROB .LE. 0.12) THEN

                          NUMPART = NUMPART + 2
                          CALL NETA (PARTTYPES, CHARGE, INDEX)
                          INDEX = NUMPART + 1

                    ELSE IF (PROB .LE. 0.123) THEN

                          NUMPART = NUMPART + 2
                          PROB = RAN(RND)
                          CALL LAMBDAK (PARTTYPES, CHARGE, INDEX)
                          INDEX = NUMPART + 1

                    ELSE

                          NBAD = NBAD + 1

                    ENDIF

              ELSE IF (RES_LEVEL .EQ. 13) THEN

C      --- N* (2600) ---
C      Probabilities:            N/Pi             5.00%
C                                NOTE:  95% of decay unaccounted for.

                    IF (PROB .LE. 0.05) THEN

                                NUMPART = NUMPART + 2
```

```
                        PROB = RAN(RND)
                        CALL NPI (PARTTYPES, CHARGE, INDEX)
                        INDEX = NUMPART + 1

                ELSE

                        NBAD = NBAD + 1

                ENDIF

        ENDIF           ! End for RES_LEVEL.

ENDIF                   ! End for RES.

RETURN
END


REAL*4 FUNCTION CROSS_SECTION (RES, NUM, E_CM)

C     Returns the total cross-section for pi-p interaction through the
C     i(th) occurance of the resonance RES.  RES = 1 for Deltas and
C     = 2 for N's.  Note the separate convention here.

      INTEGER*2       NUM, RES, J_FACT (15, 2)
      REAL*4          E_CM, GAMMA2 (15, 2), CS (15, 2), ELAS_BR (15, 2)
      REAL*4          E0 (15,2)

C     Resonace data.

      DATA    GAMMA2/ 13225.0, 19600.0, 62500.0, 22500.0, 90000.0,
      1               48400.0, 62500.0, 62500.0, 57600.0, 90000.0,
      1               0.0,     0.0,     0.0,     0.0,     0.0,
      2               40000.0, 15625.0, 22500.0, 22500.0, 24025.0,
      2               15625.0, 10000.0, 12100.0, 40000.0, 122500.0,
      2               160000.0, 90000.0, 160000.0, 0.0,   0.0 /

      DATA    CS/     94.8,    17.7,    14.5,    9.71,    9.62,
      1               9.54,    9.38,    9.21,    8.91,    4.68,
      1               0.00,    0.00,    0.00,    0.00,    0.00,
      2               31.0,    23.5,    22.5,    16.4,    15.4,
      2               15.2,    14.5,    14.2,    13.9,    6.21,
      2               5.97,    5.74,    3.86,    0.00,    0.00 /

      DATA    ELAS_BR/0.994,   0.30,    0.15,    0.10,    0.10,
      1               0.20,    0.175,   0.10,    0.40,    0.10,
      1               0.00,    0.00,    0.00,    0.00,    0.00,
      2               0.60,    0.55,    0.425,   0.60,    0.375,
      2               0.60,    0.10,    0.15,    0.15,    0.14,
      2               0.18,    0.10,    0.05,    0.00,    0.00 /

      DATA    E0/     1233.2, 1621.0, 1699.3, 1901.3, 1906.2,
      1               1911.1, 1920.8, 1930.5, 1949.8, 2421.0,
      1               0.00,   0.00,   0.00,   0.00,   0.00,
      2               1440.2, 1520.9, 1533.0, 1649.4, 1677.3,
      2               1677.3, 1699.3, 1710.2, 1721.1, 2189.4,
      2               2219.1, 2248.4, 2600.1, 0.00,   0.00 /

      DATA    J_FACT/ 4,      2,      4,      2,      6,
      1               2,      4,      6,      8,      12,
      1               0,      0,      0,      0,      0,
      2               2,      4,      2,      2,      6,
      2               6,      4,      2,      4,      8,
      2               10,     10,     12,     0,      0 /


C     Calculate the cross section.

      CROSS_SECTION = 0.50 * FLOAT(J_FACT (NUM, RES)) * CS (NUM, RES) *
      1               ELAS_BR (NUM, RES) * GAMMA2 (NUM, RES) /
      1               (GAMMA2 (NUM, RES) + 4*((E_CM - E0 (NUM, RES))**2))

      RETURN
      END


      SUBROUTINE NGAMMA (PARTTYPES, CHARGE, INDEX)

C     Decays resonance into N + Gamma.

      INTEGER*4       RND
```

```
      INTEGER*2       CHARGE, INDEX, PARTTYPES (1:20)
      REAL*4          PROB

      COMMON  /MONTE/         PROB, RND

      IF (CHARGE .EQ. 1) THEN
              PARTTYPES (INDEX) = 14
              PARTTYPES (INDEX + 1) = 1
      ELSE IF (CHARGE .EQ. 0) THEN
              PARTTYPES (INDEX) = 13
              PARTTYPES (INDEX + 1) = 1
      ELSE
              PROB = RAN(RND)
              CALL NPI (PARTTYPES, CHARGE, INDEX)
      ENDIF

      RETURN
      END


      SUBROUTINE NRHO (PARTTYPES, CHARGE, INDEX)

C     Decays resonance into N + Gamma.

      INTEGER*4       RND
      INTEGER*2       CHARGE, INDEX, PARTTYPES (1:20)
      REAL*4          PROB

      COMMON  /MONTE/         PROB, RND

C     Code

      IF (CHARGE .EQ. 1) THEN
              PARTTYPES (INDEX) = 14
C             PARTTYPES (INDEX + 1) = 22
      ELSE IF (CHARGE .EQ. 0) THEN
              PARTTYPES (INDEX) = 13
C             PARTTYPES (INDEX + 1) = 22
      ELSE
              CALL NPIPI (PARTTYPES, CHARGE, INDEX)
              RETURN
      ENDIF

      IF (PROB .LE. 0.5000) THEN
              PARTTYPES (INDEX + 1) = 7
              PARTTYPES (INDEX + 2) = 7
      ELSE
              PARTTYPES (INDEX + 1) = 8
              PARTTYPES (INDEX + 2) = 9
      ENDIF

      RETURN
      END


      SUBROUTINE DELTAPI (PARTTYPES, CHARGE, INDEX)

C     Decays resonance into Delta + Pi.

      INTEGER*4       RND
      INTEGER*2       CHARGE, INDEX, PARTTYPES (1:20)
      REAL*4          PROB

      COMMON  /MONTE/         PROB, RND

C     Code

      IF (CHARGE .EQ. 2) THEN
              IF (PROB .LE. 0.5000) THEN
                      PARTTYPES (INDEX) = 8
                      CHARGE = 1
              ELSE
                      PARTTYPES (INDEX) = 7
              ENDIF
      ELSE IF (CHARGE .EQ. -1) THEN
              IF (PROB .LE. 0.5000) THEN
                      PARTTYPES (INDEX) = 9
                      CHARGE = 0
              ELSE
                      PARTTYPES (INDEX) = 7
              ENDIF
```

```
        ELSE IF ((CHARGE .EQ. 0) .OR. (CHARGE .EQ. 1)) THEN
                IF (PROB .LE. 0.33333) THEN
                        PARTTYPES (INDEX) = 8
                        CHARGE = CHARGE - 1
                ELSE IF (PROB .LE. 0.66667) THEN
                        PARTTYPES (INDEX) = 7
                ELSE
                        PARTTYPES (INDEX) = 9
                        CHARGE = CHARGE + 1
                ENDIF
        ELSE
                PRINT *, 'Charge error -- DELTAPI'
        ENDIF

        RETURN
        END


        SUBROUTINE NETA (PARTTYPES, CHARGE, INDEX)

C       Decays resonance into N + eta.

        INTEGER*2       CHARGE, INDEX, PARTTYPES (1:20)

        IF (CHARGE .EQ. 1) THEN
                PARTTYPES (INDEX) = 14
                PARTTYPES (INDEX + 1) = 17
        ELSE IF (CHARGE .EQ. 0) THEN
                PARTTYPES (INDEX) = 13
                PARTTYPES (INDEX + 1) = 17
        ELSE
                PRINT *, 'Charge conservation error -- NETA'
        ENDIF

        RETURN
        END


        SUBROUTINE LAMBDAK (PARTTYPES, CHARGE, INDEX)

C       Decays resonance into Lambda + K.

        INTEGER*4       RND
        INTEGER*2       CHARGE, INDEX, PARTTYPES (1:20)
        REAL*4          PROB

        COMMON  /MONTE/         PROB, RND

C       Code.

        IF (CHARGE .EQ. 1) THEN
                PARTTYPES (INDEX) = 11
                PARTTYPES (INDEX + 1) = 18
        ELSE
                IF (PROB .LE. 0.5000) THEN
                        PARTTYPES (INDEX) = 10
                ELSE
                        PARTTYPES (INDEX) = 16
                ENDIF
                PARTTYPES (INDEX + 1) = 18
        ENDIF

        RETURN
        END


        SUBROUTINE NPI (PARTTYPES, CHARGE, INDEX)

C       Decays resonance into N and Pi.

        INTEGER*4       RND
        INTEGER*2       CHARGE, INDEX, PARTTYPES (1:20)
        REAL*4          PROB

        COMMON  /MONTE/         PROB, RND

C       Code.

        IF (CHARGE .EQ. 0) THEN
                IF (PROB .LE. 0.33333333) THEN
                        PARTTYPES (INDEX) = 13
```

```
                PARTTYPES (INDEX + 1) = 7
        ELSE IF (PROB .LE. 0.66666667) THEN
                PARTTYPES (INDEX) = 14
                PARTTYPES (INDEX + 1) = 9
        ELSE
                PARTTYPES (INDEX) = 13
                PARTTYPES (INDEX + 1) = 8
        ENDIF
ELSE IF (CHARGE .EQ. 1) THEN
        IF (PROB .LE. 0.5000) THEN
                PARTTYPES (INDEX) = 14
                PARTTYPES (INDEX + 1) = 7
        ELSE
                PARTTYPES (INDEX) = 13
                PARTTYPES (INDEX + 1) = 8
        ENDIF
ELSE IF (CHARGE .EQ. 2) THEN
        PARTTYPES (INDEX) = 14
        PARTTYPES (INDEX + 1) = 8
ELSE IF (CHARGE .EQ. -1) THEN
        PARTTYPES (INDEX) = 13
        PARTTYPES (INDEX + 1) = 9
ELSE
        PRINT *, 'ERROR -- Invalid charge (NPI).'
ENDIF

RETURN
END

SUBROUTINE NPIPI (PARTTYPES, CHARGE, INDEX)

C       Decays resonance into N and 2 Pi's.

INTEGER*4       RND
INTEGER*2       CHARGE, INDEX, PARTTYPES (1:20)
REAL*4          PROB

COMMON  /MONTE/         PROB, RND

C       Code.

IF (CHARGE .EQ. 2) THEN
        IF (PROB .LE. 0.5000) THEN
                PARTTYPES (INDEX) = 14
                PARTTYPES (INDEX + 1) = 8
                PARTTYPES (INDEX + 2) = 7
        ELSE
                PARTTYPES (INDEX) = 13
                PARTTYPES (INDEX + 1) = 8
                PARTTYPES (INDEX + 2) = 8
        ENDIF
ELSE IF (CHARGE .EQ. 1) THEN
        IF (PROB .LE. 0.33333333) THEN
                PARTTYPES (INDEX) = 14
                PARTTYPES (INDEX + 1) = 8
                PARTTYPES (INDEX + 2) = 9
        ELSE IF (PROB .LE. 0.66666667) THEN
                PARTTYPES (INDEX) = 14
                PARTTYPES (INDEX + 1) = 7
                PARTTYPES (INDEX + 2) = 7
        ELSE
                PARTTYPES (INDEX) = 13
                PARTTYPES (INDEX + 1) = 8
                PARTTYPES (INDEX + 2) = 7

        ENDIF
ELSE IF (CHARGE .EQ. 0) THEN

        IF (PROB .LE. 0.33333333) THEN
                PARTTYPES (INDEX) = 13
                PARTTYPES (INDEX + 1) = 7
                PARTTYPES (INDEX + 2) = 7
        ELSE IF (PROB .LE. 0.66666667) THEN
                PARTTYPES (INDEX) = 13
                PARTTYPES (INDEX + 1) = 8
                PARTTYPES (INDEX + 2) = 9
        ELSE
                PARTTYPES (INDEX) = 14
                PARTTYPES (INDEX + 1) = 7
                PARTTYPES (INDEX + 2) = 9
        ENDIF
```

```
      ELSE IF (CHARGE .EQ. -1) THEN

              IF (PROB .LE. 0.5000) THEN
                      PARTTYPES (INDEX) = 13
                      PARTTYPES (INDEX + 1) = 7
                      PARTTYPES (INDEX + 2) = 9
              ELSE
                      PARTTYPES (INDEX) = 14
                      PARTTYPES (INDEX + 1) = 9
                      PARTTYPES (INDEX + 2) = 9
              ENDIF

      ELSE

              PRINT *, 'ERROR -- Invalid charge (NPIPI).'

      ENDIF

      RETURN
      END


      SUBROUTINE SIGMAK (PARTTYPES, CHARGE, INDEX)

C     Decays resonance into N and 2 Pi's.

      INTEGER*4       RND
      INTEGER*2       CHARGE, INDEX, PARTTYPES (1:20)
      REAL*4          PROB

      COMMON  /MONTE/         PROB, RND

C     Code.

      IF (CHARGE .EQ. 2) THEN

              PARTTYPES (INDEX) = 19
              PARTTYPES (INDEX + 1) = 11

      ELSE IF (CHARGE .EQ. 1) THEN

              IF (PROB .LE. 0.5000) THEN
                      PARTTYPES (INDEX) = 20
                      PARTTYPES (INDEX + 1) = 11
              ELSE
                      PARTTYPES (INDEX) = 19
                      PROB = RAN(RND)
                      IF (PROB .LE. 0.5000) THEN
                              PARTTYPES (INDEX + 1) = 10
                      ELSE
                              PARTTYPES (INDEX + 1) = 16
                      ENDIF
              ENDIF

      ELSE IF (CHARGE .EQ. 0) THEN

              IF (PROB .LE. 0.5000) THEN
                      PARTTYPES (INDEX) = 20
                      PROB = RAN(RND)
                      IF (PROB .LE. 0.5000) THEN
                              PARTTYPES (INDEX + 1) = 10
                      ELSE
                              PARTTYPES (INDEX + 1) = 16
                      ENDIF
              ELSE
                      PARTTYPES (INDEX) = 21
                      PARTTYPES (INDEX + 1) = 11
              ENDIF

      ELSE IF (CHARGE .EQ. -1) THEN

              PARTTYPES (INDEX) = 21
              PROB = RAN(RND)
              IF (PROB .LE. 0.5000) THEN
                      PARTTYPES (INDEX + 1) = 10
              ELSE
                      PARTTYPES (INDEX + 1) = 16
              ENDIF

      ELSE
```

```
        PRINT *, 'ERROR -- Invalid charge (SIGMAK).'
      ENDIF

      RETURN
      END
```

Appendix C

The NNEveSim code

This appendix contains the complete VAX **FORTRAN** implementation of the `NNEveSim` model. The code contained herein is actually the code used in the calculations *with* the inclusion of Fermi momentum (c.f. Chapter IV). The code is, again, fairly well documented, and should help the reader understand its structure. As with the code for the `PiNRP` model (Appendix B), this program does not at all correspond to the **FORTRAN** 77 standard.

```
      PROGRAM NN_COLLIDE
C
C     PISCAT common block for particle data.  See below comments for details.
C
C     File:             UDISK:[RMARQ.PISCAT.SOURCE]COMMON.FOR
C     By:               Ron Marquardt
C     Created:          December 6, 1988
C     Modified:         ???
C
C     Project:          PISCAT program for thesis research (pion rescattering)
C
C     The design of this program is simple.  A nuclear reaction at zero
C     impact parameter is simulated by following a single nulceon of
C     given energy through a tube of length of the nuclear diameter
C     (currently Au).  When an nucleon/nucleon interaction has been
C     determined to occur (via an analytic Monte Carlo), a single particle
C     is read from a list previously created by the LUND particle Monte
C     Carlo simulator.  The history of this produced particle (the
C     "primary") is then followed within the nucleus.  If an interaction
C     occurs, the proper resonances are created and decayed and the
C     final state particles (the "secondaries") are added to the output
C     list.  The secondaries are NOT propagated through the nucleus.
C
C     NOTE:  Only one particle is "created" per nucleon/nucleon interaction
C     and thus total cross sections remain arbitrary.  However, the
C     observed results hold, on average, for a multiplicity greater than
C     one.  Also, the secondary paricles are not allowed to reinteract.
C     On average, these contributions will be second order.  So on an
C     event by event basis, the model is simplistic, but it should
C     recover the general physics in the average.
C
C     Variables:  (as of 12/6/88)
C

C     Global

      INCLUDE 'DISK$USER:[RMARQ.PISCAT.SOURCE]COMMON.FOR'

      DATA     A_TARG  /197/
      DATA     Z_TARG  /96/

C     Local

C     The following arrays are used to bin the results of the program.
C     All indices from 1:7 refer to particle type (see below parameter
C     definitions), from 0:2*NRBIN to distance l along the zero impact
C     parameter tube that the nucleons travel within, 0:NRBIN to the
C     radial distance at which a final state particle was produced,
C     and 0:NPBIN to the primary particle momentum (binned from 0 to 2.5
C     GeV/c only).  The array NUM_PART bins all final state particles
C     of each type.  NPART_R is the number of final state particles
C     produced at a binned radius from the nuclear center (1: primaries
C     only, 2: secondaries only, and 3: both).  NPART_L is the number
C     of primary particles, of given type, created at a distance l along
C     the nuclear tube, having momentum binned from 0 to 2.5 GeV.
C     NPART_PLP bins only those particles found in NPART_L (primaries)
C     that also undergo an interaction before leaving the nucleus.  Thus
C     a probability of a given produced particle interacting can be
C     recovered from NPART_L and NPART_PLP.  Finally, NPART_NOINT keeps
C     track of particles of a given type and momentum that fail to
C     interact (should be consistent with prior numbers).

      PARAMETER NPBIN = 250
      PARAMETER NRBIN = 64

      INTEGER*4        NUM_PART (1:5, 1:7), NPART_R (1:3, 1:7, 0:NRBIN+1)
      INTEGER*4        NPART_L    (1:7, 0:2*NRBIN, 0:NPBIN+1)
      INTEGER*4        NPART_PLP (1:7, 0:2*NRBIN, 0:NPBIN+1)
      INTEGER*4        NPART_NOINT (1:7, 0:NPBIN+1)

C     We define often used array look-up values to make the code more
C     readable.  The parameters LFREE, AU_RAD, and AU_DIAM are fixed
C     for normal nuclear matter (LFREE) and the dimensions of the
C     Au nucleus (AU_RAD and AU_DIAM).

      PARAMETER        PI_P     = 1
      PARAMETER        PI_N     = 2
      PARAMETER        K_P      = 3
      PARAMETER        K_N      = 4
      PARAMETER        K_0      = 5
```

```
      PARAMETER          K_0B     = 6
      PARAMETER          PI_0     = 7
      PARAMETER          LFREE    = 1.953125        ! nucleon mean free path (fm)
      PARAMETER          P_FERMI  = 0.028
      PARAMETER          PI       = 3.141592653589793
      PARAMETER          AU_RAD   = 6.4             ! Au nuclear radius (fm)
      PARAMETER          AU_DIAM  = 12.8            ! Au nuclear diameter (fm)

C     Next are the variables used for housekeeping and general program
C     utility.

      LOGICAL            DATA_EOF, DORES
      INTEGER*4          RND, NP_HIGH, NBAD_DECAY, NEVE, TYPE
      INTEGER*2          PTYPE (1:7), I, J, K, STATUS
        INTEGER*2        CHARGE, NUM, PARTTYPES (1:20), OLD_TYPE
      REAL*4             PENERGY, MASS, GAMMA, P, PX, PY, PZ, PT, P_REL
      REAL*4             LFORM (1:7), L, L0, R, A, B, D, SECNDS
      REAL*4             DIST, VX, VY, VZ, ALPHA
      CHARACTER*80       INFILE, OUTFILE

      DATA     NP_HIGH /0/
      DATA     NBAD_DECAY      /0/
      DATA     PTYPE    /17, -17, 28, -28, 19, -19, 23/
      DATA     LFORM    /1.414, 1.414, 0.3997, 0.3997, 0.3965, 0.3965, 1.4621/

C     Code

C     We read in the user parameters.  The input and output file names
C     are given as well as the choice to form resonances or not and
C     a fudge factor to relate the standard formation time of particles
C     to a parameterized value (alpha*lform) to simulate the effects of
C     a co-moving frame.

      WRITE (*, 10)
 10   FORMAT ('$Input file specification for particle list:  ')
      READ (*, 20) J, INFILE
 20   FORMAT (Q, A80)

      OPEN (UNIT=20, FILE=INFILE(1:J), STATUS='UNKNOWN',
      1        FORM='UNFORMATTED')

      WRITE (*, 30)
 30   FORMAT ('$Input file specification for output summary:  ')
      READ (*, 40) J, OUTFILE
 40   FORMAT (Q, A80)

      OPEN (UNIT=25, FILE=OUTFILE(1:J), STATUS='UNKNOWN',
      1        FORM='FORMATTED')

      CALL INPUT_YN ('$Form pion/nucleon resonances?  \', DORES, 'Y')

      CALL INPUT_REAL ('$Co-moving frame formation time factor?  \',
      1        ALPHA, 1.0)

C     Now, loop around until we run out of LUND events in the file.
C     We initialize the loop control variables and the random number
C     seed.

      NEVE = 0
      DATA_EOF = .FALSE.
      RND = NINT (SECNDS (0.0) * 2000.0) + 1

      DO WHILE (.NOT. DATA_EOF)

C     The distance until the next nucleon/nucleon collision is given
C     simply by an analytic Monte Carlo with respect to the nuclear
C     mean free path.  As the natural log of a random number (0 < # < 1)
C     is negative, the Monte Carlo term is subtracted at each stage.

              L = 0.0

              DO WHILE (L .LE. AU_DIAM)

                      L = L - LOG(RAN(RND))*LFREE

                      IF (L .LE. AU_DIAM) THEN

C     Now that we know a valid nucleon/nucleon collision occurs, we
C     pick the next particle on the list and use it.

                      READ (20, ERR=100) TYPE, PENERGY, MASS,
```

```
     1                        GAMMA, P, PX, PY, PZ, PT
                    NEVE = NEVE + 1

                    I = 1
                    DO WHILE (PTYPE (I) .NE. TYPE)
                            I = I + 1
                    ENDDO
                    TYPE = I

C     We calculate the relative momentum of the pion-nucleon system by
C     including a Fermi momentum distribution for the nucleons.  We first
C     determine the magnitude of P, by Monte Carlo, and then the projection
C     of P, in the direction of travel of the pion, again via an
C     appropriate Monte Carlo technique.

                    P_REL = RAN(RND)*P_FERMI
                    P_REL = P_REL*COS(PI*RAN(RND))

                    P = P + P_REL
                    IF (P .LT. 0) P = -P

C
C     In general, we try to use TYPE defined below for the particles
C     we are dealing with:
C
C             TYPE              PARTICLE
C
C     ---------------------------------------------
C               1                 Pi +
C               2                 Pi -
C               3                 K  +
C               4                 K  -
C               5                 K0 (long)
C               6                 K0 (short)
C               7                 Pi 0
C
C     However, the codes returned by LUND (and stored in the PTYPE array)
C     give K0 and K0 bars.  As these are not the proper eigenstates,
C     we must translate to K0 (long) and K0 (short) used in this
C     program.  Thus we randomly change the type below.  This should
C     not profoundly change things, but it will make everything consistent.
C

                    IF ((TYPE .EQ. 5) .OR. (TYPE .EQ. 6)) THEN
                            IF (RAN(RND) .LT. 0.5000) THEN
                                    TYPE = 5
                            ELSE
                                    TYPE = 6
                            ENDIF
                    ENDIF

                      OLD_TYPE = TYPE

                    IF (ISKAON (TYPE) .OR. (.NOT. (DORES))) THEN

                            NUM_PART (1, TYPE) = NUM_PART (1, TYPE) + 1

C                           R = ABS(L - AU_RAD)*NRBIN/AU_RAD
C                           NPART_R (1, TYPE, NINT(R)) =
C    1                              NPART_R (1, TYPE, NINT(R)) + 1
C                           NPART_R (3, TYPE, NINT(R)) =
C    1                              NPART_R (3, TYPE, NINT(R)) + 1
C
C                           IF (P .LE. 2.5) THEN
C                             NPART_L (TYPE, NINT(L*NRBIN/AU_RAD),
C    1                          100*NINT(P)) = NPART_L (TYPE,
C    2                          NINT(L*NRBIN/AU_RAD), 100*NINT(P)) + 1
C                             NPART_NOINT (TYPE, 100*NINT(P)) =
C    1                        NPART_NOINT (TYPE, 100*NINT(P)) + 1
C                           ELSE
C                                   NP_HIGH = NP_HIGH + 1
C                           ENDIF

                    ELSE

C     Now we need to know the distance to the nuclear surface.  We take
C     the current position, (x = r, y = 0, z = 0) and a unit vector
C     in the direction of travel and find DIST, the distance to the outer
C     surface.

                            IF (TYPE .EQ. 1) THEN
                                    CHARGE = 1
```

```
                                ELSE IF (TYPE .EQ. 2) THEN
                                        CHARGE = -1
                                ELSE
                                        CHARGE = 0
                                ENDIF

                                VZ = PZ / P
                                R  = (L - (AU_DIAM/2.0))
                                B  = R**2 - AU_RAD**2
                                A  = 2*R*VZ

                                DIST = (-A + SQRT(A**2 - 4*B))/2.0

C       Next we consider the formation time of the particle, due to
C       the uncertainty principle, and varying inversely proportional
C       to the mass.

                                L0 = GAMMA*LFORM(TYPE)*ALPHA
                                 D = - LOG (RAN(RND))*L0

                                DIST = DIST - D
                                IF (DIST .LT. 0) THEN
                                        STATUS = -1
                                ELSE
                                CALL PIN_COLLIDE (CHARGE, P, DIST, STATUS,
     1                                  NUM, PARTTYPES)
                                        D = D + DIST
                                ENDIF

C       STATUS indicates the result of the Monte Carlo.  If 0, then
C       no collision occured in the nucleus.  If 1, then a collision
C       happened and the particles in PARTTYPES are vaild.  Lastly, if
C       2, then a collision occured, but a bad decay (see PIN_COLLIDE
C       for details) resulted.

                        IF (STATUS .EQ. 0) THEN

                                NUM_PART (1, TYPE) = NUM_PART (1, TYPE) + 1
                                NUM_PART (4, TYPE) = NUM_PART (4, TYPE) + 1

C                               R = ABS(L - AU_RAD)*NRBIN/AU_RAD
C                               NPART_R (1, TYPE, NINT(R)) =
C    1                                  NPART_R (1, TYPE, NINT(R)) + 1
C                               NPART_R (3, TYPE, NINT(R)) =
C    1                                  NPART_R (3, TYPE, NINT(R)) + 1
C
C                               IF (P .LE. 2.5) THEN
C                                 NPART_L (TYPE, NINT(L*NRBIN/AU_RAD),
C    1                              100*NINT(P)) = NPART_L (TYPE,
C    2                              NINT(L*NRBIN/AU_RAD), 100*NINT(P)) + 1
C                                 NPART_NOINT (TYPE, 100*NINT(P)) =
C    1                            NPART_NOINT (TYPE, 100*NINT(P)) + 1
C                               ELSE
C                                       NP_HIGH = NP_HIGH + 1
C                               ENDIF
C
                        ELSE IF (STATUS .EQ. 2) THEN

                                NBAD_DECAY = NBAD_DECAY + 1

                        ELSE IF (STATUS .EQ. -1) THEN

C       Formed outside the nucleus.

                                NUM_PART (1, TYPE) = NUM_PART (1, TYPE) + 1
                                NUM_PART (5, TYPE) = NUM_PART (5, TYPE) + 1

                        ELSE

C       Now we have a list of particles to go through, in the Geant
C       particle code form.  We translate to our own, and bin if it is a
C       particle which we are interested in.

C                       IF (P .LE. 2.5) THEN
C                         NPART_L (OLD_TYPE, NINT(L*NRBIN/AU_RAD),
C    1                      100*NINT(P)) = NPART_L (OLD_TYPE,
C    2                      NINT(L*NRBIN/AU_RAD), 100*NINT(P)) + 1
C                         NPART_PLP (OLD_TYPE, NINT(L*NRBIN/AU_RAD),
C    1                      100*NINT(P)) = NPART_PLP (OLD_TYPE,
C    2                      NINT(L*NRBIN/AU_RAD), 100*NINT(P)) + 1
C                       ELSE
```

```
C                               NP_HIGH = NP_HIGH + 1
C                       ENDIF
                NUM_PART (3, TYPE) = NUM_PART (3, TYPE) + 1

                DO I = 1, NUM

                        TYPE = LUND_TYPE (PARTTYPES(I))
                        IF (TYPE .GT. 0) THEN

                                NUM_PART (1, TYPE) = NUM_PART (1, TYPE) + 1
                                NUM_PART (2, TYPE) = NUM_PART (2, TYPE) + 1
C                               R = L - AU_RAD
C                               R = SQRT(R**2 + 2*R*D*PZ/P + D**2)*NRBIN/AU_RAD
C                               NPART_R (2, TYPE, NINT(R)) =
C     1                                 NPART_R (2, TYPE, NINT(R)) + 1
C                               NPART_R (3, TYPE, NINT(R)) =
C     1                                 NPART_R (3, TYPE, NINT(R)) + 1
C
                        ENDIF

                ENDDO

                                                        ENDIF

                                                ENDIF
                                        ENDIF
                                ENDDO

                GOTO 200

 100            DATA_EOF = .TRUE.

 200  ENDDO

      WRITE (25, 500)
 500  FORMAT (1X, '( Data summary for 2,500 Si/Au events. )')

      IF (DORES) THEN
                WRITE (25, 510)
 510  FORMAT (1X, '( Resonances included. )')
      ELSE
                WRITE (25, 520)
 520  FORMAT (1X, '( NO Resonances included. )')
      ENDIF

      WRITE (25, 522) NEVE
 522  FORMAT (1X, 'NEVE: ', I12)

      WRITE (25, 524) NBAD_DECAY
 524  FORMAT (1X, 'NBAD_DECAY: ', I12)

      WRITE (25, 1)
 1    FORMAT (1X, ' ')

      WRITE (25, 540)
 540  FORMAT (1X, '( Histogram of the total number of particles )')
      WRITE (25, 555)
 555  FORMAT (1X, '1-ALL, 2-SECONDARY, 3-MADE RES, 4-IN NUC NO-RES, 5-'
      1         'OUTSIDE NUC NO-RES')

      WRITE (25, 1)

      DO I = 1, 7

                WRITE (25, 550) I, NUM_PART (1, I), NUM_PART (2, I),
      1                         NUM_PART (3, I), NUM_PART (4, I),
      2                         NUM_PART (5, I)
 550  FORMAT (1X, I3, I12, 5X, I12, 5X, I12, 5X, I12, 5X, I12)

      ENDDO

      WRITE (25, 1)

C     WRITE (25, 560)
C 560 FORMAT (1X, '( NPART_R histograms... )')
C
C     WRITE (25, 1)
C
C     WRITE (25, 570)
C 570 FORMAT (1X, '(Primaries only... )')
```

```
C
C     DO I = 1, 7
C             WRITE (25, 1)
C             WRITE (25, 580) I
C 580 FORMAT (1X, '(Particle type ', I3)
C
C             DO J = 0, NRBIN
C             WRITE (25, 590) J, NPART_R (1, I, J)
C 590 FORMAT (1X, I3, 5X, I12)
C             ENDDO
C
C     ENDDO
C
C     WRITE (25, 1)
C
C     IF (DORES) THEN
C
C             WRITE (25, 571)
C 571         FORMAT (1X, '(Secondaries only... )')
C
C             DO I = 1, 7
C                     WRITE (25, 1)
C                     WRITE (25, 581) I
C 581 FORMAT (1X, '(Particle type ', I3)
CC
C                     DO J = 0, NRBIN
C                             WRITE (25, 591) J, NPART_R (2, I, J)
C 591 FORMAT (1X, I3, 5X, I12)
C                     ENDDO
C
C             ENDDO
C
C             WRITE (25, 1)
C
C             WRITE (25, 573)
C 573 FORMAT (1X, '(Primaries and Secondaries... )')
C
C             DO I = 1, 7
C                     WRITE (25, 1)
C                     WRITE (25, 583) I
C 583 FORMAT (1X, '(Particle type ', I3)
C
C                     DO J = 0, NRBIN
C                             WRITE (25, 593) J, NPART_R (3, I, J)
C 593 FORMAT (1X, I3, 5X, I12)
C                     ENDDO
C
C             ENDDO
C
C     ENDIF
C
C     WRITE (25, 1)
C     WRITE (25, 1)
C
C     IF (.NOT. DORES) GOTO 700
C
C     WRITE (25, 600)
C 600 FORMAT (1X, '(P(INT))')
C
C     DO I = 1, 7
C
C             IF (ISKAON (I)) GOTO 640
C
C             WRITE (25, 1)
C             WRITE (25, 1)
C
C             WRITE (25, 610) I
C 610 FORMAT (1X, 'Particle:  ', I3)
C
C             WRITE (25, 1)
C
C             DO J = 0, 2*NRBIN
C
C                     DO K = 1, 250
C
C                             NPART_L (I, J, 0) = NPART_L (I, J, 0) +
C     1                               NPART_L (I, J, K)
C                             NPART_PLP (I, J, 0) = NPART_PLP (I, J, 0) +
C     1                               NPART_PLP (I, J, K)
C
C                     ENDDO
```

```
c
c     DIST used as a temporary real # here.
c
c                       IF (NPART_L (I, J, 0) .NE. 0) THEN
c                       DIST = FLOAT(NPART_PLP (I, J, 0)) /
c     1                         FLOAT(NPART_L (I, J, 0))
c                       ELSE
c                       DIST = 0.0
c                       ENDIF
c
c                       WRITE (25, 620) J, DIST
c 620 FORMAT (1X, 3X, I4, 5X, G14.8)
c
c               ENDDO
c
c 640 CONTINUE
c     ENDDO
c
c 700 CONTINUE

c     DO I = 1, 7
c
c               WRITE (25, 600) I
c 600 FORMAT (1X, '( Particle # [NPART_L] ', I2)
c
c               WRITE (25, 1)
c
c               DO J = 0, 2*100
c
c                       WRITE (25, 1)
c                       WRITE (25, 610) J
c 610 FORMAT (1X, '( Distance ', I4)
c
c                       DO K = 0, NPBIN
c
c                               WRITE (25, 620) K, NPART_L (I, J, K)
c 620 FORMAT (1X, I3, 5X, I12)
c
c                       ENDDO
c
c               ENDDO
c
c     ENDDO
c
c     DO I = 1, 7
c
c               WRITE (25, 630) I
c 630 FORMAT (1X, '(Particle # [NPART_PLP] ', I2)
c
c               WRITE (25, 1)
c
c               DO J = 0, 2*NRBIN
c
c                       WRITE (25, 1)
c                       WRITE (25, 640) J
c 640 FORMAT (1X, '(Distance ', I4)
c
c                       DO K = 0, NPBIN
c
c                               WRITE (25, 650) K, NPART_PLP (I, J, K)
c 650 FORMAT (1X, I3, 5X, I12)
c
c                       ENDDO
c
c               ENDDO
c
c     ENDDO
c
c     DO I = 1, 7
c
c               WRITE (25, 660) I
c 660 FORMAT (1X, '(Particle # [NPART_NOINT] ', I2)
c
c               WRITE (25, 1)
c
c               DO K = 0, NPBIN
c
c                       WRITE (25, 670) K, NPART_NOINT (I, J)
c 670 FORMAT (1X, I3, 5X, I12)
c
c               ENDDO
```

```
C
C     ENDDO

      CLOSE (20)
      CLOSE (25)

      END

      LOGICAL FUNCTION ISKAON (TYPE)

C     Returns whether particle of type TYPE is a Kaon.

      INTEGER*4       TYPE

      IF ((TYPE .EQ. 3) .OR. (TYPE .EQ. 4) .OR. (TYPE .EQ. 5) .OR.
      1         (TYPE .EQ. 6)) THEN
                ISKAON = .TRUE.
      ELSE
                ISKAON = .FALSE.
      ENDIF

      RETURN
      END


      INTEGER*4 FUNCTION LUND_TYPE (TYPE)

C     Translates from Geant to LUND particle codes (0 if uninteresting).

      INTEGER*2       TYPE, I

      IF        (TYPE .EQ. 8)  THEN
                I = 1
      ELSE IF (TYPE .EQ. 9)  THEN
                I = 2
      ELSE IF (TYPE .EQ. 11) THEN
                I = 3
      ELSE IF (TYPE .EQ. 12) THEN
                I = 4
      ELSE IF (TYPE .EQ. 10) THEN
                I = 5
      ELSE IF (TYPE .EQ. 16) THEN
                I = 6
      ELSE IF (TYPE .EQ. 7)  THEN
                I = 7
      ELSE
                I = 0
      ENDIF

      LUND_TYPE = I

      RETURN
      END
```

Bibliography

[1] T. Abbott et al. A single arm spectrometer detector system for high energy heavy-ion experiments. (Unpublished), May 1989.

[2] M. Aguilar-Benitez et al. Review of particle properties. *Phys. Lett.*, 170B, April 1986.

[3] T. Åkesson et al. Search for quark deconfinement: Strangeness production in pp, dd, $p\alpha$, and $\alpha\alpha$ collisions at $\sqrt{s_{NN}} = 31.5$ and 44 GeV. *Phys. Rev. Lett.*, 55(23):2535–2538, December 1985.

[4] B. Andersson et al. A general model for jet fragmentation. *Z. Phys. C*, 20, 1983.

[5] R. Brun et al. GEANT3–User's Guide. Technical Report DD/EE/84-1, CERN Data Handling Division, CERN, September 1987.

[6] M. Fæssler. Experimental results from ion reactions at high energy ($E/A \gg 800 MeV$). Technical Report CERN-EP/84-78, European Organization for Nuclear Research, June 1984. Lecture given at the Winter College for Fundamental Nuclear Physics, 1984.

[7] N. K. Glendenning and J. Rafelski. Kaons and quark-gluon plasma. *Phys. Rev.*, C31(3), March 1985.

[8] K. Gottfried and V. Weisskopf. *Concepts of Particle Physics*, volume I. Clarendon Press, Oxford, 1986.

[9] M. Gyulassy. Quark matter 1987: Concluding remarks. In H. Satz, H. J. Specht, and R. Stock, editors, *Proceedings of the Sixth International Conference on Ultra-Relativistic Nucleus-Nucleus Collisions—Quark Matter 1987*, 1988.

[10] I. Hughes. *Elementary Particles.* Cambridge University Press, Cambridge, 1985.

[11] Y. Iga et al. Monte carlo event generator MCMHA for high energy hadron-nucleus collisions and intranuclear cascade interactions. *Z. Phys. C*, 38, 1988.

[12] S. L. Kang et al. The deconfining phase transition–influence of hadron resonances and strangeness. *Phys. Lett.*, B174(2), July 1986.

[13] C. M. Ko and L. Xia. K^+/π^+ enhancement in heavy-ion collisions. *Phys. Rev.*, C38(1), July 1988.

[14] J. Kuti. Quark confinement and the quark model. Technical report, Central Research Institute for Physics, Budapest, Hungary, 1977. Notes from CERN lectures.

[15] R. J. Ledoux. Experiments in relativistic heavy ion physics. Notes from lecture given March 17, 1988.

[16] K. S. Lee et al. Quark-gluon plasma versus hadron gas. What one can learn from hadron abundances. *Phys. Rev.*, C37(4), April 1988.

[17] T. W. Ludlam. Relativistic heavy ion beams at Brookhaven: High energy nuclear beams in the AGS and RHIC. In *Proceedings of the Second International Conference on Nucleus-Nucleus Collisions*, June 1985.

[18] T. Matsui et al. Strangeness production in ultrarelativistic heavy ion collisions. I. Chemical kinetics in the quark-gluon plasma. Technical Report CTP #1320, MIT Center for Theoretical Physics, November 1985.

[19] L. McLerran. The physics of the quark-gluon plasma. *Rev. Mod. Phys.*, 58(4), October 1986.

[20] Y. Miake, G. Stevens, et al. Preliminary spectrometer results from E-802. In H. Satz, H. J. Specht, and R. Stock, editors, *Proceedings of the Sixth International Conference on Ultrarelativistic Nucleus-Nucleus Collisions—Quark Matter 1987*, 1988.

[21] S. Nagamiya. Towards new forms of nuclear matter with relativistic heavy-ion collisions. In C. Détraz et al., editors, *Proceedings of the Third International Conference on Nucleus-Nucleus Collisions*, 1988.

[22] S. Nagamiya and M. Gyulassy. *High-Energy Nuclear Collisions*, chapter 3. Plenum Publishing Corp., New York, 1982.

[23] I. Otterlund et al. Particle densities in ultrarelativistic heavy ion reactions extrapolated from proton–nucleus collisions. *Z. Phys. C*, 20, 1983.

[24] D. Perkins. *Introduction to High Energy Physics*. Addison-Wesley, Menlo Park, CA, 1987.

[25] B. Schwarzschild. Nuclear matter *in extremis* sought with ultrarelativistic heavy-ion beams. *Physics Today*, pages 17–21, March 1988.

[26] E. Segrè. *Nuclei and Particles: An Introduction to Nuclear and Subnuclear Physics*. Benjamin/Cummings, Menlo Park, CA, 1977.

[27] H. Sorge et al. Relativistic quantum molecular dynamics approach to nuclear collisions at ultrarelativistic energies. Technical report, Institut für Theoretische Physik, Universität Frankfurt, 1988.

[28] M. Tannenbaum. Relativistic heavy ion collisions. Technical Report BNL 40066, Brookhaven National Laboratory, June 1987. Invited talk, CEBAF 1987 Summer Workshop.

[29] L. van Hove. Possible manifestation of quark-gluon plasma in ultrarelativistic nucleus-nucleus collisions. In *Proceedings of the Second International Conference on Nucleus-Nucleus Collisions*, June 1985.