5-4-2007

Development of Cosmic Ray Simulation Program -- Earth Cosmic Ray Shower (ECRS)

Sampath S. Hakmana Witharana

Recommended Citation

Hakmana Witharana, Sampath S., "Development of Cosmic Ray Simulation Program -- Earth Cosmic Ray Shower (ECRS)." Dissertation, Georgia State University, 2007.
https://scholarworks.gsu.edu/phy_astr_diss/12

DEVELOPMENT OF COSMIC RAY SIMULATION PROGRAM – EARTH COSMIC RAY SHOWER (ECRS)

by

Sampath Sanjeewa Hakmana Witharana

Under the direction of Xiaochun He

ABSTRACT

ECRS is a program for the detailed simulation of extensive air shower initiated by high energy cosmic ray particles. In this dissertation work, a Geant4 based ECRS simulation was designed and developed to study secondary cosmic ray particle showers in the full range of Earth's atmosphere. A proper atmospheric air density and geomagnetic field are implemented in order to correctly simulate the charged particles interactions in the Earth's atmosphere.

The initial simulation was done for the Atlanta (33.46^0 N , 84.25^0 W) region. Four different types of primary proton energies (10^9, 10^{10}, 10^{11} and 10^{12} eV) were considered to determine the secondary particle distribution at the Earth's surface. The geomagnetic field and atmospheric air density have considerable effects on the muon particle distribution at the Earth's surface.

The muon charge ratio at the Earth's surface was studied with ECRS simulation for two different geomagnetic locations: Atlanta, Georgia, USA and Lynn Lake, Manitoba, Canada. The simulation results are shown in excellent agreement with the data from NMSU-WIZARD/CAPRICE and BESS experiments at Lynn Lake. At low momentum, ground level muon charge ratios show latitude dependent geomagnetic effects for both Atlanta and Lynn Lake from the simulation. The simulated charge ratio is 1.20 ± 0.05 (without geomagnetic field), 1.12 ± 0.05 (with geomagnetic field) for Atlanta and 1.22 ± 0.04 (with geomagnetic field) for Lynn Lake. These types of studies are very important for analyzing secondary cosmic ray muon flux distribution at the Earth's surface and can be used to study the atmospheric neutrino oscillations.

Keywords: *Cosmic rays, Monte Carlo simulation, Air shower, Geomagnetic Field, Geant4*

DEVELOPMENT OF COSMIC RAY SIMULATION PROGRAM – EARTH COSMIC RAY SHOWER (ECRS)

by

Sampath Hakmana Witharana

A Dissertation Submitted in Partial Fullfilment of the Requirements for the Degree of

Doctor of Philosophy

in the College of Arts and Sciences

Georgia State University

2007

DEVELOPMENT OF COSMIC RAY SIMULATION PROGRAM –

EARTH COSMIC RAY SHOWER (ECRS)

by

SAMPATH HAKMANA WITHARANA

Major Professor:	Dr. Xiaochun He
Committee:	Dr. Unil Perera
	Dr. Richard Miller
	Dr. William Nelson
	Dr. Brian Thoms

Electronic Version Approved:

Office of Graduate Studies
College of Arts and Sciences
Georgia State University
May 2007

To my parents and wife Inosha

Acknowledgments

This work would never have been completed without the help and the support of many people.

First and foremost, I want to thank my advisor Dr. Xiaochun He for having given me the possibility of studying in the nuclear physics group at Georgia State University (GSU) and for the valuable guidance, discussion, and the help gave me to reach the results presented in this dissertation. I want to also express my deep and sincere gratitude to him for his enthusiasm, inspiration, and great efforts to explain things clearly and strongly. His wide knowledge and logical way of thinking have been of great value for me.

I wish to express my warm and sincere thanks to Dr. Unil Perera, who gave me the opportunity to join the graduate program at GSU and for his essential assistance and advice in this study. I owe my sincere gratitude to Dr. William Nelson, Dr. Richard Miller, and Dr. Brian Thoms, for their detailed review, guidance, and excellent advice have been of great value in this study.

I am deeply grateful to Carola Butler, who has spent a lot of time in proof-reading and friendly help on this thesis. Her extensive discussions around my work and interesting explorations in operations have been very helpful for the study.

I would also like to thank the nuclear physics group at GSU, for all their help and kindness during the study. In particular, I am grateful to Dr. John Wilson, Dr. Jun Ying, Qu, Robert, Pushpa, Christopher, past members of the group, Dr. Petitt, Dr. Gobinda, and Chris who have always been willing to hear and give good advice.

During this work I have collaborated with many colleges and friends for whom I

have great regard, and I wish to extend my warmest to all those who have helped me with my work in the Department of Physics and Astronomy at GSU.

Last, but not least, I owe my loving thanks to my parents and wife Inosha. They have lost a lot due to my research abroad. Without their encouragement, understanding, and unconditional love it would have been impossible for me to finish this work.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

Since the discovery of cosmic rays early in the twentieth century, many experiments have been developed to study cosmic ray radiation. A big effort has been made to study cosmic rays' propagation, acceleration, and interaction with the electromagnetic radiation and the interstellar matter. These studies have provided important information about the interstellar matter and magnetic fields in interstellar matter, remnants of supernova, and spinning neutron stars, as well as specific information concerning cosmic ray sources. However, fundamental questions concerning the nature of dark matter, the matter-antimatter symmetry of the universe, particle origin and acceleration are still not completely understood. From the astrophysics point of view, there are some difficulties in fully understanding the origin of cosmic rays with energies above 10^{15} eV. It is unknown, at the present time, the origin of primary proton energies above about 4×10^{19} eV, based on GZK [1] cutoff theory. This theory pointed out that the universe is not transparent to protons with energies above about 4×10^{19} eV as they interact with the 2.7K microwave background radiation.

It is also known that cosmic ray radiation is involved in different scientific applications including soft fails in electronic logic or memories, [2] muon radiography for detection of high Z material [3], and monitoring the environmental or biological effects of cosmic ray particles [4] etc. In order to study these in details, it is necessary to know the atmospheric cosmic ray flux of high energetic charged particles. Even though many experiments have been performed in the study of the cosmic ray flux, detailed cosmic ray shower development is far too complex to be fully described by only experimental data. Therefore it is important to model by simulation of transport and interaction of each individual cosmic ray shower particle employing our present knowledge on interactions, decays, and particle transport in matter.

In this dissertation, details of a new simulation toolkit, so–called ECRS (Earth Cosmic Ray Shower) for cosmic ray studies, are presented. ECRS is a useful and flexible tool to study high energy cosmic ray interactions, to support the interpretation of extensive air shower measurements, and to optimize the design of future cosmic ray experiments. The dissertation is organized as follows:

A brief introduction to the history of cosmic ray research is provided in chapter 2. Following the introduction, primary and secondary cosmic ray fluxes are presented with their properties and common detection methods. More details of both indirect and direct detection techniques, and their limitations are also presented.

The details of Geant4 toolkit are given in Chapter 3. Geant4 plays a major role in the simulation study, and the ECRS simulation programme is developed based on Geant4. More specific details of Geant4, including architecture, geometry, physics processes, and particle tracking are presented.

The novel features of the ECRS simulation are discussed in Chapter 4. The important and distinguishing features of ECRS are presented with geomagnetic field implementation. The geomagnetic field consistency results are also provided.

The analysis of secondary cosmic ray showers for four different primary energies from the ECRS simulation are described in Chapter 5. Both of the magnetic field and the density effects are presented. The chapter also contains preliminary results discussed with the theoretical model.

The details of an extensive cosmic ray air shower study of the muon charge ratio from the ECRS simulation are presented in chapter 6. The details of regenerated primary particle energy distribution and simulation procedure are also discussed. The simulated results are compared with two sets of experimental data.

The conclusion of this dissertation work and the discussion of the future work that can be carried out from this work are presented in Chapter 7.

Chapter 2

Cosmic Rays

2.1 History

The study of radioactive materials in the period from 1898 to 1912 was very interesting because this field offered direct insight into the nature of the atom, whose structure was still unknown. Electrometers were often used to measure the very small flux of particles coming from radioactive materials. In these detectors, the current leaking between two electrodes under the voltage called leakage current can be observed without ionization caused by the passage of charged particles. Victor Hess studied this phenomenon by taking electrometers onto lakes where there should have been less contamination (no change in leakage) and into caves (leakage disappeared). Finally, in 1912, Victor Hess and two assistants solved the problem by flying in a balloon to an altitude of about 6 km and discovered the evidence of a very penetrating radiation coming from outside our atmosphere. Figures 2.1 (a) and (b) [5] show the results of Victor Hess's study of cosmic ray radiation. This was the first observation of cosmic rays. In general, cosmic rays are high-energy charged particles, mainly protons, originating in outer space, that travel at nearly the speed of light and strike the Earth from all directions. For this work, Hess was awarded the Nobel prize in 1936. His work was immediately followed by more detailed studies such as that of Kolhrster, who showed that the particle flux increases very rapidly with altitude, with a 10-fold increase at only 10 km. Finally, Pfotzer showed in 1936 that the flux does not continue to increase, but reaches a peak at about 15 km, after which it diminishes rapidly. Fig. 2.1 (c) [6] shows the exponential increase in cosmic rays with altitude up to a height of about 15 km.

Before man-made accelerators reached high energies (GeV), from the period of the 1930s to 1950s, cosmic rays were used as a main source of particles for high energy physics investigations. During this period new types of subatomic particles including the positron and the muon were investigated. Although these applications continue, since the dawn of the space age, the main focus of cosmic ray research has been directed towards astrophysical investigations of where cosmic rays originate, how they get accelerated to such high velocities, what role they play in the dynamics of the Galaxy, and what their composition tells us about matter from outside the solar system.

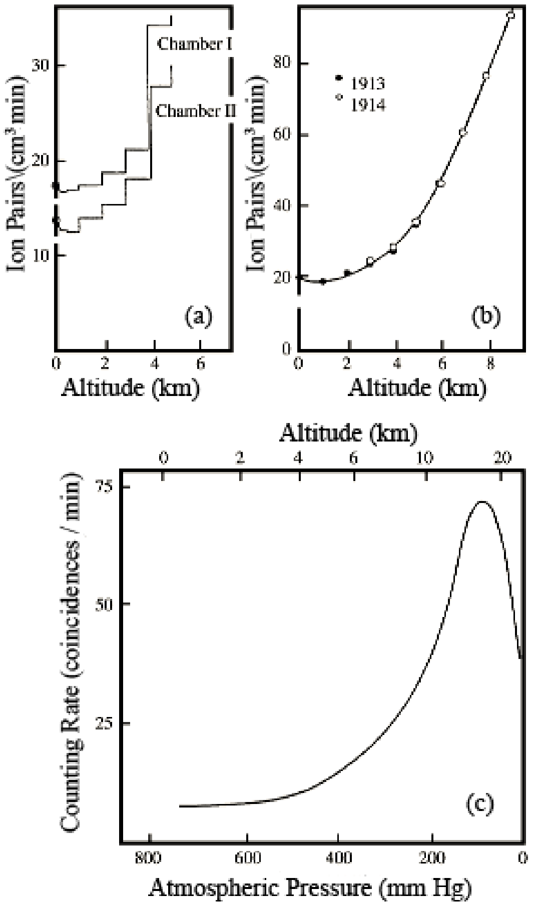

Figure 2.1: (a) and (b) [5] in 1912, Victor Hess used a balloon to take two ionization chambers up to a height of about 6 km, and showed that the flux of particles increases with altitude. (c) [6] Pfotzer showed in 1936 that there is an exponential increase in cosmic rays with altitude up to a height of about 15 km, after which it decreases.

The origin of high-energy cosmic rays remains an important unsolved problem in astrophysics. Possible sources are the Sun, other stars, and more exotic objects, such as supernovae and their remnants, neutron stars and black holes, as well as active galactic nuclei and radio galaxies. Cosmic rays are mainly categorized into three major groups [7]:

- **Galactic Cosmic Rays (GCR)** – The source of GCR is not known, but they originate far outside our solar system. GCR are the most typical cosmic rays, and their flux in the solar system is modulated by solar activity.
- **Solar Cosmic Rays (SCR)** – SCR energetic particles originate mostly from solar flares, coronal mass ejections, and shocks in the interplanetary medium. SCR particles have energies up to several hundred MeV/nucleon. During strong solar flares, the flux of cosmic rays at the Earth's orbit can increase by hundred percent over the course of hours or days.
- **Anomalous Cosmic Rays (ACR)** – ACR are most likely produced by neutral atoms in interstellar space. These are ionized by solar UV radiation or charge exchange with the solar wind. They are then picked up and convected back to the outer heliosphere.

The acceleration of these cosmic rays is still partly an open question, especially in the ultra high energy range. It is currently thought that galactic cosmic rays up to about 10^{15} eV are accelerated at shocks driven by supernova explosions via the mechanism of diffusive shock acceleration. In this theory, charged particles are accelerated as they scatter within the converging plasma flow across the shock[8]. The close association of energetic particles with collisionless shocks observed in interplanetary space, as well as those at the Earth's bow shock, provide direct, convincing evidence that astrophysical shocks accelerate particles to high energies.

Each of the above categorized cosmic rays is divided into two major groups, namely primary cosmic rays and secondary cosmic rays. Particles which start showers are referred to as primary cosmic rays. The particles created in the air shower are known as secondary cosmic rays. Numerous secondary particles are produced when primary cosmic rays interact with the atmosphere that pass right through everyone's body every second.

2.2 Primary Cosmic Ray Distribution

Primary cosmic rays are stable charged particles that have been accelerated to enormous energies by astrophysical sources somewhere in our universe. Particle detectors in satellites, and a serious of more precise measurements have determined that primary cosmic ray particles consist of 92% protons and 6% alpha particles, with the remainder being heavy nuclei. Figure 2.2 [9] shows the major components of primary cosmic rays as a function of energy.

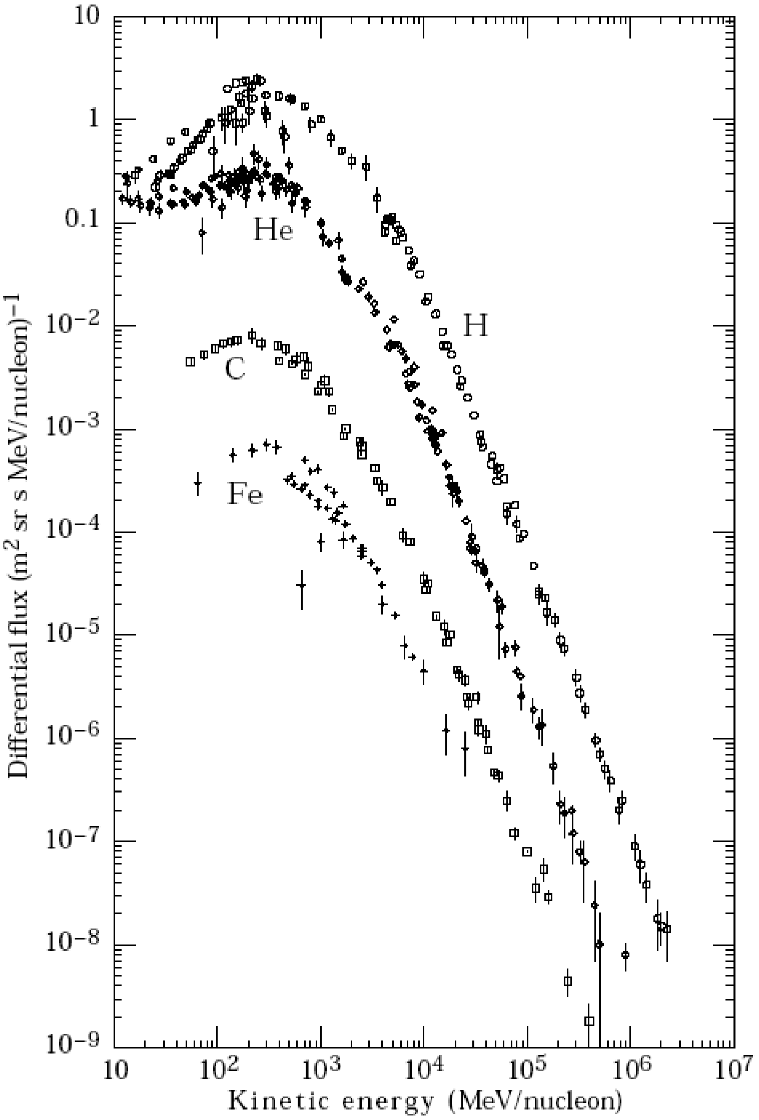

Figure 2.2: Major components of primary cosmic rays as a function of energy.

The primary cosmic ray energy spectrum is represented by a power-law function dN/dE $\propto$ E$^{\gamma}$ in a wide range from about 10^9 to 10^{20} eV, where dN/dE is the differential flux, E is the energy per nucleon, and γ is the differential spectral index of the cosmic ray flux. Figure 2.3 shows the primary cosmic power-law flux distribution as a function of energy. The spectra of primary particles have been measured in a variety of ways, including particle detectors in satellites, balloon–borne detectors, counters that monitor the flux of neutrons and muons at the Earth's surface, and at higher energies by wide area arrays of particle detectors. In the spectrum, the flux decreases, from values of several 1000 (m^2 sr s)$^{-1}$ at GeV energies, to values below 0.01 (km^2 sr year)$^{-1}$, at energies exceeding 10^{15} eV. The spectrum can be divided into four regions with very distinct behavior. There are energies below 1 GeV, energies from 1 GeV to a knee around 4×10^{15}eV, a second knee at about 400×10^{15} eV, and an ankle above 4×10^{20}eV [10, 11].

The first energy region, with energies below 1 GeV, has a very distinctive character. Its shape and cut-off is strongly dependent on the phase of the solar cycle, which means that low-energy, incoming charged particles are modulated by the solar wind. Significant correlations between solar activity (eleven year cycle) and the intensity of low energy cosmic rays have been observed in several experiments[12]. In addition, the lower-energy cosmic rays are affected by the geomagnetic field, which they must penetrate to reach the top of the atmosphere. The second energy region, with energies between 1 GeV and the first knee region, is characterized by differential spectral index $\gamma = 2.7$. Current studies suggest that these cosmic rays are most likely produced in supernova explosions. This is based on their distinctive chemical composition. The elements carbon, nitrogen, oxygen, and those from the iron group, have the same relative abundance in the solar system and in cosmic rays. For the third energy region, from the second knee to the ankle at $4{\times}10^{20}$ eV, the spectral index γ steepens to 3.2. The origin of the cosmic rays in this region is unclear and is a subject of much debate. Above the ankle, the spectrum flattens again to spectral index γ = 2.8, and this is interpreted as a cross–over from the steeper, galactic component, to a harder, extra–galactic source [13, 14].

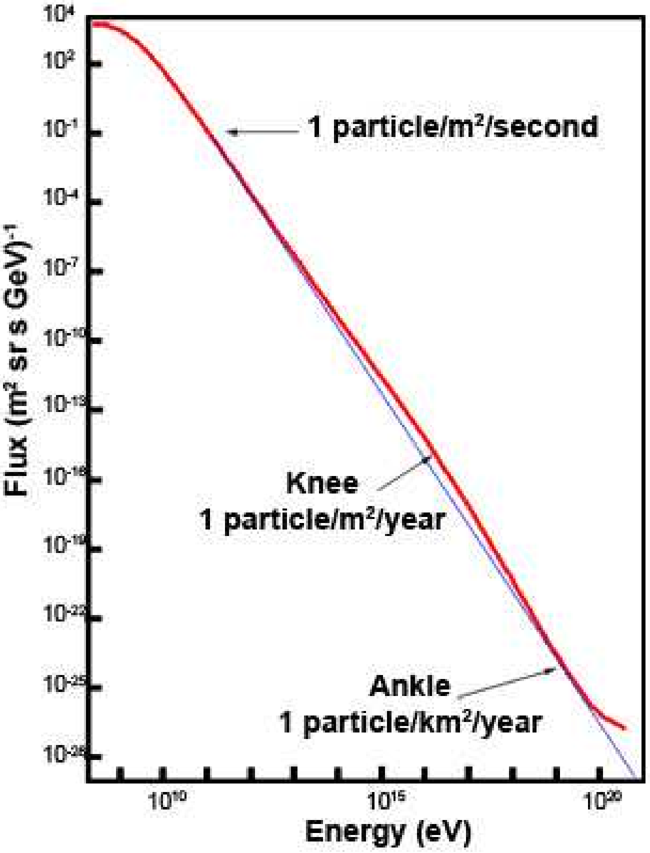

Figure 2.3: The primary energy spectrum of cosmic rays shows changes of power-law at the knee region. The blue straight line and red curved line represent the theoretical power-law, and experimental measurement, respectively.

2.3 Secondary Cosmic Ray Shower

Primary high energy cosmic ray particles, mostly protons, interact with atmospheric matter primarily through the strong interaction and create numerous particles called secondary cosmic rays. Figure 2.4 shows possible secondary cosmic ray particle components and branches in the atmosphere. Secondary cosmic ray cascade components can be divided into three groups: muonic, hadronic, electromagnetic.

When a high energy primary particle impacts atomic nuclei of air atoms in the upper atmosphere pions and kaons are produced. Mesons are produced in the hadronic decay and give rise to the muonic component. Mostly, charged pions decay within a relatively short distance (meters) into muons and neutrinos. Neutral pions decay into two gamma–rays. A muon interacts very little with matter except through ionization. Because of this, muons can travel large distances and commonly reach the Earth's surface. Most muons are produced at an altitude about 15 km in the atmosphere and lose about 3 GeV to ionization before reaching the Earth's surface. Some of these muons decay into an electron or positron, a neutrino and an antineutrino before reach the Earth's surface.

The electromagnetic component consists of electrons and positrons initiated by decay of neutral and charged mesons. Muon decay is the dominant source of low-energy electrons at sea level. The gamma rays from the neutral pions may create new particles, electrons and positrons, by the pair-creation process. Electrons and positrons may produce more gamma rays by bremsstrahlung radiation. Secondary photons then undergo pair production or produce Compton electrons. The secondary electrons and positrons can radiate more photons in a multiplicative process. This multiplication continues until the ioniza-

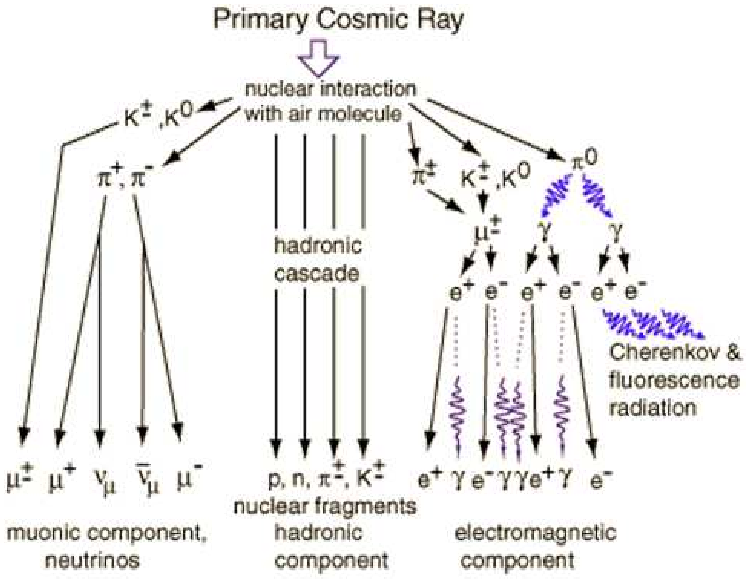

Figure 2.4: Schematic representation of several possible secondary particle branches of the atmospheric cosmic ray cascade.

tion losses start to dominate the radiation losses, dissipating all their energy in excitation and ionization of atoms. Knock-out electrons also make a small contribution to shower development at low energy.

2.4 Cosmic Ray Particle Detection Methods

A number of different detection techniques have been developed for cosmic ray studies. These detector techniques can be divided into two groups, namely direct experiment and indirect experiment. While direct experiments are focused on primary particle detection at high altitude (above 15 km), indirect experiments detect extensive air shower (EAS), or secondary cosmic rays in ground based observation. Indeed direct measurements are generally capable of detecting individual primary cosmic ray nuclei, and provide a direct measurement of their spectra. Ground–based indirect measurements can infer information about the nature and energy of primary particles only from the EAS originating from their interaction with air nuclei. Indirect measurements can be subdivided into different detection techniques such as detection of charge and muon particles in ground–based air shower experiments, detection of deep underground muons, Cherenkov light detectors, fluorescent light detectors, and a combination of all of the above techniques. Figure 2.5 shows common direct and indirect detection methods for cosmic ray shower studies.

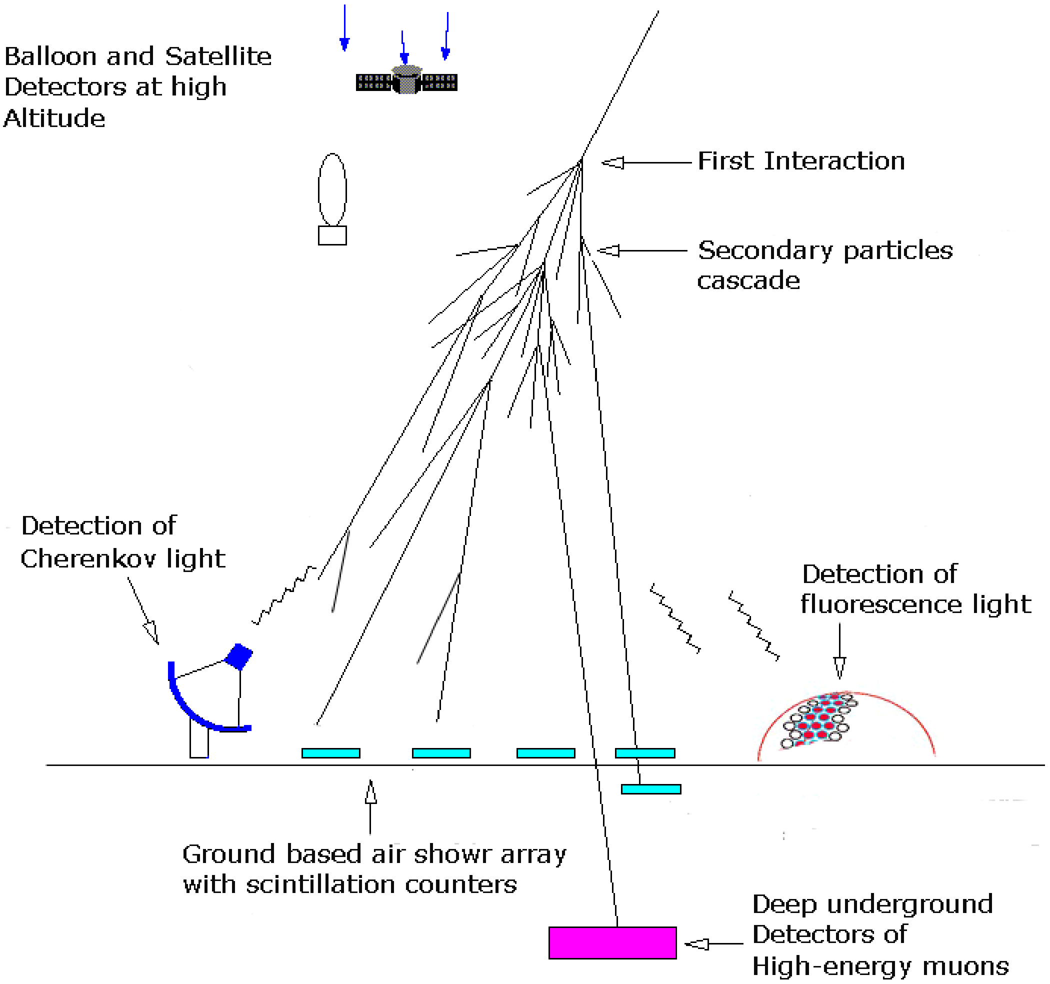

Figure 2.5: Schematic diagram of cosmic ray detection methods for cosmic ray showers. Balloon and satellite experiments are used to measure the primary cosmic ray components. Cherenkov, fluorescent, and ground–based air shower array experiments are carried out to measure the secondary shower components.

2.4.1 Direct Experimental Methods

The main purpose of this type of experiment is to detect primary cosmic ray particles and measure their charges, energies, and directions by using balloon or satellite–based instruments at high altitudes. Some of the experimental measurements, like those of the GSFC group [15] in balloons and the proton satellite experiment [16], have reached the second knee and a little beyond, but with limited charge and energy resolution. Calorimetric experiments include emulsion chamber instruments, such as JACEE [21] and RUNJOB [18], and have accumulated enough exposure to reach the knee. The ATIC [19] experiment has shown good charge and energy measurements, but below the knee. Other experiments have also incorporated transition radiation detectors and gaseous Chereknov counters for energy measurement, such as the TRACER [20] balloon instrument, which has shown excellent charge resolution.

2.4.2 Indirect Experimental Methods

Indirect measurements are mainly concerned with energies above the first knee region, studying the secondary air shower energy, lateral distribution, longitudinal distribution, and propagation time.

In air, particles travel faster than the speed of the light. The result is that they emit light called Cerenkov radiation. The Cerenkov light is beamed in the forward direction of the air shower, and can be measured with optical detectors. Another indirect technique, called fluorescent light detection is used to measure the energy and arrival time of primary particles. The amount of light collected can then be used to estimate the total amount of energy released from the fluorescent light detectors. The scintillation light is detected using a lens or a mirror, and imaged onto a camera located at the focal plane. This fluorescence technique can be made to work on clear moonless nights using very fast camera elements to record light flashes a few microseconds in duration. Some of the experiments, such as HiRes [22], DICE [23], BLANCA [24], and CACTI [25], measure the profile of the cascade and total energy deposition by either Cherenkov radiation or air fluorescence event by event technique.

The other common indirect detection method is ground based air shower array technique, which has explored cosmic rays with energies above 100 TeV. This type of experiment consists of many detectors, typically 100 or more, located over hundreds of meters associated with data acquisition and Global Positioning System (GPS) units. The direction of the primary cosmic rays can be calculated by measuring the time of arrival of the shower front at the individual stations. Also, air shower arrays can be used to study the composition and energy spectra of cosmic rays in the knee region. They measure the lateral distributions of electrons and muons with scintillation counters, and also sample the hadronic components of showers. More well known experiments of this kind such as KASKADE [26], TIBET [27], and Akeno [28] detect the particles in showers at selected observation level, measuring lateral distribution, energy and time of arrival. Also, AUGUR [83], BLANCA , and HiRes-CASA measure the longitudinal and lateral profiles in shower event by event measurement.

Chapter 3

Geant4 – Simulation Toolkit

3.1 Introduction

Geant4 is a result of two independent projects done at CERN, Europe and KEK, Japan [29] in 1993. Both groups sought to investigate robust computer software framework for particle detector simulation for high energy particle physics by improving the existing Geant3 [30] program. Later, these two activities merged and a proposal was submitted to the CERN detector research and development committee to construct a simulation program based on an object-oriented programming. Initially, more than 100 scientists and engineers from more than 10 experiments in Europe, Russia, Japan, Canada, and the United Sates engaged in the collaboration. In December 1998, the first research and development was completed, and the Geant4 collaboration was established in January 1999 to continue the development and refinement of the toolkit and to provide maintenance and user support.

Currently, Geant4 has been applied to different research fields such as medicine, astronomy, space science, and radiation protection. Several studies have reported the validation of the physics processes in Geant4 and its applications [31, 32].

3.2 Geant4 Design

Geant4 is a general purpose toolkit, written in an advanced software engineering technique based on object-oriented methodology and C++ for the simulation of the passage of particles through matter. It provides a comprehensive set of particles, detector geometries, materials, tracking management, detector response and visualization, and user interface. It also provides extensive physics processes such as electromagnetic, hadronic and optical physics to describe interactions of particles with matter across a wide energy range. In addition there is a large set of utilities, including a set of random number generators, physics units and constants, and particle data group compliant particle management.

Users can construct stand-alone applications built upon an object-oriented framework. Its modular architecture enables the user to pick only necessary modular components according to the specific needs. The common modular and hierarchical structure for the Geant4 toolkit is shown in Fig. 3.1, where sub categories are linked by a uni–directional flow of dependencies.

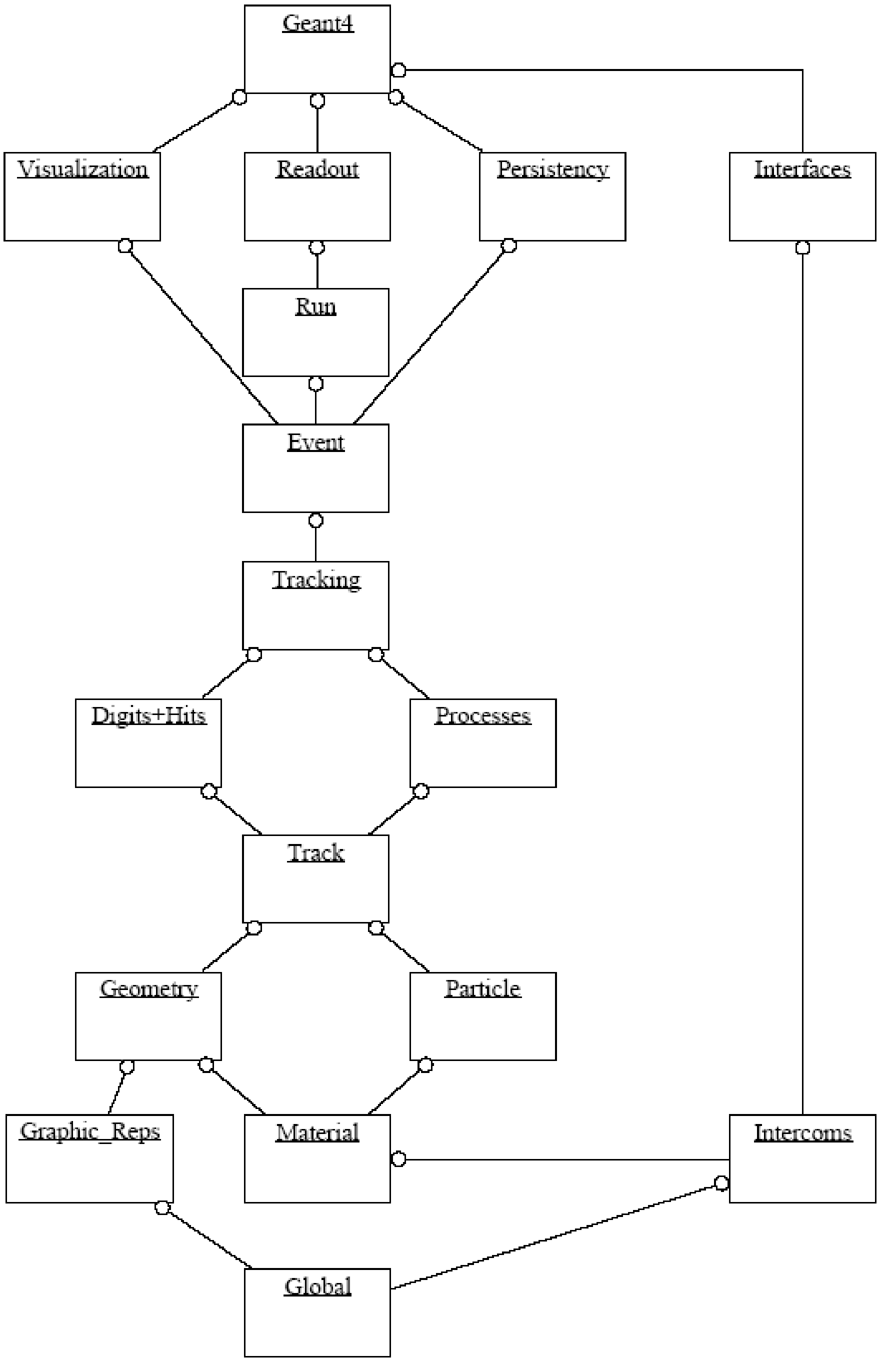

Figure 3.1: Hierarchical structure of the Geant4 toolkit. The open circles on the joining lines represent a relationship to the adjoined category.

Categories at the bottom of the diagram in Fig. 3.1 are used by virtually all higher categories and provide the foundation of the toolkit. The global category covers the system units, constants, numerics, and random number handling. The two categories, material and particles, are used to describe the physical properties of particles and materials for the simulation of particle–matter interactions. The geometry category provides the ability to describe a geometrical structure and propagate a particle efficiently through it. The track category contains classes for tracks and steps, used by the processes category. The processes category contains implementations of modules of physical interactions: electromagnetic interactions of leptons, photons, hadrons, ions, and hadronic. All processes are invoked by the tracking category, which manages their contributions to the evolution of the track's state, and provides information in the sensitive volume for hits and digitization. The event category manages the events in terms of their tracks and the run category manages collections of the events that share a common beam and detector implementation.

In general, the key categories of the simulation of the passage of particles through matter are:

- Geometry and materials
- Particle interaction in matter
- Event and tracking management
- Digitization and hit management
- Visualization and User interface

3.2.1 Geometry and Materials

Geant4 has the ability to create a geometrical model with a large number of components of different shapes and materials, and to define sensitive elements that record information needed to simulate detector responses.

There are two concepts used in defining the geometry of detector construction: the outer “world” and the inner “daughter” volumes. The world volume is a reference volume for the first daughter volume of the detector. The world volume is conceived as the largest volume that includes all the detector components in three-dimensional space. Each detector component is defined as a geometrical volume whose center is placed at a point in the reference frame of another volume. These volumes incorporate materials which are defined as elements or compounds. Compounds are defined by their atomic composition as given by their chemical formulae or weight fraction. Their densities at a given temperature and pressure are also defined. In general, geometry in Geant4 refers to the volumes built in the simulation with appropriate materials. These geometries can be defined as sensitive components that record information on particle hits.

3.2.2 Particle Interactions in Matter

Particle interaction in matter is associated with particle types and their energies. Once primary particles are initiated with appropriate physics processes, they are tracked through the system until they stop, decay, or are transported beyond the limits of the world volume. The generation of primary particles can be done using the event generator or the particle gun class, which can create a beam of particles incorporated with its type, position, direction of motion, and kinetic energy.

For propagating particles between two interaction points, their space and time coordinates, as well as their energy have to be updated. Therefore, it is useful to distinguish a particular initial and final state for a particle interaction or decay. Interaction or decay is determined by the cross section of a hadronic reaction, together with the atmospheric density distribution along the flight path, and the probability. A decay length and an interaction length are determined independently at random, and the shorter one is taken as the actual path length. By this procedure it is also decided whether there is a particle decay or interaction.

Physics Processes

The Geant4 physics process list contains a variety of complementary and alternative physics processes covering the physics of photons, electrons, muons, hadrons, and ions from 250 eV up to several PeV. Geant4 physics processes provide the electromagnetic interactions of leptons, photons, hadrons, and ions. This class is further sub-divided, as listed below.

- *Standard* – handling process common to all charged particles, including electron, positron, and photon interaction
- *Muons* – handling muon interactions
- *Hadron* - handling charged hadron interactions
- *Low Energy* - alternative models extended down to lower energies

- *X-ray* - Specific codes for X-ray physics
- *Optical photons* - Provides codes for optical photon interactions

Each sub–group is provided with different interaction processes, as shown in Table 3.1. The interaction includes the processes of ionization, bremsstrahlung, multiple scattering, Compton and Rayleigh scattering, photoelectric effect, pair conversion, annihilation, synchrotron and transition radiation, scintillation, refraction, reflection, absorption, and the Cherenkov effect.

Table 3.1: Electromagnetic, hadronic, and other major interactions.

Physics Class	Interactions
Standard	– Multiple Scattering – Transition Radiation – Scintillation – Cerenkov Effect – Photoabsorption
Muons	– Ionization – Bremsstrahlung – Muon Photonuclear – e^- - e^+ Pair Production
Electrons	– Ionization – Bremsstrahlung – e^- - e^+ Annihilation – e^- - e^+ Annihilation into μ^- - μ^+ – e^- - e^+ Annihilation into Hedrons – Synchrotron Radiation
Gamma	– Photoelectric Effect – Compton Scattering – Gamma Conversion into e^- - e^+ – Gamma Conversion into μ^- - μ^+
Hadron	– Ionization – Delta-ray Production
Low Energy	– Compton Scattering – Rayleigh Scattering – Gamma Conversion – Photoelectric Effect – Bremsstrahlung
Optical photons	– Interactions of optical photons

The standard electromagnetic process category provides a variety of implementations of electron, positron, photon and charged hadron interactions. The photon processes include Compton scattering, γ–conversion into electron and muon pairs [34], and the photoelectric effect. Electron and positron processes include bremsstrahlung, ionization, δ–ray production, positron annihilation, and synchrotron radiation. The energy loss process manages the continuous energy loss of particles due to ionization and bremsstrahlung. Multiple scattering processes are included for all charged particles.

Three processes contribute to the energy loss of muons: bremsstrahlung, ionization, and the direct production of electron positron pairs. Each processes also simulates the corresponding discrete δ–ray production process. Furthermore, Geant4 provides the nuclear interaction of muons on the production of hadrons [36]. This is important for the simulation of high energy cosmic rays with high energy muons, muon propagation, and muon-induced hadronic backgrounds at energies above 10 GeV.

The continuous energy loss of charged hadrons is mainly contributed to by ionization, which also stimulates the direct process of δ–ray production. Geant4 includes photonuclear reactions which convert the energy flow of electrons, positrons, and photons into the energy flow of mesons, baryons, and nuclear fragments.

A set of physics processes is implemented in Geant4 for low energy electromagnetic physics of particles, such as electrons, photons, positively and negatively charged hadrons, and positive ions. The low energy processes include the photoelectric effect, Compton scattering, Rayleigh scattering, bremsstrahlung, and ionization. A photon conversion process is also implemented, based on experimental data. In addition, fluorescent emission from excited atoms is also incorporated. The implementation of electron and gamma processes is based on data libraries that provide data for the determination of cross–sections.

There are other physics lists such as, Quark Gluon String Punch-Through (QGSP) for high energy regimes beyond 10 GeV and the Bertini Cascade model (BERT) for hadrons with energies lower than 10 GeV. The Bertini cascade list is implemented with excitation, pre–equilibrium state, nuclear explosion, fission, and evaporation. Additional neutron interactions (donated HP list), and gamma and electro–nuclear interactions (donated as GN list) are also applied.

3.2.3 Event and Tracking Management

The event management is the main unit in simulation. The event management provides an abstract interface to external physics event generators. The physics event generator provides the primary particles that define a physics event. This event class avoids keeping any transient information that is not meaningful after the processing of an event is complete. It contains primary vertices and primary particles before processing the event. After processing, it has hits and digitizations generated by the simulation, and optionally, it can store trajectories of the simulated particles.

In the simulation, each particle is transported step–by–step through the medium. The *G4step* class stores changes in the track properties between the start and end points, including coordinates, volume, energy, and momentum. Another class, called *G4Track*, keeps all information on the final status of the particle after the completion of each step. The tracking scans all physics processes and actions for the given particle, and decides which one is to be invoked. During the physics process characteristics are represented by following three tracking actions:

- *At rest* - particle decay at rest
- *Along step* - energy loss or secondary particle productions
- *Post step* - secondary particle production by a decay or interaction

The main functions of the tracking action are listed below:

- Actions before tracking the particles - clear secondary particles
- Pre–tracking user intervention process
- Construct a trajectory if it is requested
- Inform beginning of tracking to physics process
- Track the particle step-by-step while it is alive
- Post–tracking user intervention process
- Destroy the trajectory if it was created

3.2.4 Digitization and Hit Management

In Geant4, a hit is a snapshot of a physical interaction or an accumulation of interactions of a track or tracks in a sensitive detector component. On the other hand, the term “digit” represents a detector output, for example, an ADC or TDC count, or a trigger signal. A digit is created from one or more hits and/or other digits. Geant4 provides only the abstract classes for both detector sensitivity and hit or digit. A sensitive detector creates hits using the information given in the current step. Hits, which are user-defined objects derived from class G4VHit, are collected in an event object. At tracking time, when the step is inside a volume which has a pointer to a sensitive detector, this sensitive detector is invoked with the current step information. In contrast to a sensitive detector, which is invoked automatically at tracking time, the digitization module must be invoked by the users code. Digitization may be done during event processing, at the end of each event, or even after some number of events have been processed.

3.2.5 Visualization and User Interface

There are three main visualization drivers which have been integrated into Geant4 toolkit, such as OpenGL [37], DAWN [38], and HepRep (High Energy Physics Representable). OPENGL is a standard specification defining a cross-language cross-platform applications program interference for a writing application that produces 3D computer graphics. This is widely used and suited for real–time, fast visualization. DAWN renderers high-quality, precise output visualization which is well suited to preparing technical high-quality postscript output for presentation or documentation. HepRep is suited for visualizing 3D representations of the geometry module and trajectories in the simulation.

Various user interface tools such as Root, Perl, JAVA, etc., can be used to implement the command or GUI based interface for running the simulation code.

In the next chapter, a specific implementation of ECRS simulation is presented for studying cosmic ray shower distribution in the atmosphere.

Chapter 4

ECRS Simulation

Recent studies show that the cosmic ray energy spectrum extends up to $\sim 10^{20}$ eV [39], which is several orders of magnitude higher than the highest energy achieved by man-made accelerators on the Earth. This implies that such energetic particles cannot be detected directly and it is necessary instead to measure the secondary cosmic ray particles produced by the primary cosmic ray interactions in the atmosphere.

The interpretation of EAS measurements is performed by comparing experimental data with model predictions of the shower development in the atmosphere. Cosmic ray shower development is a complex problem involving many aspects: interactions of high energy particles, properties of the atmosphere and the geomagnetic field, etc. A Monte Carlo simulation program is one of the most convenient and effective tools for quantitatively analyzing the transport and interactions of each individual shower particle employing our present knowledge on interactions, decays, and particle transport in matter.

Several simulation programs have been developed for studying the secondary cosmic ray showers, such as, MOCCA [53], ARIES [41], and CROSIKA [42], which are well known simulation programs for cosmic ray studies. However common drawbacks identified with these codes are lack of well defined geomagnetic field and precise atmospheric models for the particle interaction. On the other hand, all the above simulations are based on theoretical physics interaction models (for high energy), which were derived some time ago. Knowledge of high energy interactions is much improved from that of fifteen years ago. The other common problem is that in some of these simulations it is difficult to understand in detail which assumptions have been made in the program, specially for the interaction models.

In order to address the common problems of the existing simulation programs, present knowledge of particle interaction models are incorporated with a geomagnetic field and realistic air layers. For this purpose a Geant4 [43] based simulation program, called ECRS (Earth Cosmic Ray Simulation) has been developed to study secondary cosmic ray particle showers in the full range of Earth's atmosphere. The geometry, magnetic field implementation, and physics processers are described in the following sections.

4.1 Structure of ECRS

The purpose of ECRS is to model secondary cosmic ray shower particles at a user defined location in the Earth's atmosphere. When highly energetic primary cosmic ray particles interact with air nuclei in the upper atmosphere, they create numerous secondary particles, each carrying a fraction of the primary energies. Although some of these secondary particles, mainly muons are high enough in energy to reach the Earth's surface, some of other secondaries interact in a similar manner as the primary, generating new sets of secondaries. The multiplication of these processes continues until the energies of the newly created particles are too low for further particle generation. In an air shower simulation, it is necessary to consider all possible particle interactions and processing to model the secondaries in the atmosphere. Also, it is very important to study the statistical distribution of secondary particles from air shower events. To evaluate such processes, ECRS has an internal monitoring procedure that constantly checks and records a particle's status such as position and energy until it reaches the Earth's surface or another defined area.

Figure 4.1 shows a schematic modular diagram of the ECRS simulation program. The simulation is done with a collection of interacting module. In general, every module can be replaced virtually without altering others. Input Divide Language (IDL) is a set of user defined input parameters which controls ECRS program execution. It mainly includes primary particle type (mainly proton), energy, launching location, direction, and other parameters for the visualization.

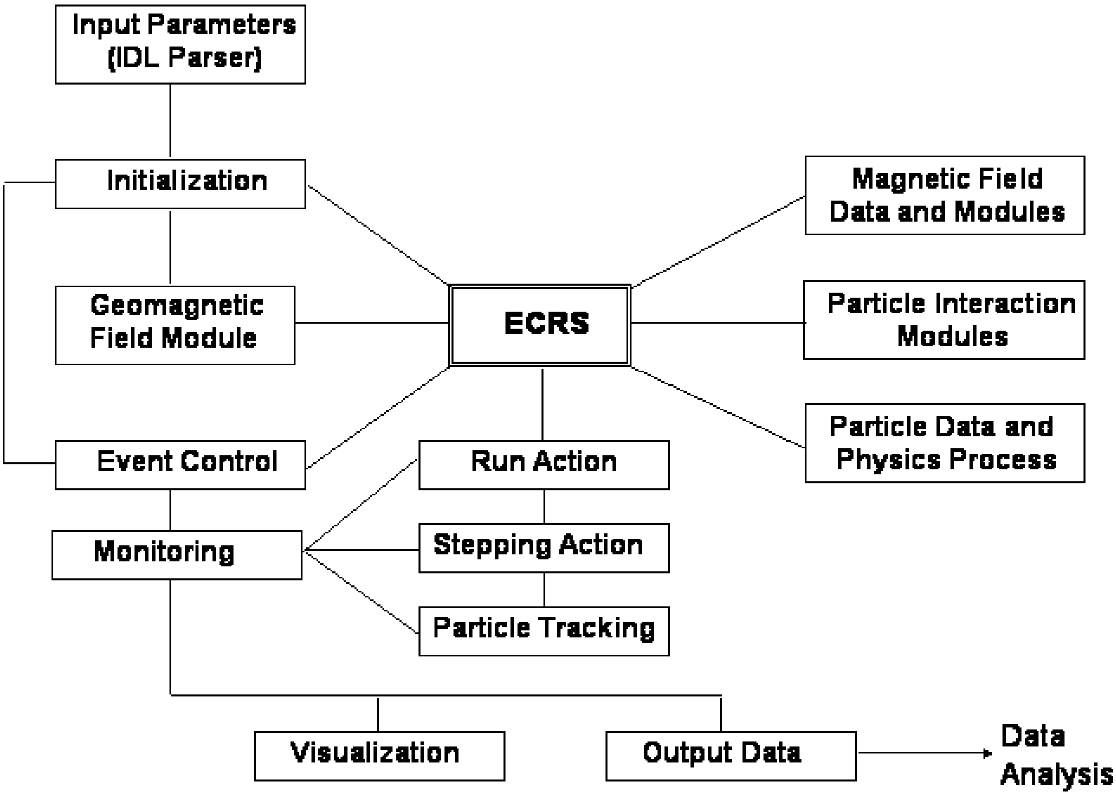

Figure 4.1: The main structure of the ECRS program

According to the instruction of IDL, run action is generating primary particles event by event. It is controlled the flow of the program and managed the event loops within a run. All necessary details of secondary particles are recorded in the event and tracking action processes.

During the tracking and stepping actions, an output data file or visualization is generated by monitoring the routine and control. This output file contains a list of all secondary particles in the atmosphere together with their particle types, energies, positions, and 3-momenta. The ECRS geometry and constrictive methods are discussed in Section 4.2.

4.2 Geometry

In the ECRS program, a real Earth scale (r = 6137 km) has been implemented with a proper set of atmospheric layers. The density, pressure, and air chemical composition are parameterized according to the US standard atmospheric model [44], and discussed in the Section 4.2.2 in detail. The geomagnetic fields are implemented in ECRS in order to correctly simulate the trajectories of charged particles created from cosmic ray particle interactions. The geomagnetic field implementations are discussed in section 4.4.

4.2.1 Coordinate System

The ECRS coordinate system is a Cartesian coordinate system with the origin placed at the center of the Earth. Figure 4.2 shows a schematic representation of the coordinate system of ECRS. The XY plane is located in the equator plane and the positive

Z-axis points upwards along the direction of geographic north.

$O^{'}$ is a user specified geographical location on the Earth's surface above which the initial particle is launched at given altitude. The *shower axis* of a shower with zenith angle θ is defined as the straight line that passes through the intersection points of the $O^{'}$ and O. The azimuth angle ϕ is the angle between the horizontal projection of the shower axis and the X-axis. The $X^{'}Y^{'}$ plane is tangent to the Earth's surface at point $O^{'}$. The Earth radius r_e is taken as 6.3712 x 10^6 m from the Earth's center.

To optimize the simulation time, all the secondary particle positions are recorded based on an XYZ coordinate system. However it is important to convert this coordinate system XYZ to alternative coordinate $X^{'}Y^{'}Z^{'}$ in order to analyze the particle shower distribution at the Earth's surface. This can be done by rotating XYZ around the point $O^{'}$ as shown in the appendix.

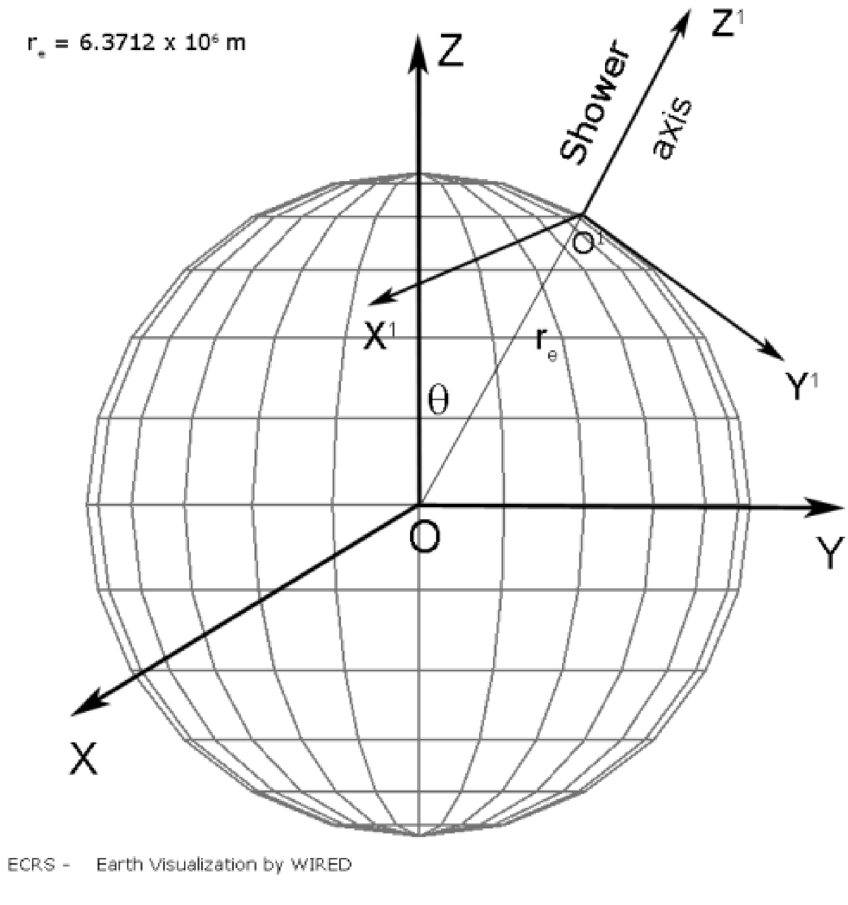
$r_e = 6.3712 \times 10^6$ m
Z
Z1
Shower
axis
O1
X1
r_e
Y1
θ
O
Y
X
ECRS - Earth Visualization by WIRED

Figure 4.2: ECRS coordinate system

4.2.2 Atmospheric Model

Earth's atmosphere is a layer of air gases surrounding the Earth and retained by the Earth's gravity. In the air shower simulation, particle propagation strongly depends on the atmospheric characteristics. There are two main atmospheric characteristics that have been considered in the ECRS simulation: air density and chemical composition.

The Earth's atmosphere has been extensively measured and studied during the last few decades. A variety of models and parameterizations of measured data have been published. Among them, the so-called US standard atmosphere is a widely used model based on experimental data. This standard data was used to set up atmospheric layers in the ECRS simulation.

The atmospheric medium is inhomogeneous and atmospheric air density plotted as a function of altitude is shown in Fig.4.3. The density of the atmosphere diminishes six orders of magnitude as the altitude increases from 0 to 100 km, and another additional six orders of magnitude for the range from 100 to 300 km. The chemical composition of the atmosphere plotted as a function of altitude is shown in Fig. 4.4. The mean molecular weight for the chemical composition of air remains constant below 90 km in altitude, and then diminishes for higher altitudes. This constant value (28.9) is the mean molecular weight for a mixture of air gases such as Oxygen, Nitrogen, Argon, and trace elements.

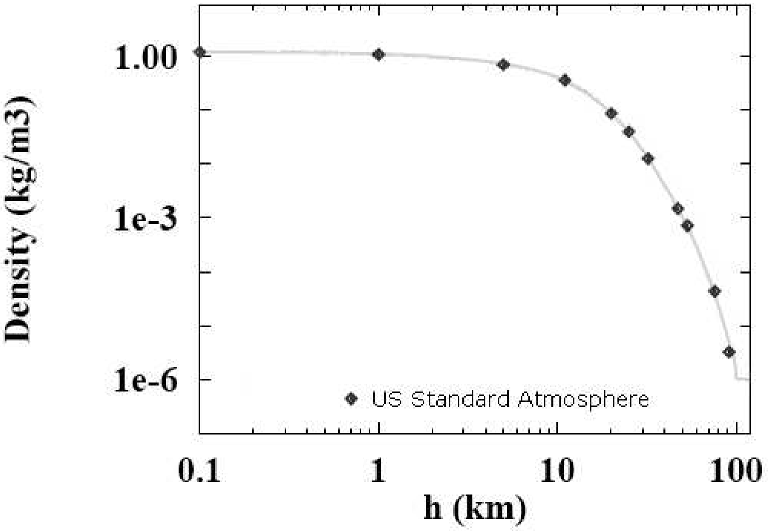

Figure 4.3: Atmospheric density as a function of altitude.

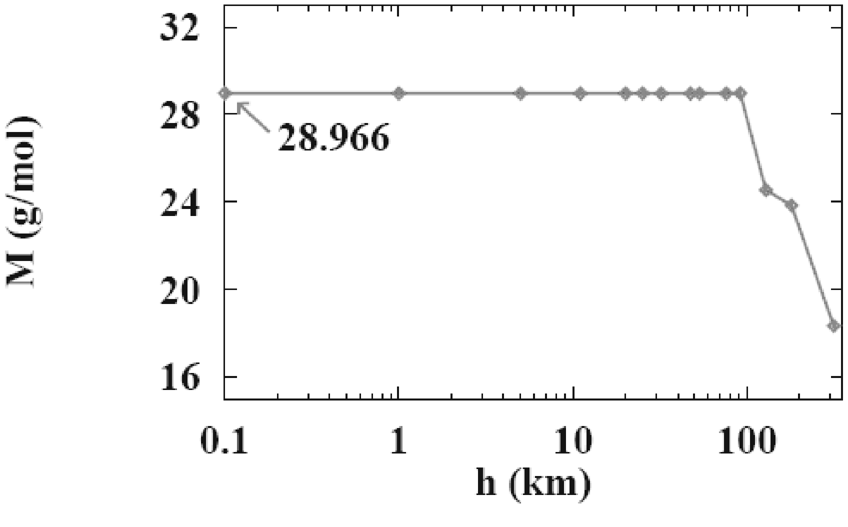

Figure 4.4: Atmospheric mean molecular weight as a function of altitude.

There are 37 layers set up following the US standard atmospheric model for the vertical altitude region from 0 to 100 km. Beyond that altitude, it has a vacuum layer up to the vertical altitude height at 5000 km, which is set as the boundary of the simulation. Table 4.1 shows the key parameters: air layer boundaries, density with corresponding temperature used for the simulation. The chemical composition of atmospheric air has been considered as a mixture of 75.521% N, 23.143% O, 1.288% Ar, 0.048% CO_2 and trace elements.

Table 4.1: Atmospheric air density and temperature model configuration in ECRS

Layer	Layer Thickness (km)	Density (kg/m^3)	Temperature (kelvin)
1	0 - 1	1.16	284.9
2	1 - 2	1.05	278.4
3	2 - 3	9.56 x 10^{-1}	271.90
4	3 - 4	8.63 x 10^{-1}	265.40
5	4 - 5	7.77 x 10^{-1}	258.90
6	5 - 6	6.97 x 10^{-1}	252.41
7	6 - 7	6.64 x 10^{-1}	245.92
8	7 - 8	5.57 x 10^{-1}	239.45
9	8 - 9	4.95 x 10^{-1}	232.97
10	9 - 10	4.39 x 10^{-1}	226.94
11	10 - 11	3.88 x 10^{-1}	220.01
12	11 - 12	3.37 x 10^{-1}	216.65
13	12 - 13	2.88 x 10^{-1}	216.65
14	13 - 14	2.46 x 10^{-1}	216.65
15	14 - 15	2.10 x 10^{-1}	216.65
16	15 - 16	1.66 x 10^{-1}	216.65
17	16 - 17	1.42 x 10^{-1}	216.65
18	17 - 18	1.21 x 10^{-1}	216.65
19	18 - 19	1.04 x 10^{-1}	216.65
20	19 - 20	8.89 x 10^{-2}	216.65
21	20 - 21	7.57 x 10^{-2}	217.58
22	21 - 22	6.45 x 10^{-2}	218.57
23	22 - 23	5.50 x 10^{-2}	219.56
24	23 - 24	4.69 x 10^{-2}	220.55
25	24 - 25	4.00 x 10^{-2}	221.55
26	25 - 26	3.42 x 10^{-2}	222.53
27	26 - 27	2.92 x 10^{-2}	223.52
28	27 - 28	2.50 x 10^{-2}	224.51
29	28 - 29	2.14 x 10^{-2}	225.50
30	29 - 30	1.84 x 10^{-2}	226.50
31	30 - 40	8.46 x 10^{-3}	236.51
32	40 - 50	1.71 x 10^{-3}	266.92
33	50 - 60	5.04 x 10^{-3}	258.01
34	60 - 70	1.63 x 10^{-3}	233.29
35	70 - 80	3.99 x 10^{-3}	208.39
36	80 - 90	8.21 x 10^{-4}	188.89
37	90 - 100	1.39 x 10^{-4}	188.42

4.3 Primary Particle Event Generation

The primary particle event is generated by using an interface of an event generator or the particle gun class, which creates a beam of particles by defining their type, position, direction of motion and kinetic energy. The primary event energy and incident direction are defined as a fixed value or selected at random within a given energy range. The two different geomagnetic field modes are introduced to enabled and disabled the magnetic field in order to study the magnetic field effects.

In particularly, the secondary particle energy cut (threshold value) is introduced for gamma, neutron and electron particles. This is done as to avoid the production of a large number of low energy secondary particles, which would deteriorate the performance of the simulation without enhancing the accuracy of the calculations. In the present ECRS simulation, the production threshold value for gamma, neuron, and electron particles is 250 keV.

All major physics processes, which were described in 3.2.2 are incorporated for more than 50 elementary particles: γ, $\mathrm{e}^{\pm}$, $\mu^{\pm}$, π^{0}, $\pi^{\pm}$, $\mathrm{K}^{\pm}$, K^{0}, η, the baryons p, n, K^{*0}, $\mathrm{K}^{*\pm}$, $\Delta^{\pm}$, Δ^{0}, Δ^{++}, and the corresponding anti–baryons, ν_e, ν_μ, and corresponding anti–neutrinos. Sets of extra hadronic interactions are used for the energies above 10^{12} eV. These interactions processes, QGSP, QGSP BERT, LHEP BERT and LHEP GN are predominated from $p\bar{p}$ colliders and extrapolated to higher energies.

4.4 Geomagnetic Field

The origin of the Earth's magnetic field is not completely understood. However, it has been modeled with a combination of internal and external magnetic fields. Both the internal and the external magnetic fields have been implemented in the ECRS simulation. When simulating the propagation of cosmic rays in the Earth's magnetosphere, the transformation of the vector position and direction of the particles from one coordinate system to anther is an important issue. Three additional parameters are used to define particle position: altitude, longitude, and latitude, which define the geographic coordinate system. Conversion to this coordinate system is presented in Appendix.

The following sections cover the details of internal and external magnetic field implementation, along with geomagnetic field validation studies and primary results.

4.4.1 Internal Magnetic Field

The first suggestion that the Earth was like a giant bar magnet was by William Gilbert (1544 – 1603), physician to Queen Elizabeth I. The Earth's internal magnetic field is created by electric currents generated by the rotating iron/nickel core. Its magnetic moment is 6.4 x 10^{21}Am^2. It can be represented to a first approximation by a magnetic dipole. The actual magnetic field lines are distorted from the symmetrical dipole shape by the solar wind, which is discussed in Section 4.4.2. The typical magnetic field lines run parallel to the surface of the Earth at the equator, and normal to the Earth's surface at the magnetic poles. The average magnetic field strength is 0.5 gauss, which is several hundred times weaker than the field around a bar magnet.

A precise model of the internal geomagnetic field is given by the Internal Geomagnetic Reference Field (IGRF) [46] model. In this model the geomagnetic field is considered to be current free outside the Earth. This can be defined by the gradient of the scalar potential given by,

$$V(r,\theta,\phi) = \sum_{n=1}^{n_{max}} \left(\frac{R_e}{r}\right)^{n+1} \sum_{m=0}^{n} (g_n^m \cos m\phi + h_n^m \sin m\phi) P_n^m(\cos\theta) \tag{4.1}$$

where R$_e$ is the Earth radius, r, ϕ, and θ are the spherical geographic coordinates, P$_n^m$ is the Schmidt normalized Legendre polynomial of degree n and order m and, g$_n^m$ and h$_n^m$ are the Gauss spherical harmonic coefficients. The Gauss coefficients are derived from magnetic field measurements made by geomagnetic stations, ship–towed magnetometers, and satellites. The International Association of Geomagnetism and Astronomy (IAGA) issues a new set of Gauss coefficients defining the new IGRF model for a particular epoch (usually a 5 year period). When special retrospective studies are completed, the IGRF is corrected or modified and so-called as a Definitive Geomagnetic Reference Field (DGRF). Table 4.3 shows an example of the Gauss coefficients, up to the degree 8, computed for the DGRF of IGRF field model from 1900 to 2000. The last column in the table gives the rate of annual change (secular variation (sv)) of the coefficients projected from the last five year model to the year 2005. The listed IGRF values are used to model the internal Earth's magnetic field in the ECRS program.

The Earth–centered geomagnetic dipole field is obtained by equation 4.1 to terms of degree 1 (n =1). It defines the Earth's centered geomagnetic dipole magnetic dipole field field that has an axis titled with respect to the Earth's rotation axis. The moment B$_0$ of the geomagnetic field dipole is given by,

$$B_0 = \sqrt{(g_1^0)^2 + (g_1^1)^2 + (h_1^1)^2} \tag{4.2}$$

The geomagnetic spherical coordinates θ_{dip} and ϕ_{dip} of the geomagnetic dipole axis are defined by

$$\cos\theta_{dip} = \frac{g_1^0}{B_0} \quad , \qquad \tan\phi_{dip} = \frac{h_1^1}{g_1^1}. \tag{4.3}$$

Figure 4.5 shows the ECRS internal magnetic field lines near the Earth's surface before implementation of the external magnetic field. The magnetic field lines are symmetric around the geomagnetic axis, which is inclined 11.3° from the Earth's axis of rotation. This angle of inclination was calculated directly from the IGRF/DGRF Gauss coefficients.

Table 4.2: Definitive and International Geomagnetic Reference Field Values

			DGRF	DGRF	DGRF	DGRF	DGRF	DGRF	sv
g/h	n	m	1900	1920	1940	1960	1980	2000	nT/yr
g	1	0	-31543	-31060	-30654	30421	-299992	-29615	14.6
g	1	1	-2298	-2317	-2292	-2169	-1956	-1728	10.7
h	1	1	5922	5845	5821	5791	5604	5186.1	-21.2
g	2	0	-677	-839	-1106	-1555	-1997	-2267.7	-14.4
g	2	1	2905	2959	2981	3002	3027	3044	-3.7
h	2	1	-1061	-1259	-1614	-1967	-2129	-2481.6	-22.7
g	2	2	924	1407	1566	1590	1663	1670.9	-3.6
h	2	2	1121	823	528	206	-200	-458	-11.1
g	3	0	1022	1111	1240	1302	1281	1339.6	-1.1
g	3	1	-1469	-1600	-1790	-1992	-2180	-2288	-3.5
h	3	1	-330	-445	-499	-414	-336	-227.6	5.6
g	3	2	1256	1205	1232	1289	1251	1252.1	-1.2
h	3	2	3	103	163	224	271	293.4	-4.5
g	3	3	572	839	916	878	833	714.5	-8.5
h	3	3	523	293	43	-130	-252	-491.1	-8
g	4	0	876	889	914	957	938	932.3	-2.7
g	4	1	628	695	762	800	782	786.8	2.2
h	4	1	195	220	169	135	212	272.6	1.7
g	4	2	660	616	550	504	398	250	-8
h	4	2	-69	-134	-252	-276	-269	-231.9	1
g	4	3	-361	-424	-405	-394	-419	-403	4.5
h	4	3	-210	-153	-72	3	53	119.8	5.1
g	4	4	134	199	265	269	199	111.3	-1.9
h	4	4	-75	-57	-141	-255	-297	-303.8	-0.3
g	5	0	-184	-221	-241	-222	-218	-218.8	-1.4
g	5	1	328	326	334	362	357	351.4	0.7
h	5	1	-210	-122	-33	16	46	43.8	-0.3
g	5	2	264	236	208	242	261	222.3	-2.6
h	5	2	53	58	71	125	150	171.9	1.5
g	5	3	5	-23	-33	-26	-74	-130.4	-1.2
h	5	3	-33	-38	-75	-117	-151	-133.1	2
g	5	4	-86	-119	-141	-156	-162	-168.6	0
h	5	4	-124	-125	-113	-114	-78	-39.3	3.8
g	5	5	-16	-62	-76	-63	-48	-12.9	-0.2
h	5	5	3 43	69	81	92	106.3	107	-0.5
g	6	0	63	61	57	46	48	72.3	0.4
g	6	1	61	55	54	58	66	68.2	0.3
h	6	1	-9	0	4	-10	-15	-17.4	-0.7
g	6	2	-11	-10	-7	1	8	74.2	0.7
h	6	2	83	96	105	99	93	63.7	-1.8
g	6	3	-217	-233	-249	-237	-192	-160.9	1.9
h	6	3	2	11	33	60	71	65.1	-0.2
g	6	4	-58	-46	-18	-1	4	-5.9	-1.7

Table 4.3: Definitive and International Geomagnetic Reference Field Values cont.

			DGRF	DGRF	DGRF	DGRF	DGRF	DGRF	sv
g/h	n	m	1900	1920	1940	1960	1980	2000	nT/yr
h	6	4	-35	-22	-15	-20	-43	-61.2	-0.4
g	6	5	59	44	18	-2	14	16.9	-0.5
h	6	5	36	18	0	-11	-2	0.7	-0.2
g	6	6	-90	-101	-107	-113	-108	-90.4	0.7
h	6	6	-69	-57	-33	-17	17	43.8	1.5
g	7	0	70	73	74	67	72	79	0.2
g	7	1	-55	-54	-53	-56	-59	-74	-0.1
h	7	1	-45	-49	-52	-55	-82	-64.6	0.7
g	7	2	0	2	4	5	2	0	-0.3
h	7	2	-13	-14	-18	-28	-27	-24.2	0.3
g	7	3	34	29	20	15	21	33.3	1.1
h	7	3	-10	-13	-14	-6	-5	6.2	0.1
g	7	4	-41	-37	-31	-32	-12	9.1	0.7
h	7	4	-1	4	7	7	16	24	0.3
g	7	5	-21	-16	-9	-7	1	4	0.5
h	7	5	28	28	29	23	18	14.8	-0.8
g	7	6	18	19	17	17	11	7.3	-0.3
h	7	6	-12	-16	-20	-18	-23	-25.4	-0.1
g	7	7	6	6	5	8	-2	-1.2	0.5
h	7	7	-22	-22	-19	-17	-10	-5.8	0.2
g	8	0	11	11	11	15	18	24.4	0.1
g	8	1	8	7	7	6	6	6.6	0.2
h	8	1	8	8	8	11	7	11.9	-0.2
g	8	2	-4	-3	-3	-4	0	-9.2	-0.5
h	8	2	-14	-15	-14	-14	-18	-21.5	0.1
g	8	3	-9	-9	-10	-11	-11	-7.9	0.2
h	8	3	7	6	5	7	4	8.5	0.3
g	8	4	1	2	1	2	-7	-16.6	-0.4
h	8	4	-13	-14	-15	-18	-22	-21.5	0.4
g	8	5	2	4	6	10	4	9.1	0.2
h	8	5	5	5	5	4	4	15.5	0.1
g	8	6	-9	-7	-5	-5	3	7	0.5
h	8	6	16	17	19	23	16	8.9	-0.3
g	8	7	5	6	9	10	6	-7.9	-0.7
h	8	7	-5	-5	-4	-10	-16	-14.9	0.4
g	8	8	8	8	7	8	4	-4	0.4
h	8	8	-18	-19	-19	-20	-15	-2.1	0.4

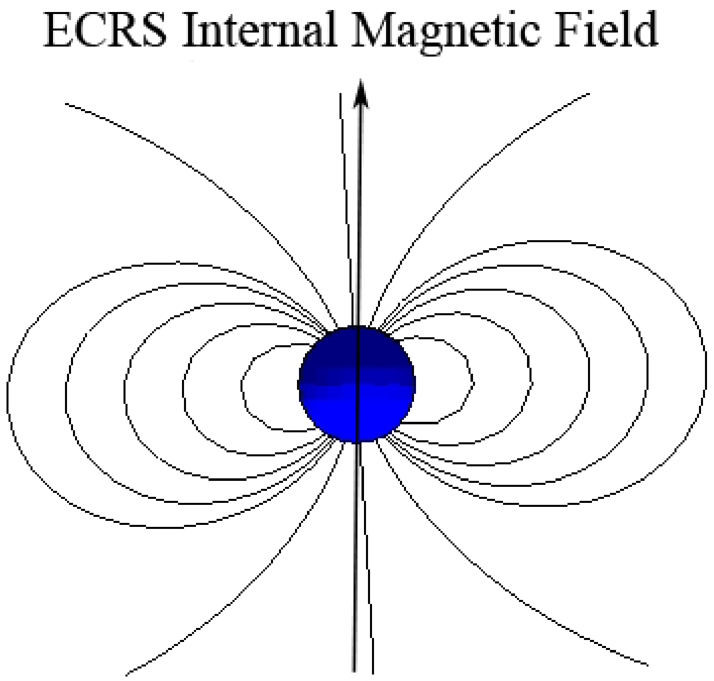

Figure 4.5: ECRS internal magnetic field lines around the Earth. This figure shows the first magnetic field implementation using the dipole approximation. The geomagnetic axis is inclined 11.3° from the Earth's axis of rotation.

4.4.2 External Magnetic Field

Beyond about three Earth radii, the Earth's magnetic field is increasingly affected by the solar wind interaction with the Earth's magnetosphere, and in this region is called external magnetic field. This distortion of the magnetic field is caused by several external sources in the magnetospheric current systems; the Chapman-Ferraro current on the magnetopause, the tail current sheet, and the field aligned Birkeland current systems I and II [47]. There are several models that have been developed for the external geomagnetic field by incorporating with above sources. The Tsyganenko model is one of the most established models, which has been developing since the mid 1980's. The latest version of the Tsyganenko2001 [48] model is used in the ECRS simulation. The next paragraphs provide a brief description of the Tsyganenko model.

The Tsyganenko model is a semi-empirical best-fit representation of the magnetic field, based on a large number of satellite observations such as IMP, HEOS, ISEE, POLAR, and Geotail. In this model the external magnetic field is influenced by the geomagnetic field that is considered to be a geomagnetic dipole. The Tsyganenko98 model is dipole tilt dependent and was primarily developed as a tail current model. It also provides seven different states of the magnetosphere corresponding to different level of geomagnetic activity [49]. However, the Tsyganenko89 model doesn't provide modeling of the continuous variation of the structure of the magnetosphere as a function of the Disturbance Storm Time (Dst) index and of the solar wind parameters. Such modeling is important when considering the changes in the magnetosphere during a magnetic storm or the change in the composition of the magnetosphere due to a change in the solar wind parameters.

The Tsyganenko96 model has explicitly introduced these two absence dependencies [50, 51]. In this model the external magnetic field is produced by a Birkeland current system depending on the dipole tilt angle *PS*, solar wind dynamic pressure P_{dyn}, *Dst*, and the components of the interplanetary magnetic field (IMF), B_y and B_z. The solar dynamic pressure is given by,

$$P_{dyn} = nV^2, \tag{4.4}$$

where n and V represent the solar wind density and velocity, respectively. The contributions of the ring current and magnetosheet currents are also confined into a specific model of the magnetopause. The magnetopause is represented by a semi–ellipsoid in the front, continued in the far tail by a cylindrical surface. The size of the magnetopause decreases, when the P_{dyn} index increases. The strength of the ring current is a function of the *Dst*, with a correction depending on P_{dyn}, to take into account the contribution of the magnetopause current on *Dst*. The amplitude of the magnetosheet current depends on P_{dyn}, B_y, and B_z. Both the shape of the ring current and of the magnetosheet current depend on the dipole tilt angle *PS*.

The Tsyganenko2001 model is not only based on both Tsyganenko89 and Tsyganenko96, but also on two additional parameters: *G1* and *G2*. The *G1* parameter was derived to quantify the energy transfer from the solar wind to the magnetosphere over the last hour. It is a function of the solar wind velocity *V*, given in km/s, of the IMF transverse component B$_\perp = \sqrt{B_y^2 + B_z^2}$, and the IMF clock angle tan $(\theta) = (\frac{B_y}{B_z})$,

$$G_1 = \sum_{i=1}^{12} V_i h(B_\perp) \sin^3 \frac{\theta_i}{2} \tag{4.5}$$

where the different value of i represent 5 min average values that cover the last hour, and

$$h(B_{\perp}) = \frac{(B_{\perp}/4)^2}{1 + B_{\perp}/4} \tag{4.6}$$

The Parameter G$_2$ quantifies the strength of the sunward convection electric field over the last hour of observation, and controls the tailward shift of the magnetotail current system. It is defined by the function

$$G_2 = a \sum_{i=1}^{12} V_i B_s, \tag{4.7}$$

where B$_s$ represents the southward component of IMF in nT, and it is selected as B$_s$ = |B$_z$| for B$_z$ < 0 and B$_s$ = 0 for B$_z$ > 0. The constant a = 0.005 was introduced to keep the parameter G$_2$ within the range 0 $\leq$ G$_2$ $\leq$ 10 for commonly observed solar wind parameter values.

The other advantage of this model is that the Birkeland currents vary in response to the interplanetary conditions. The magnetopause is specified by an empirical model with the size varying with the solar wind pressure as in the Tsyganenko96 model. Figure 4.6 shows both internal and external magnetic field lines around the Earth's surface in the ECRS. Both IGRF and Tsyganenko2001 models are taken into account in order to correctly set up the geomagnetic field lines.

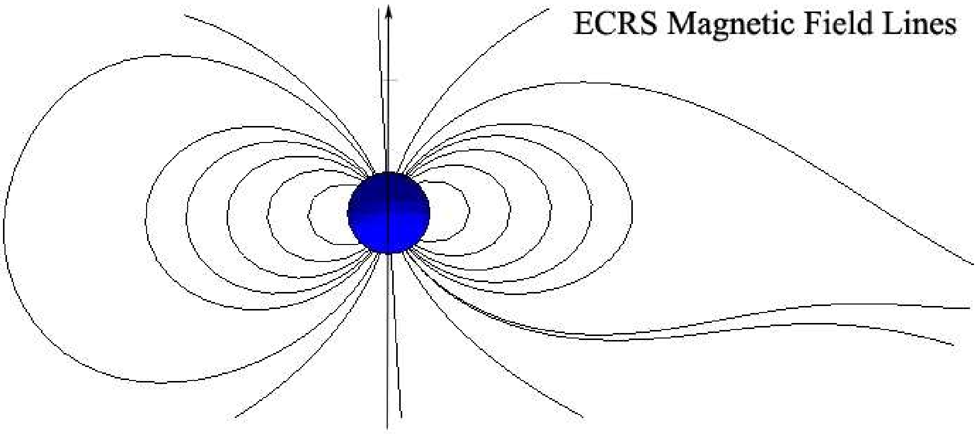

Figure 4.6: Current ECRS magnetic field model incorporating with both internal and external magnetic fields. The symmetric internal magnetic field lines of Figure 4.5 are distorted by the solar wind.

4.5 Magnetic Field Effects on Charged Particles

4.5.1 Charged Particle Rigidity

The motion of charged particles through a magnetic field is described by the non-relativistic Lorentz equation of motion

$$\frac{d\vec{p}}{dt} = q\vec{v} \times \vec{B} \tag{4.8}$$

where q, $\vec{p}$, $\vec{v}$, and $\vec{B}$ represent the particle momentum, charge, velocity, and the magnetic field respectively. This equation of motion conserves p, the magnitude of the momentum, and therefore the energy of the particle. Equation 4.8 can also be written as follows

$$\frac{d\vec{I_v}}{ds} = \frac{q}{p}\vec{I_v} \times \vec{B}, \tag{4.9}$$

where $\vec{I_v}$ represents the velocity direction and s is the path length along the particle trajectory.

The charged particle rigidity is defined as $\frac{pc}{q}$, where c is the speed of light. Equation 4.9 shows that for the same initial position and direction, charged particles with the same rigidity and charge have identical trajectories. For this reason it is more convenient to characterize the trajectories of cosmic rays as a function of their rigidity rather than their energies.

4.5.2 Tracking Charged Particles Through Magnetic Field

ECRS uses numerical methods for computing the trajectory of charged particles through a magnetic field. For this purpose, the Lorentz equation of motion (Equation 4.8) is integrated numerically.

In the Geant4 toolkit, the trajectory of a particle is divided into tracking steps. For nonlinear motion through a magnetic field, these trajectory steps are divided into smaller steps (called chords). The equation of motion is integrated over these chords. After the integration of a chord, the program checks if the particle has crossed a boundary between two different regions of the geometry. If this is the case, the intersection with a boundary is determined and the tracking is stopped at this boundary.

For the numerical integration, the chord is divided into small integration steps. The motion of the particle over a small integration step is done by a G4Stepper object. Several types of G4Steppers are available corresponding to different integration algorithms. In ECRS the 4^{th} order Runge Kutta method is used as the integration algorithms. After each integration step an estimate of accepted relative error ξ is computed. If this error is higher than the maximum accepted relative error (100 μm), a smaller integration step is chosen and the integration restarts from the previous step. When the relative error is significantly smaller than ξ, the step size is increased. This process continues until this particle reaches to a boundary.

4.6 Visualization

Visualization is one of the important modules in the ECRS simulation. ECRS visualization is designed to display Earth's geometry, magnetic field lines, particle trajectories, tracking steps, hits etc. This is critical for code debugging and testing during the initial simulation setup.

There are three visualization drivers used in ECRS: OpenGL [37], DAWN [38], and

HepRep.

Figure. 4.7 shows a snap shot of ECRS run time visualization of a cosmic ray shower near the Earth's surface from a 100 GeV primary proton particle launched toward the Earth's surface. The blue and red color trajectories represent positive and negative particles respectively. All neutral and gamma particle trajectories are not shown in the figure in order to view a clear trajectories of the charge particle. The curved trajectories are due to the magnetic field effect.

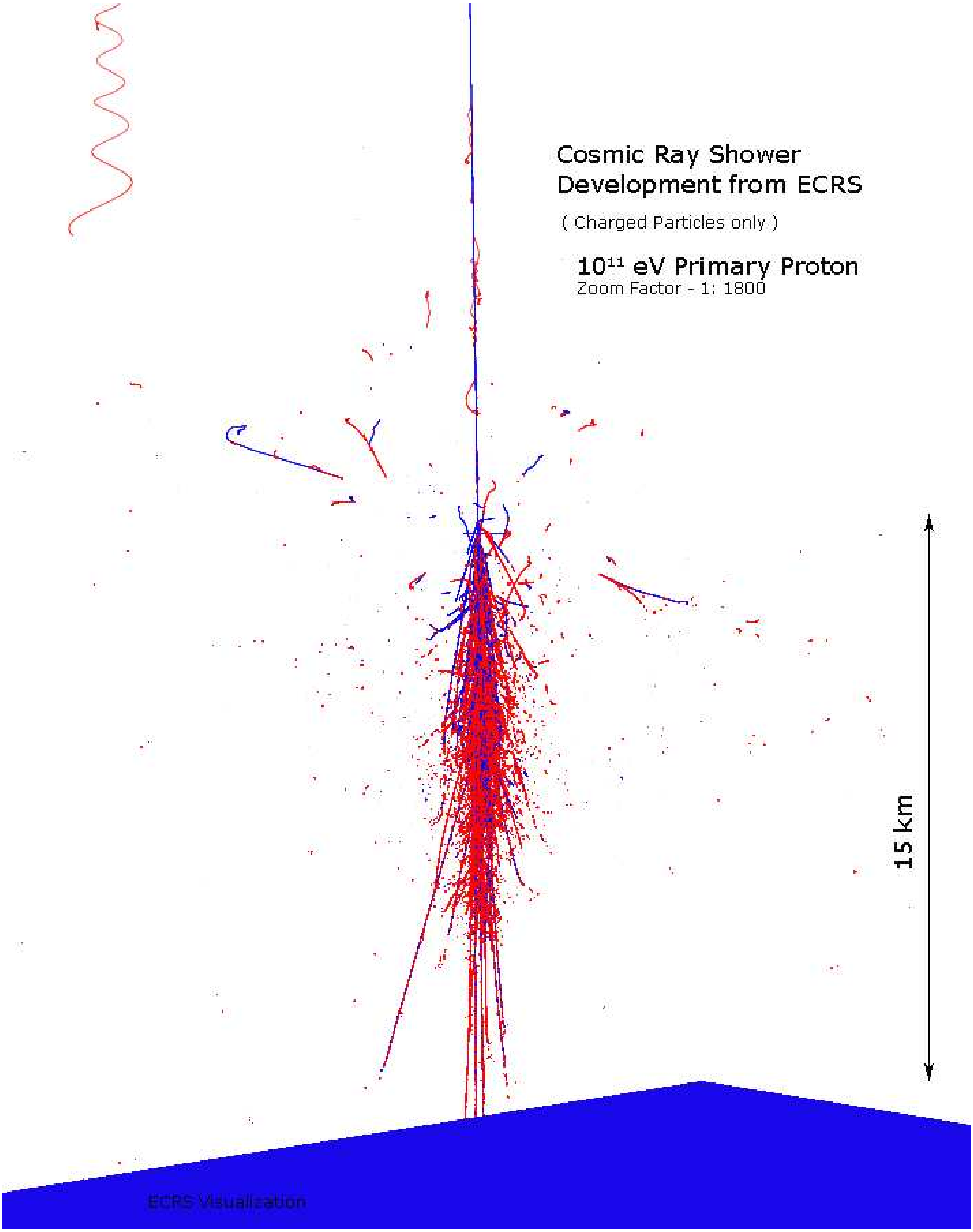

Figure 4.7: Enlarged visualization of the secondary charged particle shower near the Earth's surface from a 100 GeV primary proton event (Scale 1:1800). The red and blue trajectories represent negative and positive particles, respectively. All neutral and gamma particles are not shown in the figure in order to give a clear view of secondary charged particle trajectories. The blue area at the bottom is the surface of the Earth.

4.7 Magnetic Field Consistency Test

A reliable interpretation of ECRS results requires a full knowledge of the geomagnetic field effects implemented in the ECRS simulation. In order to validate the geomagnetic field effects, two tests were performed with the ECRS. In each case, the ECRS simulation result is compared with existing theoretical model calculation.

The first consistency test is check a set of proton particle trajectories with initial energy of 36.44 GeV, initiated in the geomagnetic equatorial plane with different impact parameters. For this analysis, particle decay and interaction channels are turned off in order to follow the full length of the proton trajectories without creating the secondaries. The ECRS result clearly shows the magnetic field effects near the Earth's surface as shown in Fig.4.8 which is very consistent with the Haillas [53] theoretical model calculation.

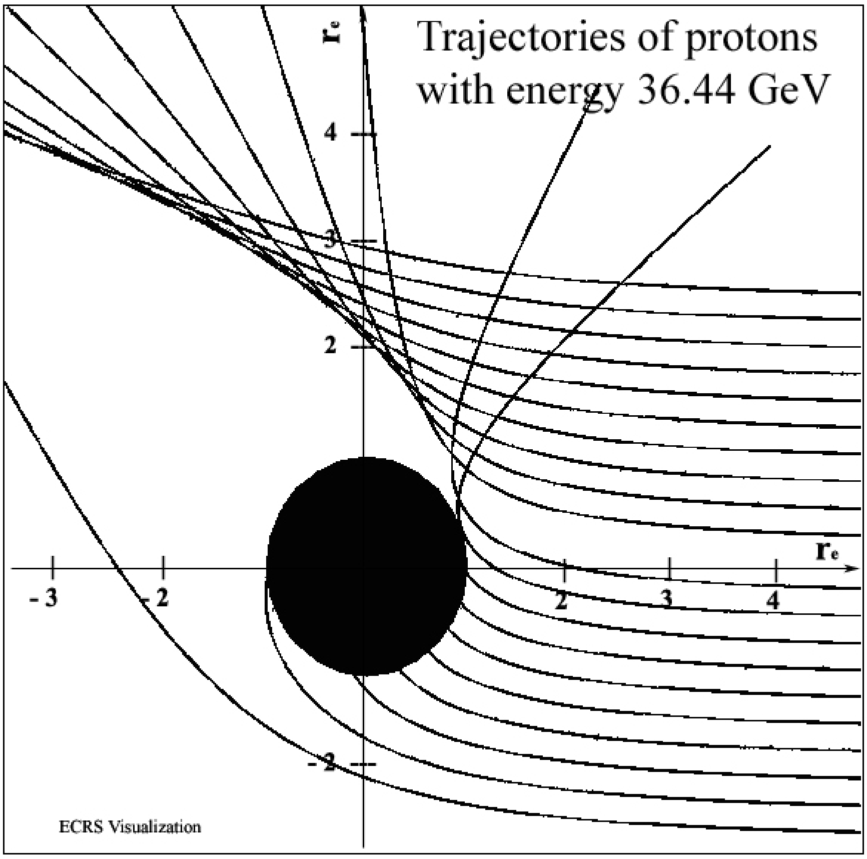

Figure 4.8: Trajectories of 36.44 GeV protons injected in the geomagnetic equatorial plane, with different impact parameters. The Earth is represented by the black sphere. The solid lines represent proton trajectories at different impact parameters.

The second test is to compute the vertical cutoff rigidities of the primary protons at selected locations on the surface of the Earth. The geomagnetic cutoff rigidity is defined as the lowest energy of a particle that can reach the surface of the Earth in vertical direction.

The backward trajectory method is used to compute the cutoff rigidity. In this method, proton particles with different energies are launched on the Earth's surface in upward direction into the air. The magnetic cutoff rigidity is determined from a given proton energy below which the protons will be trapped inside the magnetic field and returned back to the surface of the Earth. In other words, a charged particle with magnetic cutoff rigidity below this value will not reach to this particular location on the Earth from air.

Figures 4.9 and 4.10 show the trajectories of protons with different rigidities for two different locations. In Fig.4.9, the black, blue, red, and green lines represent proton trajectories with rigidities of 20.00, 15.00, 9.95, and 9.90 GV, respectively. Particles at high rigidity, which are the black and blue lines, have small trajectories bending and travel away from the Earth's surface. The particle with 9.90 GV (red line) has significant bending and return back to another point on the Earth's surface. The particle with the lowest energy that can escape the Earth's surface is 9.95 GV, which is represented by the green trajectory. This particle trajectory has several complex loops before escaping the Earth's surface indicating that below this specific rigidity, cosmic ray particles can't reach the selected position from the vertical direction. This value, 9.95 GV is the computed cutoff rigidity for the location at longitude 30^0 and latitude 30^0.

A similar analysis is shown in Fig. 4.10 for a location at longitude 30^0 and latitude 55^0 where the computed cutoff rigidity is determined to be 1.55 GV. Table 4.4 shows the

ECRS results compared with analytical calculations done by Shea and Smart [54]. The values computed in the ECRS and the analytical cutoff rigidities are within ± 0.06 GV of each other. These two consistency studies show that the geomagnetic field implementation is properly done in ECRS.

The extensive air shower simulation results from the ECRS simulation are presented in next two chapters.

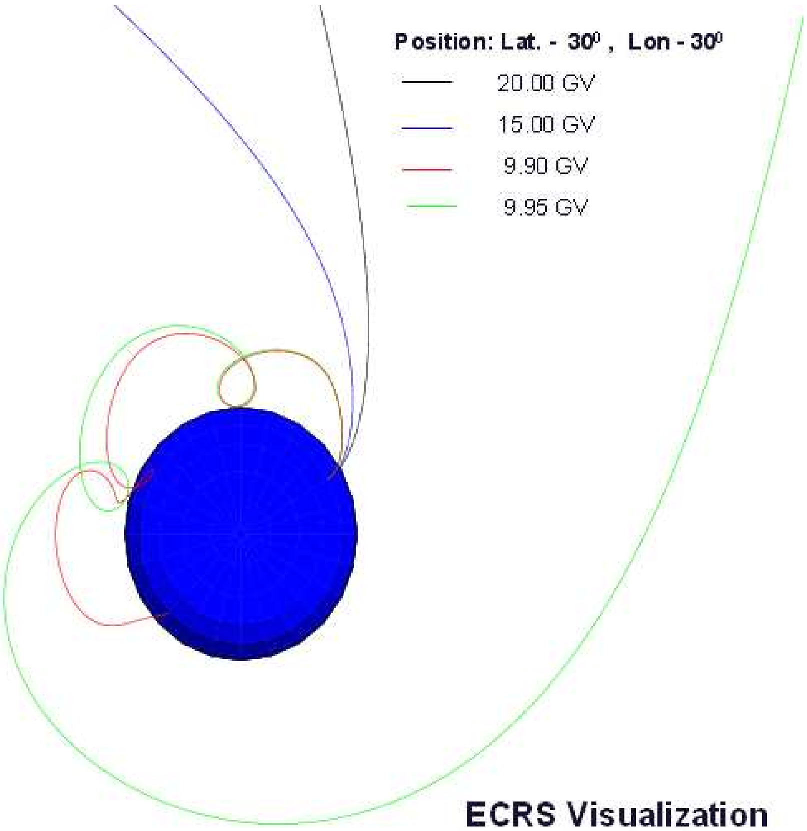

Figure 4.9: Computed trajectories from ECRS with a given geomagnetic latitude of 30^0 and longitude 30^0. The Earth is represented by the blue sphere and the black, blue, red, and green lines represent, computed backward trajectories of protons with rigidities of 20.00, 15.00, 9.95, and 9.90 GV respectively.

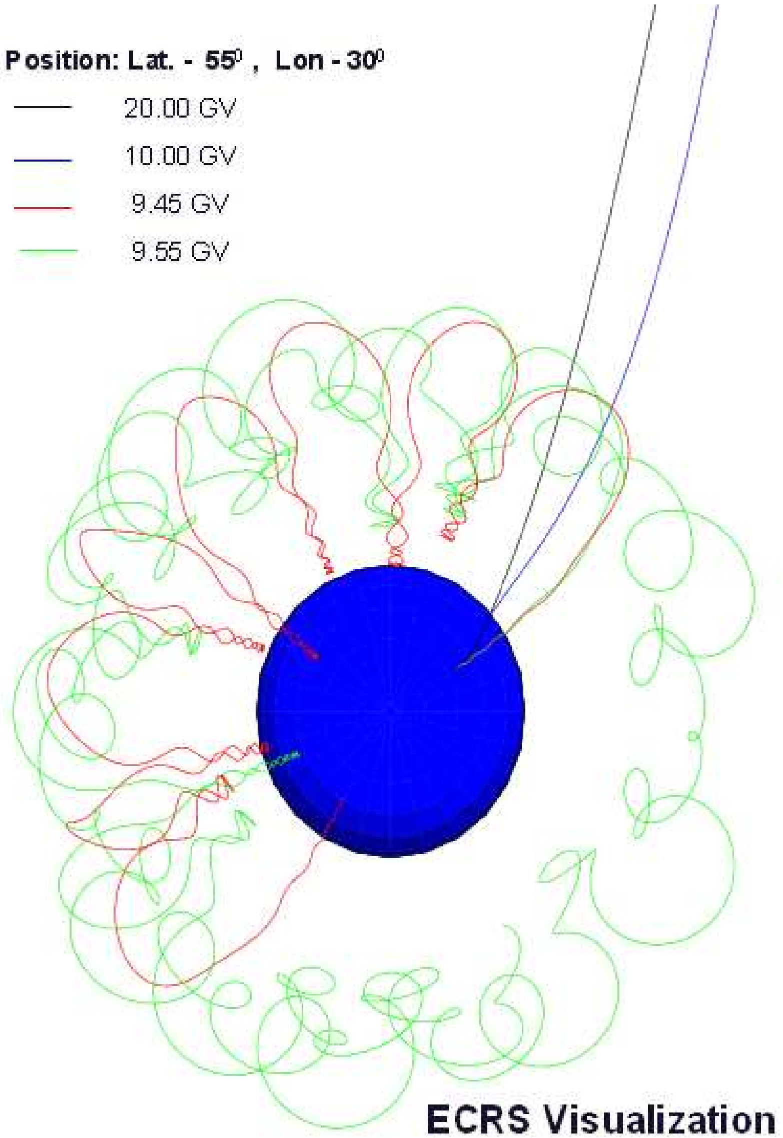

Figure 4.10: Computed trajectories from ECRS with a given geomagnetic latitude of 55^0 and longitude 30^0. The Earth is represented by the blue sphere and the black, blue, red, and green lines represent, computed backward trajectories of protons with rigidities of 20.00, 10.00, 1.55, and 1.45 GV respectively.

Table 4.4: ECRS computed cutoff rigidity results compared with Shea and Smart's analytical calculations for selected locations.

Latitude(N)	Longitude(E)	ECRS	Analytical	Location
		Computed (GV)	(GV)	
55	30	2.35	2.30	Europe
50	15	3.56	3.52	Europe
40	15	7.25	7.22	Europe
45	285	1.49	1.45	North America
40	255	2.57	2.55	North America
20	300	10.09	10.01	North America
50	105	4.31	4.25	Asia
40	120	9.30	9.25	Asia
-25	150	8.59	8.56	Australia
-35	15	4.45	4.40	South Africa

Chapter 5

ECRS Air Shower Development

The Earth's climate mainly depends on how radiation from outer space is absorbed and redistributed by the atmosphere. Any variation in the energy received at the Earth's surface will, therefore, have an immediate effect on Earth's climate. Cosmic rays are one of the main radiation sources that may influence Earth's climate, and the energy balance of the Earth [55]. Therefore, it is important to study the variation of cosmic rays in order understand the Earth's climate changes.

Recent studies concluded that cloud cover and cosmic ray flux are correlated [56, 57], and suggest that solar variability may be linked to climate changes through the solar wind and clouds. The solar wind is a continuous flow of energetic charged particles which are correlated with geomagnetic field. The cloud formation models depend on the Earth's atmospheric properties such as density and air composition. Monitoring clouds with high accuracy is a difficult task and can only be done by satellite. Simulations have an important role in studying the correlation between the cosmic ray flux and the Earth's climate.

This chapter presents a comparative analysis of the geomagnetic field and atmospheric density variation effects for cosmic ray air shower development. Section of 5.1 describes the ECRS simulation configuration and section 5.3 describes the simulation results obtained for both magnetic field and air density variations using longitudinal and lateral shower development. The different shower components analysis such as secondary muon and gamma particles distribution are also included with their lateral and longitudinal distributions.

5.1 ECRS Simulation Configuration

The ECRS simulation has been developed to study cosmic ray shower development with and without a geomagnetic field at a given location on the Earth's surface. The atmospheric air density factor is introduced to change the atmospheric density in the air. The default value of atmospheric density factor is one, which is represents normal atmospheric density. In the first analysis, the main objective is to study the effect of magnetic field as well as air density on extensive air shower development using the ECRS simulation.

The simulation of an extensive air shower was performed for Atlanta, Georgia (33.46^0 N, 84.250 W) at four primary proton energies: 10^9, 10^{10}, 10^{11}, and 10^{12} eV. The primary particles at each energy were started at 200 km above the Earth's surface as shown in Fig. 5.1 (a). The incoming primary particle direction was pointed towards Atlanta. All necessary electromagnetic and hadronic physics processes were included. The Simulation output is a list of all secondary particles coordinates, particle type, 3-momenta, and energies. The simulated output data were input to a data filtering program, which is written in C++

based on ROOT [58]. The filtering program was used to reconstruct the row data for studying longitudinal and lateral distribution of the particle shower as shown in Fig.5.1 (b).

Longitudinal distribution is defined as secondary cosmic ray particle development along the shower axis. It also allows to determine height of shower maximum (X_{max}) which is important for estimating primary particle interactions with air molecules. The shower axis is the initial direction of the primary particle and is shown in Figure 5.1 (b). Lateral distribution is defined as secondary cosmic ray particle distribution from the selected position on the Earth's surface. This parameter is very important for understanding shower particle spread at a given location on the Earth surface. For study of the magnetic field and density effects the following cases are simulated with ECRS.

- With geomagnetic field
- Without magnetic field
- Realistic air density
- Half air density (density factor is 0.5)
- Double air density (density factor is 2)

In each case, 10,000 events were simulated for 10^9, and 10^{10} eV primary energies and 1200 events for 10^{11}, and 10^{12} eV primary energies. In total, more than 100,000 events were simulated for these five cases with four sets of primary energies.

5.2 Computation Resources

In order to obtained full knowledge about secondary particle distributions from cosmic ray showers a minimum number of events have to be simulated with ECRS. This requires extensive computing resources to complete these simulations. For a example, a 10^{12} eV primary proton event requires more than seven hours to complete on 2 GHz CPU, 1GB memory node.

For this set of simulations, all simulation tasks were performed at RHIC Computing Facility at Brookhaven National Laboratory. There are more than 1000 powerful computer nodes (2 GHz cpu with 1 GB memory) shared by multiple users. ECRS air shower simulation occupied on average 40-60 nodes at a time and running continuously for five months. The simulation output was transferred to GSU for final data analysis.

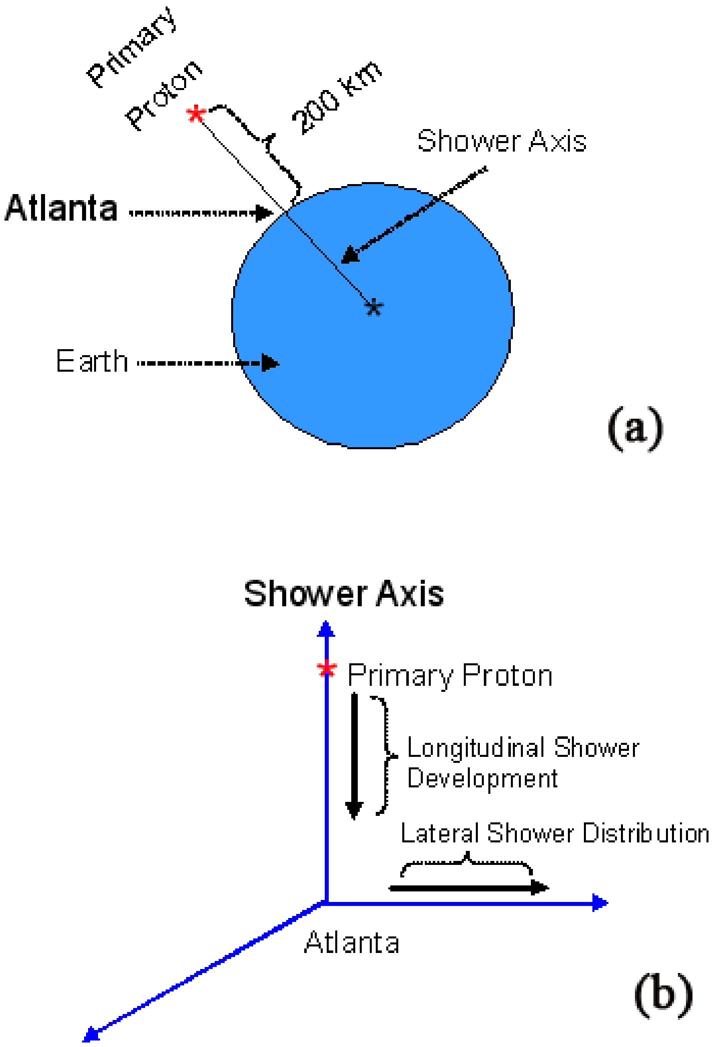

Figure 5.1: (a) Schematic diagram of the ECRS primary particle simulation. Primary protons with different energies are launched 200 km above Atlanta towards the Earth's center. The axis between the initial primary particle position and Atlanta is defined as the shower axis. The blue sphere represents the Earth. (b) Schematic diagram of longitudinal and lateral particle distribution at Atlanta after necessary axis rotation. The Z direction points upwards along the shower axis direction, and X,Y plane is on the Earth's surface.

5.3 Results and Discussion

5.3.1 Shower Features

Some of the main features of a shower cascade can be modeled within a simple Toy model of particle cascade [61]. This model assumes that a primary particle with energy E_0, splits energy equally into two particles after traveling a path length λ, and this process is repeated by the secondaries. Then, the number of particles n and energy E in a cascade at a longitudinal height X (measured from initial primary particle towards the Earth's surface) evaluated as,

$$n(X) = 2^{\frac{X}{\lambda}} \tag{5.1}$$

$$E(X) = \frac{E_0}{N(X)} \tag{5.2}$$

Further assume that particle multiplication stops when a certain energy limit E = E_f is reached. Then the maximum number of particles n_{max} is reached at this point X_{max}, and it is given by,

$$n_{max} = n(X_{max}) = \frac{E_0}{E_f} \tag{5.3}$$

The position of X_{max} follows as,

$$X_{max} = \frac{\lambda}{ln2} \times ln\left(\frac{E_0}{E_f}\right) \tag{5.4}$$

In the Toy model, consider the more general case of an initial set (A_0, $\frac{E_0}{A_0}$) of A_0 particles, each with energy $\frac{E_0}{A_0}$. Where the initial particle set A_0, ($\frac{E_0}{A_0}$) as primary nucleus of mass number A_0. The maximum number of particles and the position of the shower maximum from the initial particle is then given by,

$$N_{max}\left(A_0, \frac{E_0}{A_0}\right) = N_{max}(E_0) \propto E_0, \tag{5.5}$$

$$X_{max}\left(A_0, \frac{E_0}{A_0}\right) ln \frac{E_0}{A_0 E_f} \leq X_{max}(E_0) \tag{5.6}$$

If we identify the initial particle as primary nucleus of A$_0$, the toy model predictions can be summarized from Equation 5.5 and 5.6 as,

- n$_{max}$ increases in proportional to the primary energy
- X$_{max}$ increases as the logarithm of the primary energy (Here X$_{max}$ is measured from the primary particle initial position)
- X$_{max}$ is the same for same $\frac{E_0}{A_0}$ but different for different E_0's

Figure 5.2 shows the maximum number of particles (n) as a function of primary proton energy. All secondary particles are taken into account during the cascade development in the ECRS simulation. Triangles, stars, and solid dots show the total number of all secondary, gamma, and muon particle (n$_{max}$) variation with primary energies, respectively. This figure clearly illustrates how n$_{max}$ increases with primary particle energies for any secondary particles as stated in the Toy's model. On the other hand, it can be concluded that n$_{max}$ increases in proportional to the primary energy. This ECRS result is consistent with theoretical predictions. Other basic theoretical predictions with ECRS results are discussed in section 5.3.2.

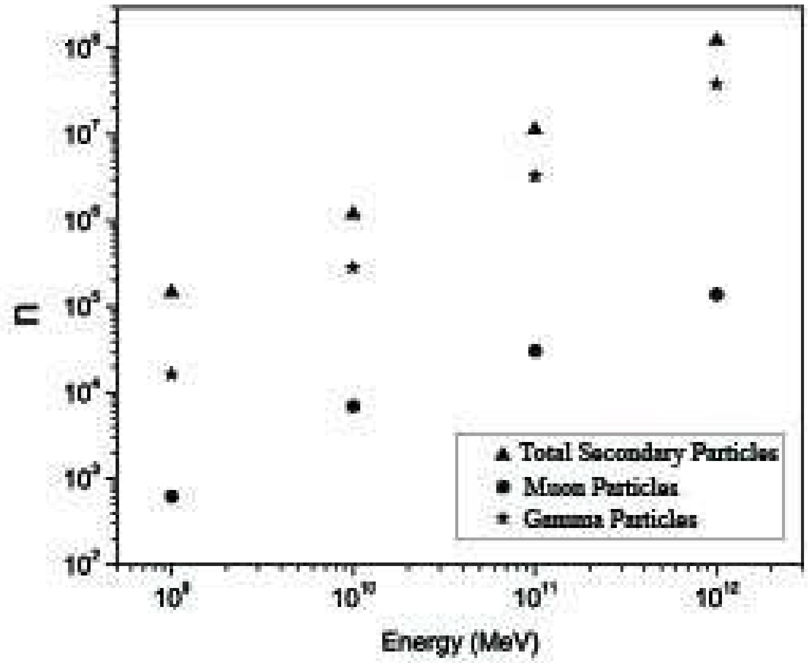

Figure 5.2: ECRS simulated secondary particle variation with primary particle's energies. Triangles, stars, and solid dots shows total, gamma, and muon maximum number of secondary particles respectively.

5.3.2 Magnetic Field Effect

The main objective of this section is to discuss the magnetic field effect on secondary cosmic ray air showers.

Energy Distribution

Figure 5.3 shows energy deposits in terms of particle numbers for total secondary particles (n) created during the cascade development. The dashed line shows the total particle energy distribution when the magnetic field is enabled, and the solid line shows the distribution when the magnetic field is disabled. Four different primary particle energies (10^9, 10^{10},10^{11}, and 10^{12} eV) were used to produce all possible secondary cascades, and it can be understood that there is no significant effect on the total secondary particle energy distribution with magnetic or without field. In general, the secondary particle energy distribution does not show any magnetic field dependency for different primary particle energies.

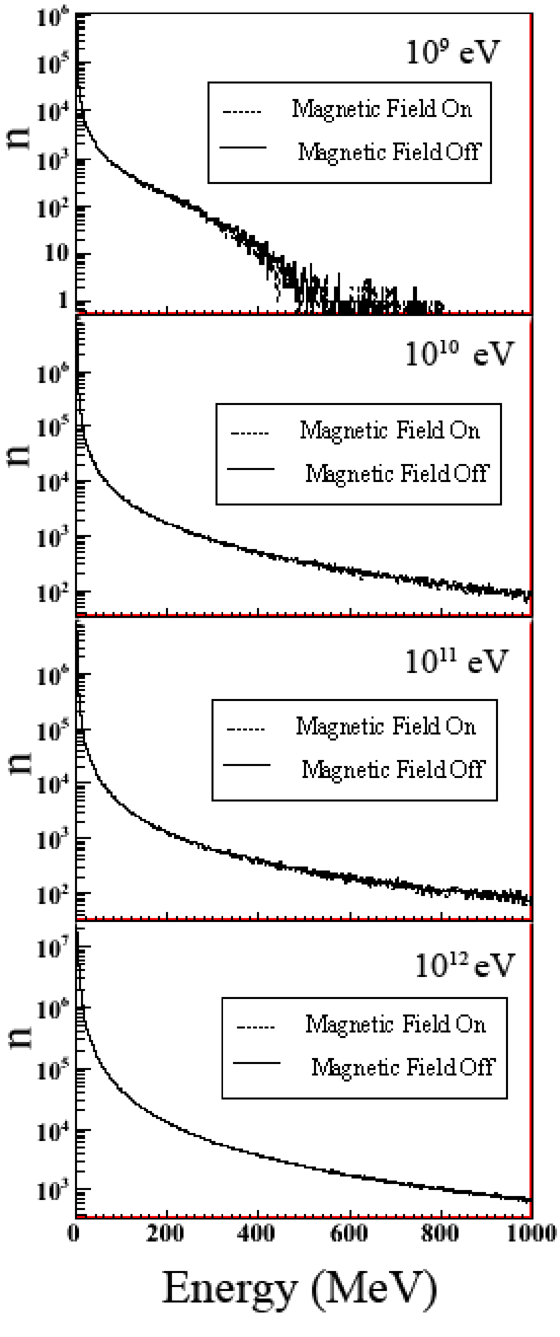

Figure 5.3: All secondary particles variation, when magnetic field is enabled (dashed line) and magnetic field is disabled (solid line) with the secondary particle energies for 10^9, 10^{10}, 10^{11}, and 10^{12} eV primary particles.

Longitudinal Distribution

In large scale cosmic ray shower observations, the longitudinal and lateral distribution provides information about primary particle direction and shower axis position, as well as size of the shower. Thus, the shower maximum (X_{max}) provides information about secondary particle distribution in the atmosphere and indicates about where these primary interactions happen. X_{max} is measured from the Earth's surface in the upward direction.

Figure 5.4 shows the total secondary particles distribution as function of altitude; the so–called longitudinal particle distribution for four different energy distributions. Each primary energy was simulated with (enabled) and without (disabled) geomagnetic field, represented by dashed, and solid lines respectively. Even though two different X_{max} were observed for 10^9 eV primary particle distribution with enabled and disabled magnetic field, there is no significant difference in either the shower maximum (X_{max}) or the number of particles (n_{max}) for primary energies above 10^{10} eV. This is because the 10^9 eV primary shower creates comparatively low energy secondary particles in the shower cascade. Due to the vertical geomagnetic cutoff rigidity (for Atlanta, it is 3.6 GV), most of these secondary particles do not have enough energy to reach the Earth surface; they are trapped in the geomagnetic field. When magnetic field is disabled, the 10^9 eV primary shower creates less number of secondary particles near the Earth's surface compared to the disabled magnetic field. On the other hand, it is clear that high energy primary particles (10^{10} eV or above) are not depending much on the cutoff rigidity, since most secondaries are higher energy than the cutoff value. Therefore, the magnetic field effects on longitudinal distribution and number of secondary particles are strongly effected for the low energy primary particles but not for the high energies. The other important characteristic is relationship between X$_{max}$ and primary particle energies. It can be seen that X$_{max}$ decreases as the primary energy increases.

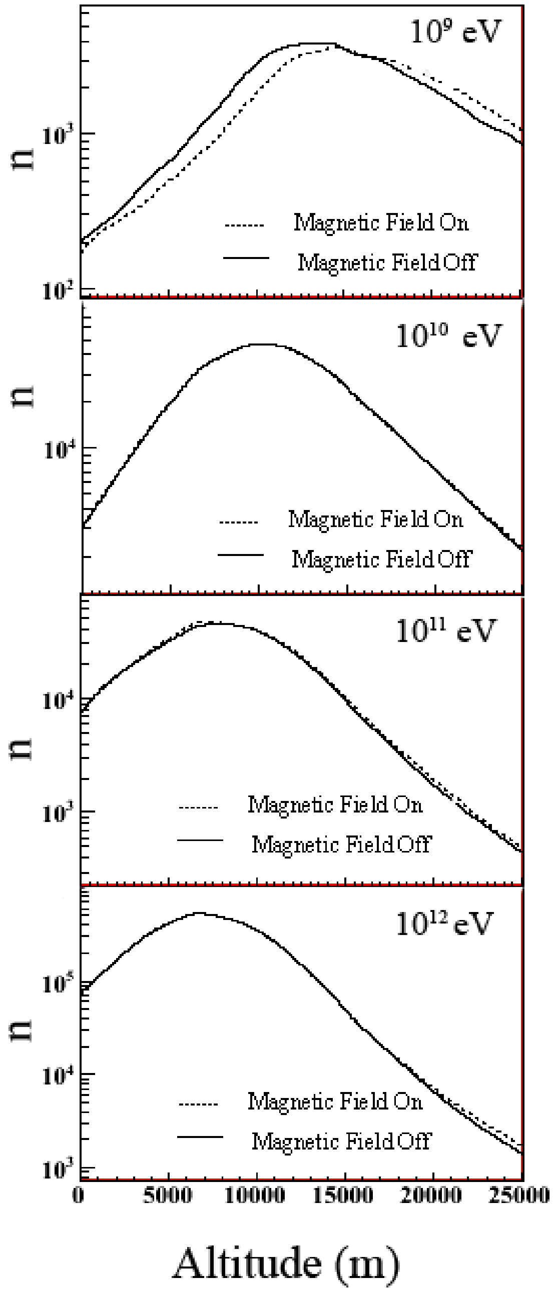

Figure 5.4: Secondary particles longitudinal development with enable and disable magnetic field. Dash and solid curve represent enable and disable magnetic field respectively for 10^9, 10^{10}, 10^{11}, and 10^{12} primary particles.

Figure 5.5 shows X$_{max}$ variation as a function of primary energies for both enabled and disabled geomagnetic field. According to the Toy model X$_{max}$ increases with the logarithm of the primary energies; X$_{max}$ was measured from initial primary particle position towards the Earth. In the ECRS simulation, X$_{max}$ is defined in the opposite direction (from Earth's surface to upward direction), and therefore, X$_{max}$ should decreases with the logarithm of the primary energy. Figure 5.5 clearly shows that X$_{max}$ decreases with the logarithm of primary energies as predicted by theory.

Similar behavior can be observed for other secondary particles as well. Among the other secondary particles, muon and gamma particles lateral distribution were also analyzed. As visible from the longitudinal distribution in Figure 5.6 (Right), the muon particle X$_{max}$ increases as primary energy increases for 10^{10}, 10^{11}, and 10^{12} eV primary particles. 10^{9} eV primary particles were not included due to the very small number of secondary muon particles created. Moreover, Figure 5.7 (Right) shows similar gamma distributions for the different primary energies. These two types of secondary particles lateral distribution are also not dependent on geomagnetic field expect energies below 10^{10} eV. This analysis is summarized below.

- ECRS secondary particles shows good agreement with theoretical predictions.
- Secondary particle longitudinal distribution does not depend on the geomagnetic field except for energies below 10^{10} eV.
- Longitudinal distribution strongly depends on the primary particle's energy, and decreases as primary particle energy increases.

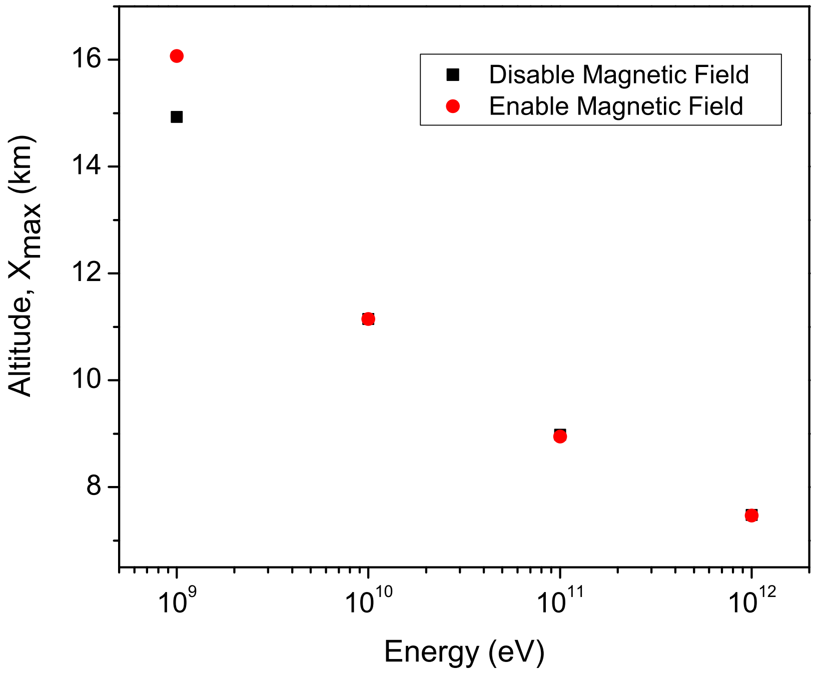

Figure 5.5: Compilation of X_{max} variation for four different primary energies. Solid dots, and solid squares show enabled, and disabled magnetic fields respectively.

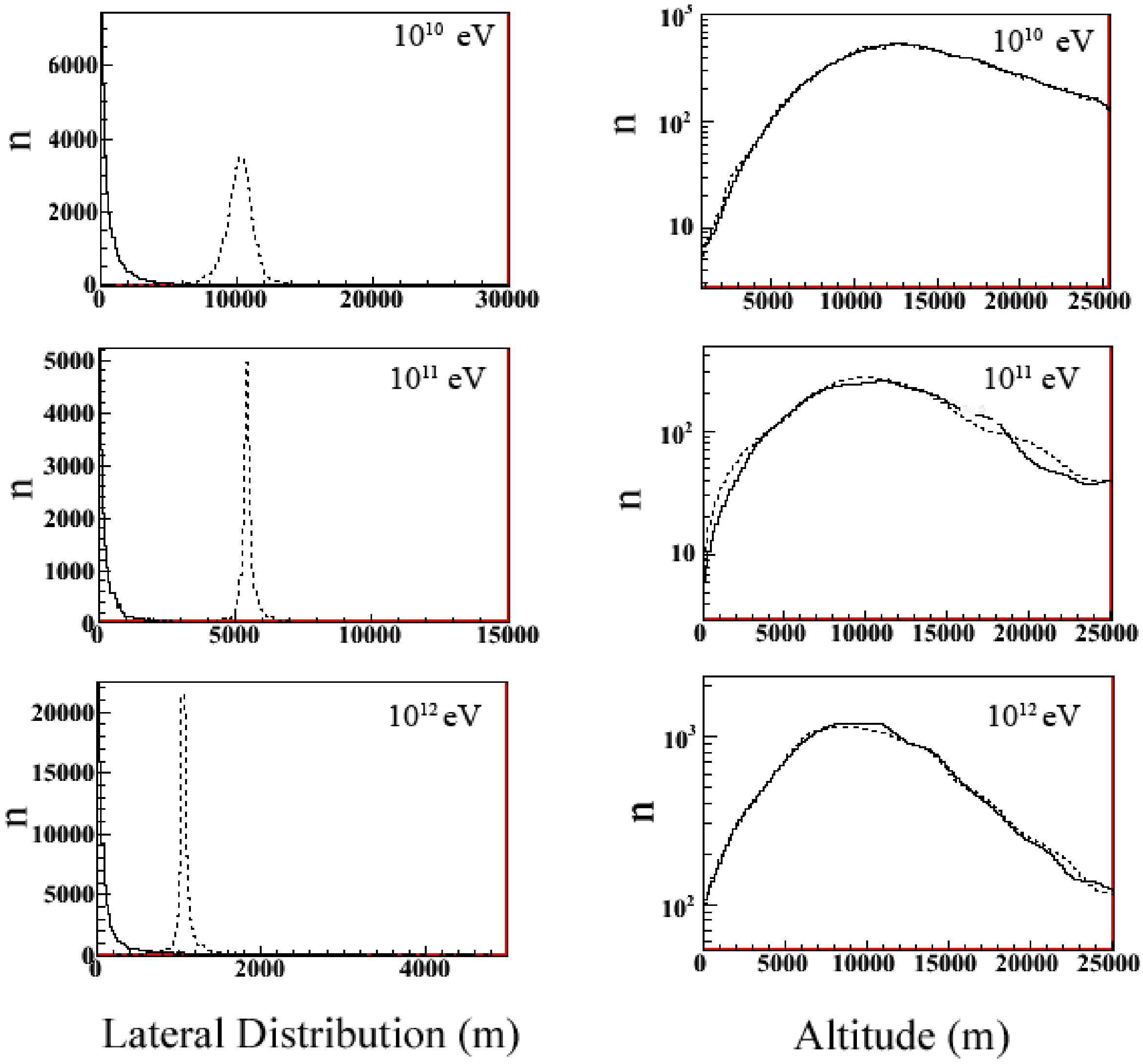

Figure 5.6: (Left) Muon particle's lateral distribution for three different primary energy variations with geomagnetic field. (Right) Muon particles' longitudinal distribution for the three different primary energies. Dashed and solid lines represent enabled and disabled magnetic fields respectively.

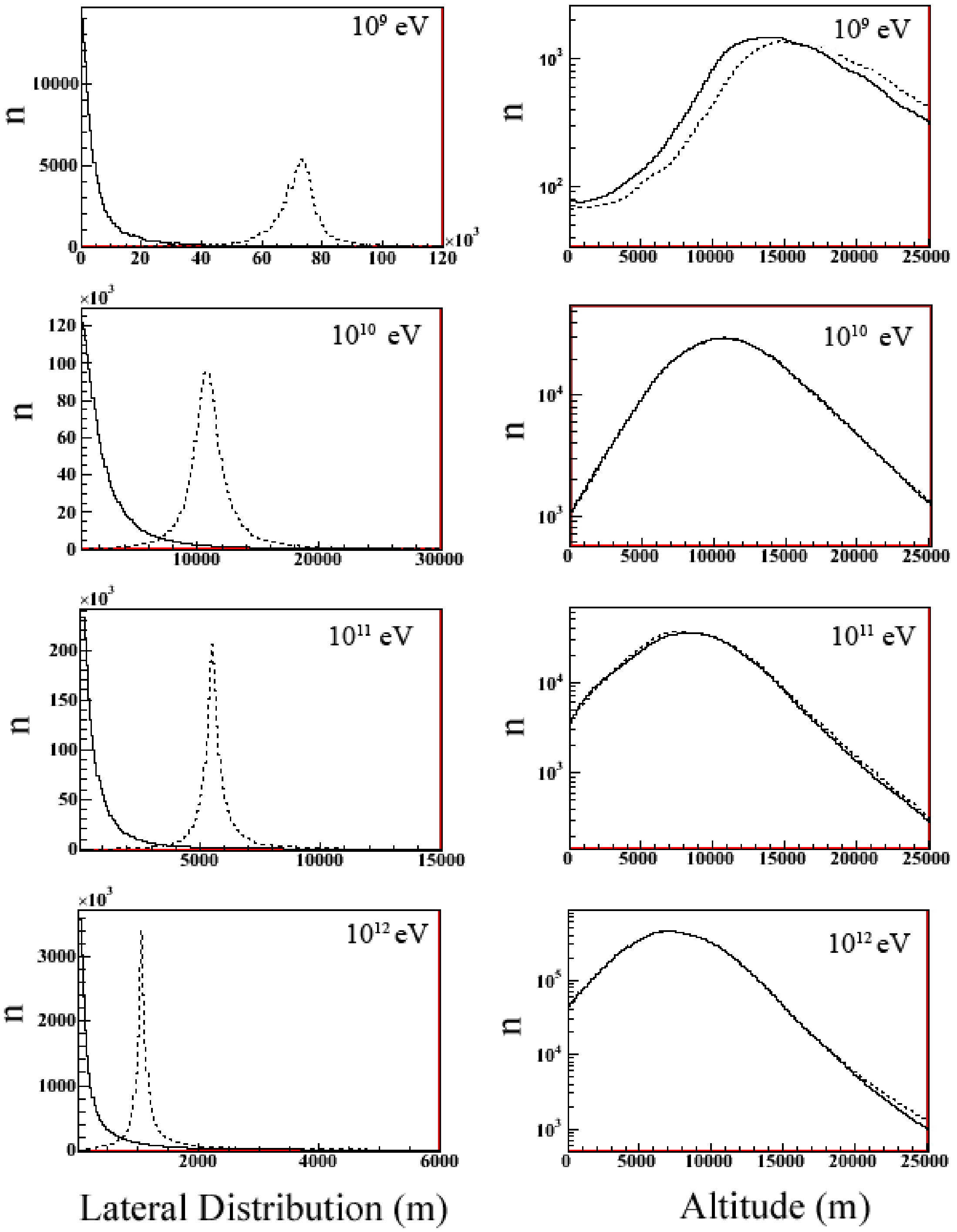

Figure 5.7: (Left) Gamma particles' lateral distribution for four different primary energy variations with geomagnetic field. (Right) Gamma particles' longitudinal distribution for the four different primary energies. Dashed and solid lines represent enabled and disabled magnetic fields respectively.

Lateral Distribution

Air showers have a lateral distribution that differs for the different shower components as well as for the various primary energies. Figure 5.8 (left) shows the total particle lateral distribution with an enabled and disabled magnetic field for three different selected primary energies, namely 10^9, 10^{10}, and 10^{11} eV . Figure 5.8 (right) shows the same distribution, but with a different scale including the primary energy 10^{12} eV. An arrival shower axis is represented by the peak position of lateral distribution. Almost all of secondary cosmic ray particles are distributed around the shower axis. It is clear that the shower axis is strongly dependent on the geomagnetic field. When the magnetic field is disabled, the shower axis is always on the y-axis for all primary energies. This is because the incoming primary particle's direction does not change all the way down to the Earth's surface, and all the particle interactions happen exactly above Atlanta. It is obvious that without the magnetic field, incoming energetic charged particles are not deflected except for slight changes due to particle collisions in the atmosphere. When the geomagnetic field is enabled, there is huge deflection in the shower axis for low energy particles, but comparatively less deflection of high energy particles. It is clear that low energy charged particles are highly dependent on the magnetic field, and this effect is less effect for high energy primary particles. Similar behavior can also be observed for secondary particles such as muons, and gammas. Figure 5.6 (Right) and 5.7 (Right) show similar lateral distributions with magnetic field for different primary energies.

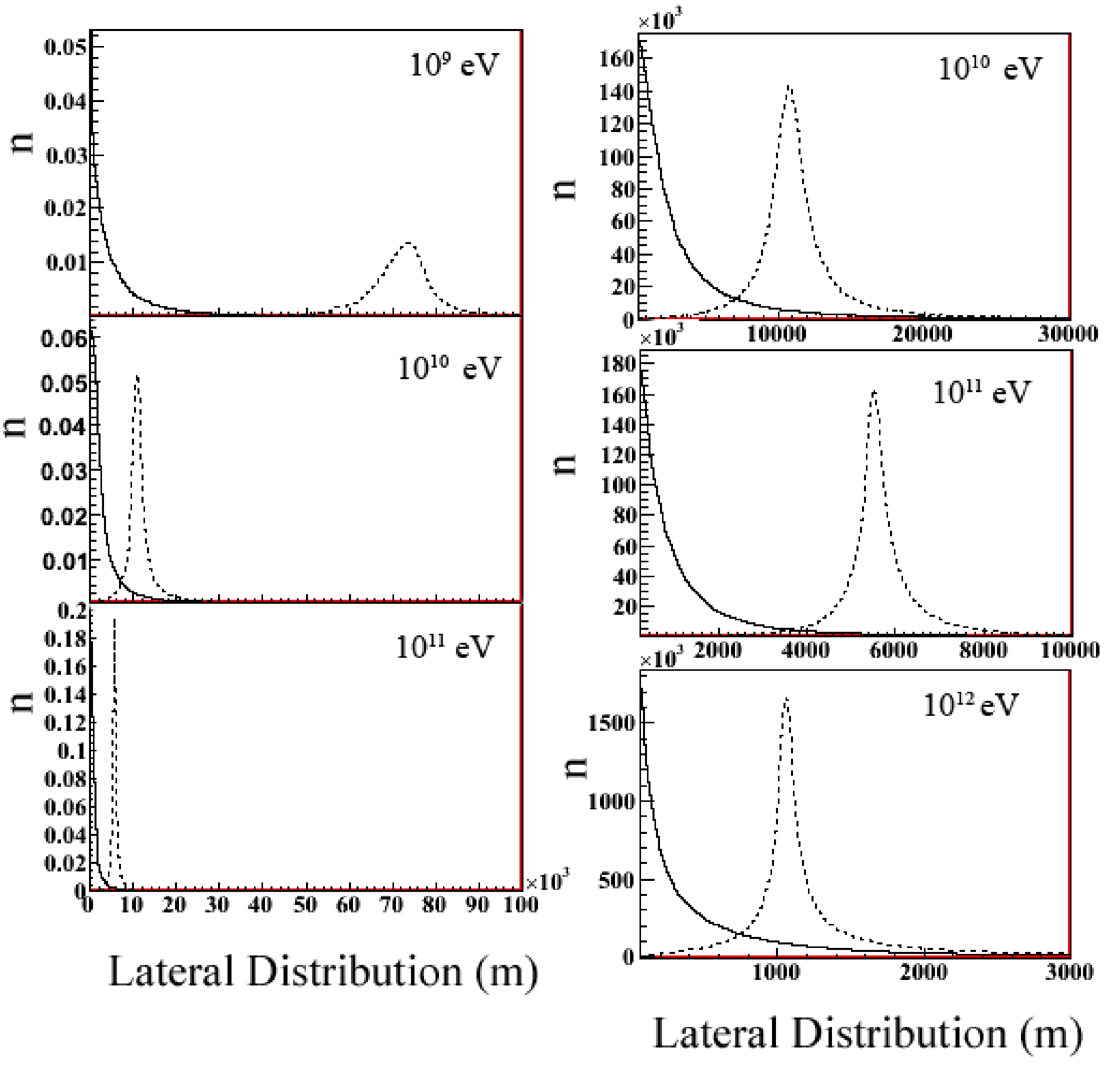

Figure 5.8: (Left) Total secondary particles' lateral development variation with enabled and disabled magnetic field for 10^9, 10^{10}, and 10^{11} eV primary particles on the same scale. (Right) Similar variations with different scales for 10^{10}, 10^{11}, and 10^{12} eV. Dashed and solid lines represent enabled and disabled magnetic fields respectively.

The other important characteristic is the lateral distribution as a function of primary particle energy. The lateral distribution of high energy particles creates large numbers of secondaries which are distributed over a large area compared to the lower energy primary particles. Also, the size of lateral distribution is an important parameter for the design of detector arrays to study the specified high energy primary charge particle energies and arrival direction. Figure 5.9 shows center of shower axis (peak position of lateral distribution) variation with primary particle energies. A 10^9 eV primary particle shower axis is approximately away 70 km from the initial shower direction, while a 10^{12} eV primary particle axis is 1.2 km from the original direction. To summarize:

- Secondary particle lateral distribution strongly depends on geomagnetic field.
- It also depends on the primary particle energy; lateral distribution decreases as primary particle energy increases.

Tables 5.1, 5.2, and 5.3 summarize secondary total, muon, and gamma particle lateral and longitudinal distribution for four different energies with enabled and disabled geomagnetic field.

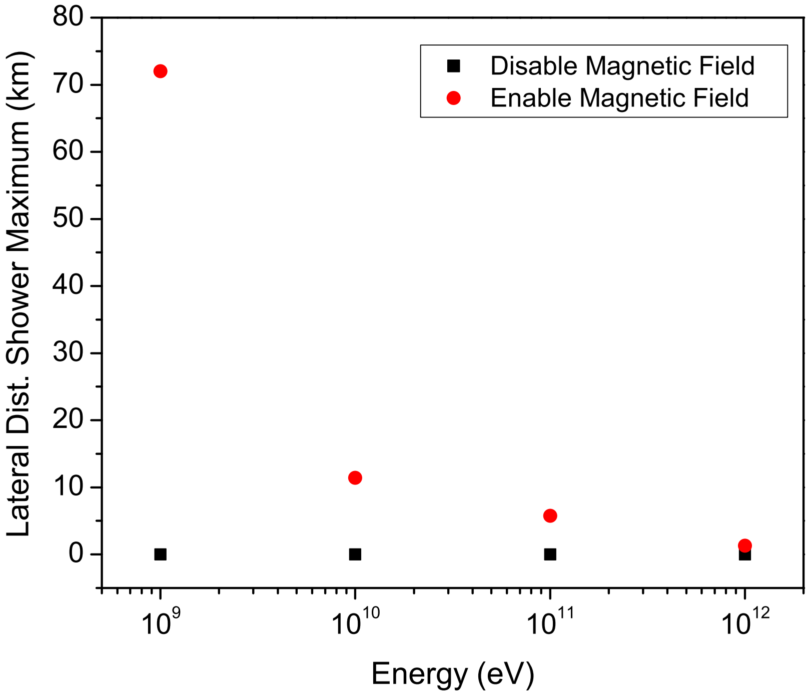

Figure 5.9: (Left) Total secondary particles lateral development variation with enabled and disabled magnetic fields for 10^9, 10^{10}, and 10^{11} eV primary particles in a same scale. (Right) Similar variation in different scales for 10^{10}, 10^{11}, and 10^{12} eV. Dashed and solid curve represent enabled and disabled magnetic fields respectively.

Table 5.1: All particle lateral and longitudinal shower maxima for different primary particle energies with magnetic and without magnetic field at Atlanta.

Primary Energy(eV)	Lateral	Distribution (m)	Longitudinal	Distribution (m)
	Magnetic Field Enabled	Magnetic Field Disabled	Magnetic Field Enabled	Magnetic Field Disabled
10^9	72.03	0.0	16.07	14.93
10^{10}	11.42	0.0	11.15	11.15
10^{11}	5.75	0.0	8.45	8.48
10^{12}	1.29	0.0	7.47	7.48

Table 5.2: Muon particle lateral and longitudinal shower maxima for different primary particle energies with magnetic and without magnetic field at Atlanta.

Primary Energy(eV)	Lateral	Distribution (m)	Longitudinal	Distribution (m)
	Magnetic Field Enabled	Magnetic Field Disabled	Magnetic Field Enabled	Magnetic Field Disabled
10^9	68.32	0.0	20.02	19.70
10^{10}	10.24	0.0	14.99	15.02
10^{11}	5.52	0.0	12.25	12.20
10^{12}	1.21	0.0	11.04	11.00

Table 5.3: Gamma particle lateral and longitudinal shower maxima for different primary particle energies with magnetic and without magnetic field at Atlanta.

Primary Energy(eV)	Lateral	Distribution (m)	Longitudinal	Distribution (m)
	Magnetic Field Enabled	Magnetic Field Disabled	Magnetic Field Enabled	Magnetic Field Disabled
10^9	71.75	0.0	16.87	15.58
10^{10}	11.17	0.0	11.67	11.61
10^{11}	5.67	0.0	8.77	8.81
10^{12}	1.27	0.0	7.72	7.72

5.3.3 Air Density Effect

The main objective of this section is to discuss the effect of air density on secondary cosmic ray particles. Three different air densities are considered as normal, half (0.5×air density) and double (2×air density) with four different primary energies, 10^9,10^{10}, 10^{11}, and 10^{12} eV.

Longitudinal Distribution

Figure 5.10 shows secondary particle (n) as a function of longitudinal distribution for three different air densities. Each different density profile was simulated with four different primary energies as mentioned above. The air density variation shows a significant effect on longitudinal shower distribution. The X_{max} (shower-max) is correlated with air density variation. It is clearly seen that X_{max} increases as density increases for four different energies. Also, X_{max} decreases as energy increases for the three different densities.

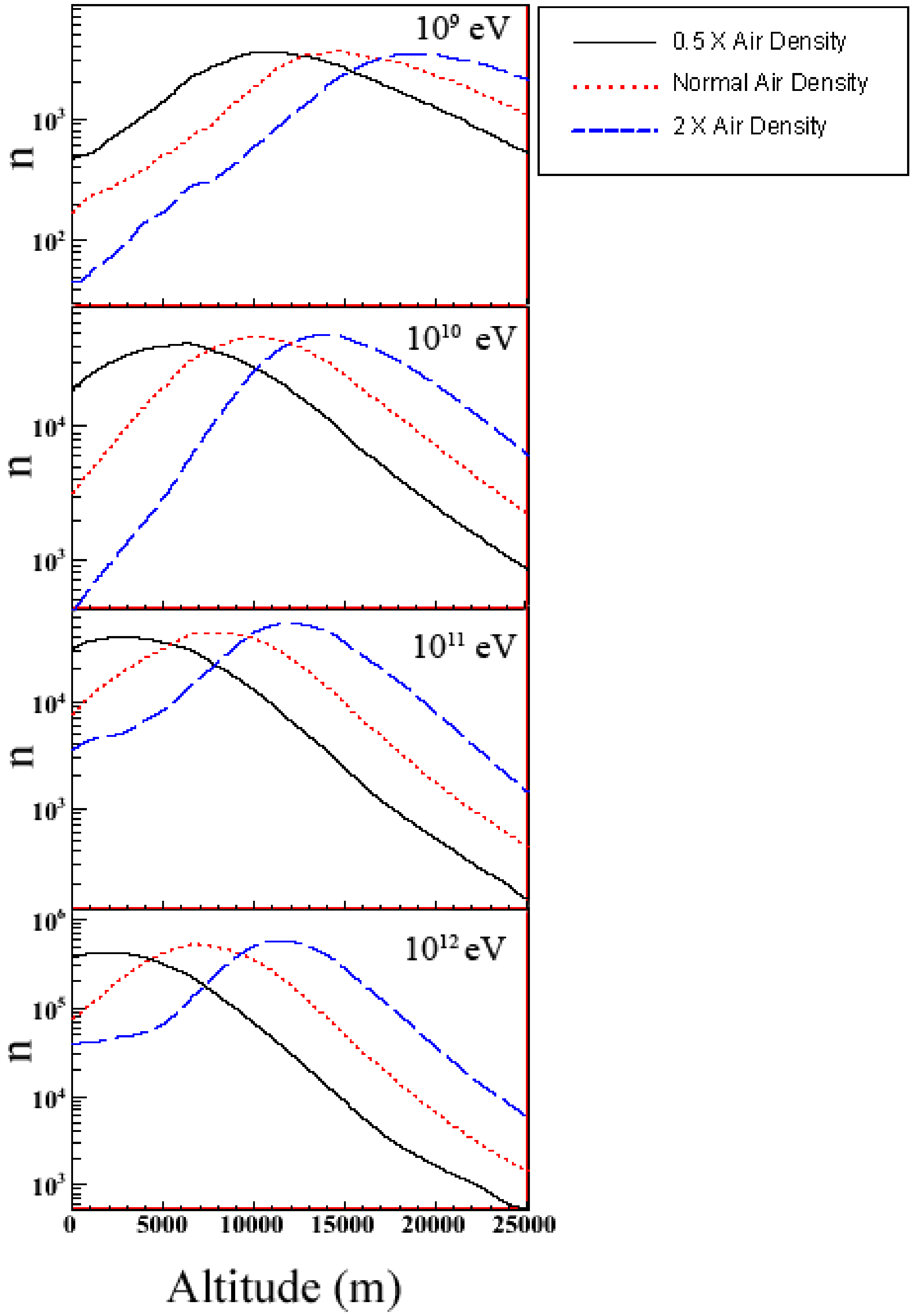

Figure 5.10: Total secondary particle longitudinal distribution (shower maximum) as a function for three air densities. Black (solid), red (short dashed), and blue (long dashed) lines represent half, normal, and double air densities respectively for 10^9, 10^{10}, 10^{11}, and 10^{12} eV primary proton energies.

The air density effects on X$_{max}$ distribution are consist with mean free path of a particle in a given medium. The formula for calculating the magnitude of mean free path depends on the characteristics of the medium the particle is interacting with, and it is given by,

$$l = \frac{1}{n\sigma} \tag{5.7}$$

where l is the mean free path, n is the number of particles per unit volume, and σ is the effective cross–section of collision. The density of the medium is correlated to σ. If, on the other hand, σ increases as air density increases, the mean free path decreases. Therefore, in a high density medium one can expect more interactions far above the Earth's surface compared to a low density medium.

Figure 5.12 (right), and Figure 5.13 (right) show the similar longitudinal distribution for secondary muons, and gammas respectively.

Lateral Distribution

Figure 5.11 shows analysis of lateral distribution of secondary particle for different air densities and primary energies. Considerable shower axis deflections are shown for the 10^9 eV and 10^{10} eV primary particles with three different air densities, but for higher energies the deflection is insignificant. It is clear that only low energy particles' lateral distributions strongly depend on air density while high energy particles' lateral distribution remain the same. Lateral distribution of secondary muons and gammas are shown in Fig. 5.12 and 5.13. Tables 5.4, 5.5, and 5.6 summarize secondary total, muon and gamma particle lateral and longitudinal variations due to air density changes.

In summary, ECRS simulation provides an important tool for EAS data reconstructions. On the other hand, cosmic ray flux is directly related to air density variations. Air density can vary due to the weather condition and climate the changes are possible area of future studies.

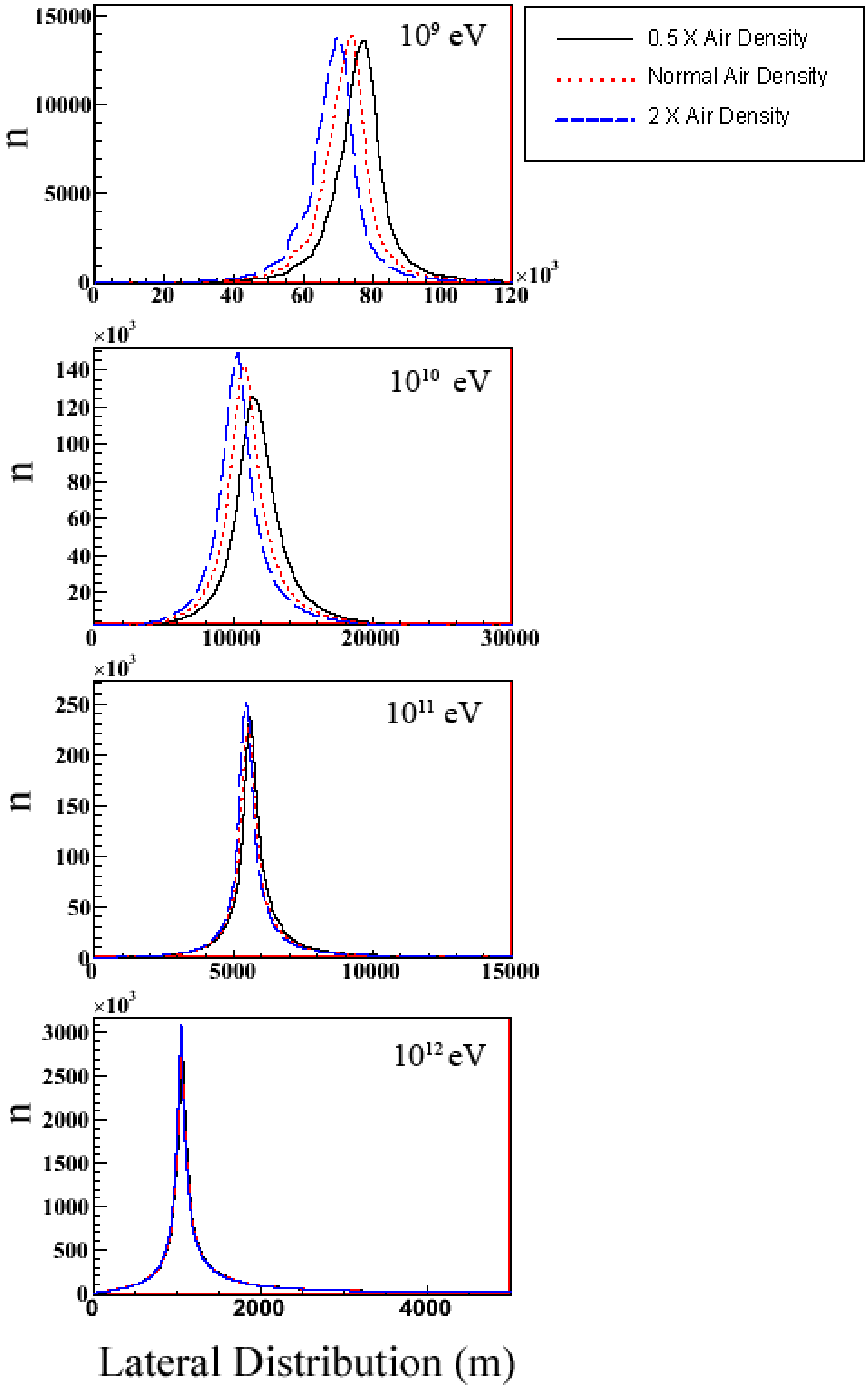

Figure 5.11: Total secondary particle lateral distribution with density variation. Black (solid), red (short dashed), and blue (long dashed) lines represent half, normal, and double air densities respectively for 10^9,10^{10}, 10^{11}, and 10^{12} eV primary proton energies.

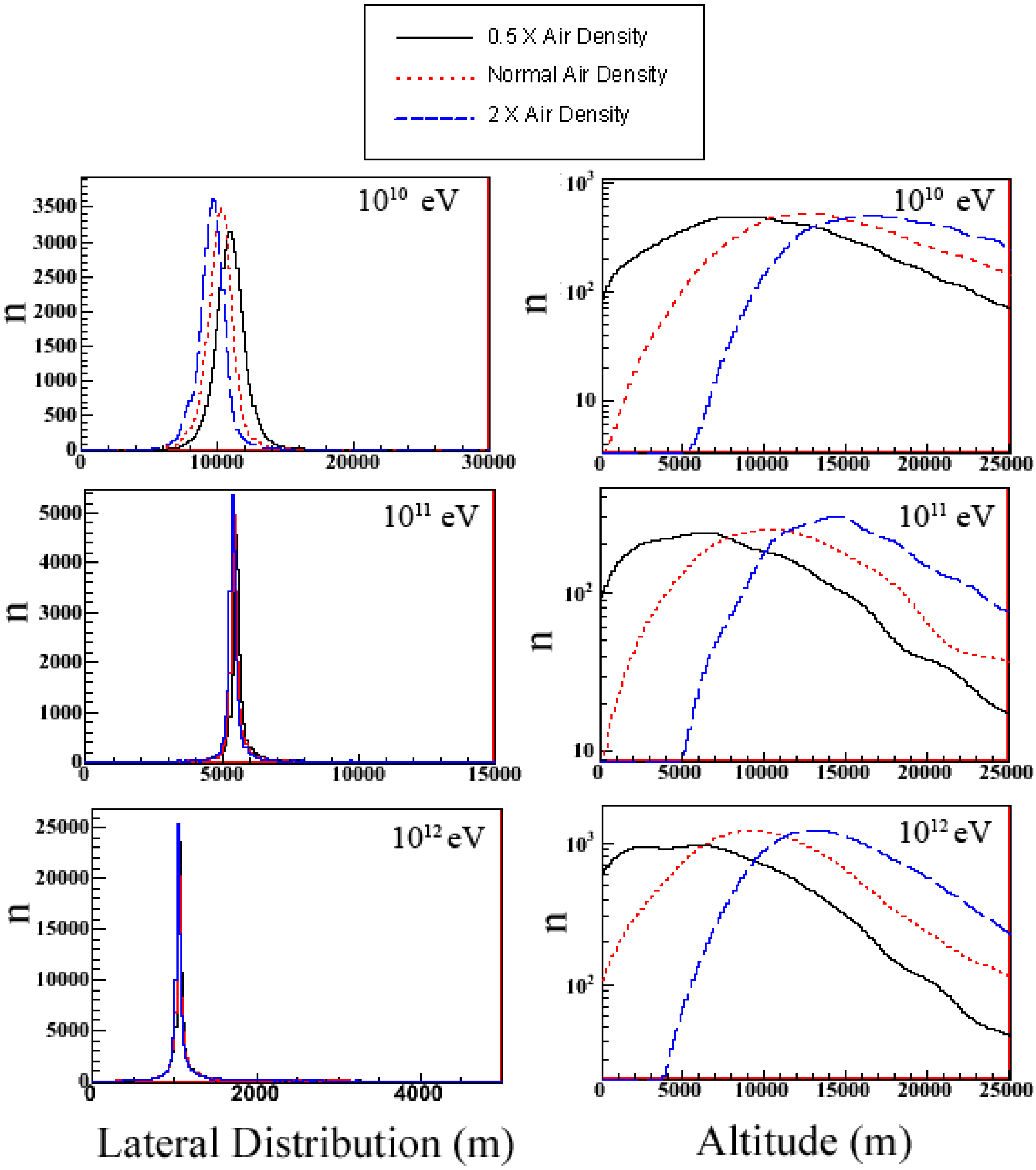

Figure 5.12: (left) Secondary muon particle lateral distribution, and (right) longitudinal distribution with density variation. Black (solid), red (short dashed), and blue (long dashed) lines represent half, normal, and double air densities respectively for 10^{10}, 10^{11}, and 10^{12} eV primary proton energies.

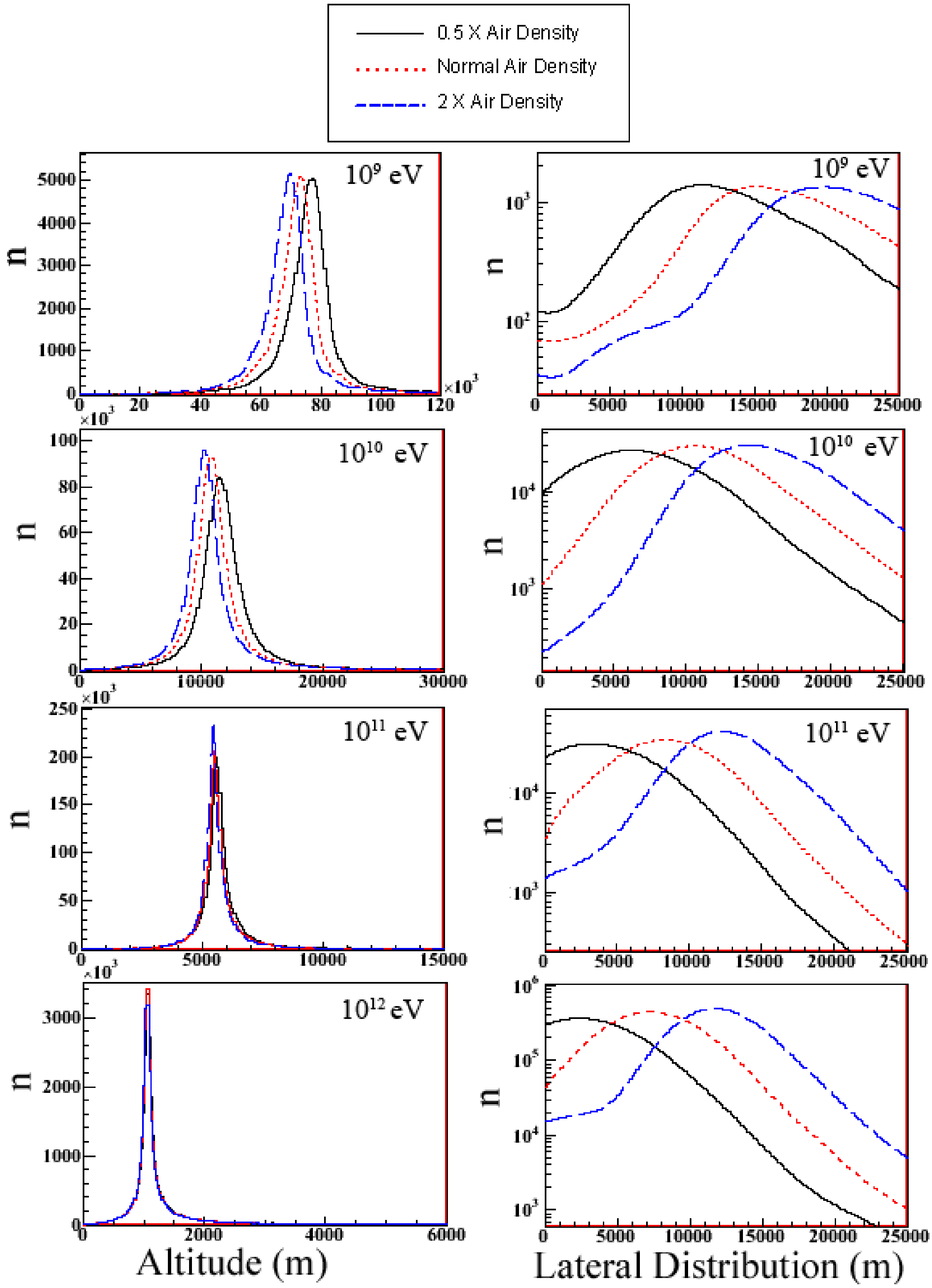

Figure 5.13: (left) Secondary gamma particle lateral distribution, and (right) longitudinal distribution with density variation. Black (solid), red (short dashed), and blue (long dashed) lines represent half, normal, and double air densities respectively for 10^9, 10^{10}, 10^{11}, and 10^{12} eV primary proton energies.

Table 5.4: Total particle lateral and longitudinal shower maxima for different primary particle energies. The three different air density variations are given for Atlanta.

Primary Energy(eV)	Lateral	Distribution	(m)	Longitudinal	Distribution	(m)
	0.5 X Density	1 X Density	2 X Density	0.5 X Density	1 X Density	2 X Density
10^9	75.90	72.03	68.56	12.7	16.07	20.31
10^{10}	12.14	11.42	10.85	7.56	11.15	15.20
10^{11}	5.86	5.75	5.68	5.31	8.45	12.28
10^{12}	1.29	1.29	1.29	4.31	7.47	11.55

Table 5.5: Muon lateral and longitudinal shower maxima for different primary particle energies. The three different air density variations are given for Atlanta.

Primary Energy(eV)	Lateral	Distribution	(m)	Longitudinal	Distribution	(m)
	0.5 X Density	1 X Density	2 X Density	0.5 X Density	1 X Density	2 X Density
10^9	72.12	68.32	64.67	16.75	20.02	22.96
10^{10}	11.00	10.24	9.65	11.33	14.99	18.61
10^{11}	5.60	5.52	5.45	9.02	12.25	16.14
10^{12}	1.21	1.21	1.20	7.95	11.04	15.29

Table 5.6: Gamma particle lateral and longitudinal shower maxima for different primary particle energies. The three different air density variations are given for Atlanta.

Primary Energy(eV)	Lateral	Distribution	(m)	Longitudinal	Distribution	(m)
	0.5 X Density	1 X Density	2 X Density	0.5 X Density	1 X Density	2 X Density
10^9	75.78	71.75	68.46	13.36	16.87	20.02
10^{10}	11.90	11.17	10.60	7.75	11.67	15.72
10^{11}	5.77	5.70	5.60	5.35	8.77	12.83
10^{12}	1.26	1.27	1.25	4.38	7.72	12.04

Chapter 6

Atmospheric Muons and Charge Ratio

The precise measurement of the muon charge ratio at the Earths surface not only provides information on the propagation of cosmic rays in the atmosphere, but also allows to study the effect of the Earth magnetic field at different geomagnetic locations. Since muons and muon neutrinos are always produced in pairs, the measurement of muons is important for studying neutrino flux in Earth-bound neutrino experiments [63, 64, 65, 66].

In evaluating the flux of muon neutrinos, a dominant systematic error arises because of the uncertainty in the flux of primary cosmic rays and the production cross section of secondary mesons [67]. The measurement of atmospheric muons is crucial estimating neutrino flux. Atmospheric muons are produced by pions decay or hadronic decays. However, $\pi \rightarrow \mu$ decay is a dominant decay process for atmospheric muon production and this contribution is about 90% below 10 GeV/c, and about 80% at 100 GeV/c for the vertical direction. For the neutrinos, $\pi \rightarrow \mu$ decay is a major decay process below 30 GeV/c for the vertical direction [68]. Therefore the precise calculation of the atmospheric muon flux is important to minimize the systematic errors in the study of neutrino oscillations.

The calculation of atmospheric muon charge ratio at the Earth's surface in Atlanta, USA (33.40^0 N, 84.20^0 W), and Lynn Lake, Manitoba, Canada (56.50^0 N, 101.00^0 W) are presented in this chapter.

6.1 Experimental Measurement of the Charge Ratio

The muon spectrum and charge ratio at the Earth's atmosphere has been measured by different experiments such as NMSU-WIZARD/CAPRICE magnetic spectrometer [69], BESS [70], BARS spectrometer [71], and OKAYAMA cosmic ray telescope [72]. Experimentally, the μ^+/μ^- charge ratio at the Earth's surface is known to be approximately 1.2 – 1.3 from energies of a few GeV up to 1 TeV, as shown in Figure 6.1. In the ECRS simulation, the muon charge ratios for selected locations were calculated and are compared with existing CAPRICE and BESS experimental data.

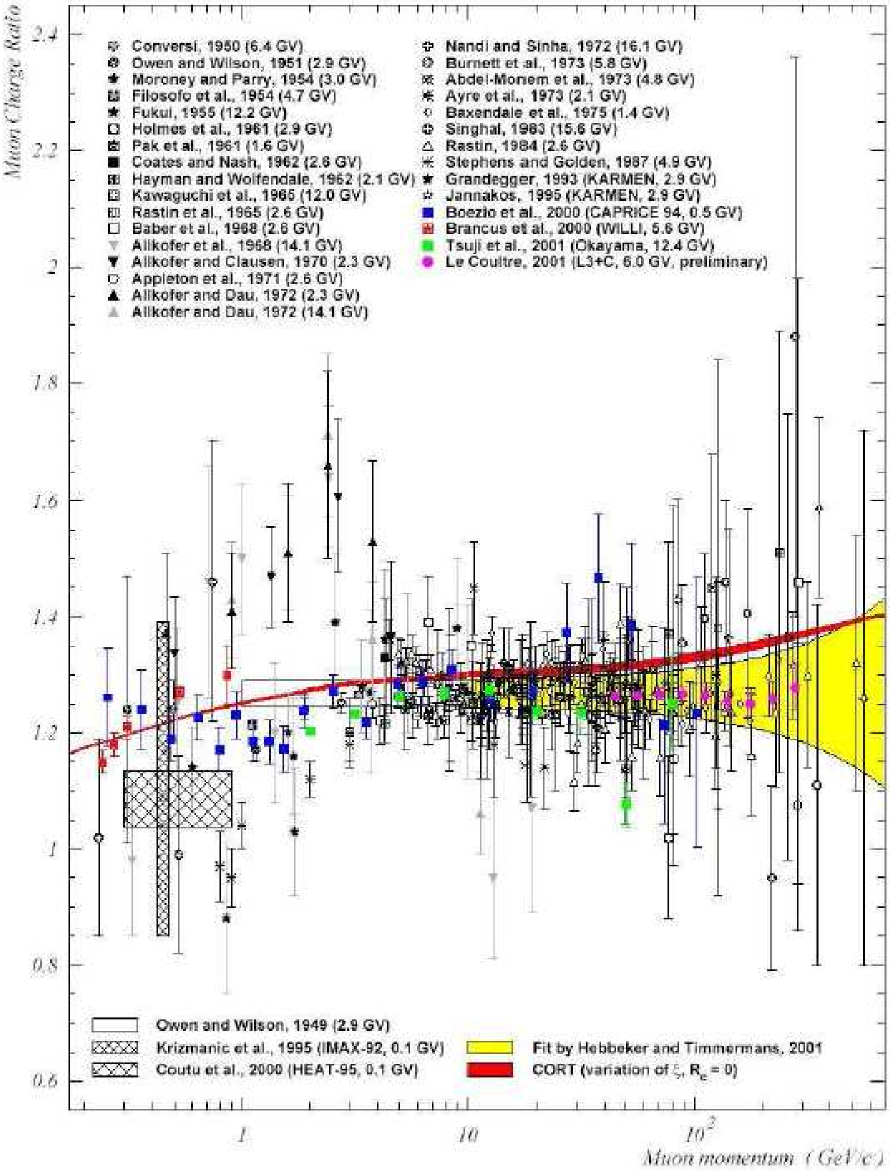

Figure 6.1: Compilation of recent muon charge ratio results at the Earth's surface as a function of muon momentum.

6.1.1 CAPRICE Experiment

The NMSU-WIZARD/CAPRICE magnetic spectrometer measured ground level muon flux and charge ratio at Lynn Lake, Manitoba, Canada (CAPRICE 94 data set) and at Fort Summer, New Mexico, USA (CAPRICE97 data set). This spectrometer was designed as a balloon-borne apparatus [74] and included a ring imaging Cherenkov detector, a time-of-flight system, a superconducting magnetic spectrometer for particle tracking, and a silicon-tungsten imaging calorimeter. It was sensitive to the muon momentum range from 200 MeV/c to 120 GeV/c.

6.1.2 BESS Experiment

The BESS detector [75] was designed as a high resolution spectrometer with a large acceptance to precise measurements of primary and secondary cosmic rays as well as exotic particles. BESS spectrometer measured ground level muon flux and charge ratio at Tsukuba, Japan (BESS95 data set) and Lynn Lake, Manitoba, Canada (BESS97-99 data set). All the detector components of the BESS spectrometer are grouped in a simple cylindrical package. In the central region, a uniform magnetic field (1 Tesla) was produced by a thin superconducting solenoidal coil. The outer part is made up of time of flight scintillator hodoscopes. The energy loss information in the scintillation counters was used to identify the single charged particles.

6.2 Primary Spectrum

In order to calculate muon flux and μ^+/μ^- charge ratio at the Earth's surface, the primary cosmic ray proton particle energy spectrum is required as input of the ECRS simulation. Measurement of the primary spectrum [76] indicates that about 90% of the flux is protons. The number of primary protons with energies above 10^{13} eV is very low compared to the lower energies. Therefore the first assumption is that most of the muon particles are produced by primary protons with energies from 10^9 eV to 10^{13} eV.

In the ECRS simulation incoming primary proton particles' energies was selected randomly ranging from 10^9 eV to 10^{13} eV and launched towards the Earth's center. In general, the muon flux and charge ratio accuracy can be improved by introducing the properly defined primary cosmic ray particle energy spectrum. It is the purpose of this section to discuss the model employed for the primary proton spectrum for the ECRS simulation.

A semiempirical model for the integral primary cosmic ray spectrum has been proposed by Nikolsky, Stamenov, and Ushev(NSU) [77]. In the NSU method, the primary particle flux is given by,

$$F = F_0 E_0^{-\gamma} \sum_A B_A \left(1 + \delta_A \frac{E_0}{A}\right)^{-\alpha} \tag{6.1}$$

Here E_0 is the energy per particle in GeV, F_0 = 1.16 $\mathrm{cm}^{-2}\mathrm{s}^{-1}\mathrm{sr}^{-1}$, $\gamma = 1.62 \pm 0.03$, and $\alpha = 0.4$. δ_A specifies the region of the knee in the primary spectrum where $\delta_A = 6 \times 10^{-7}$ for a proton and $\delta_A = 10^{-5}$ for A $\geq$ 4. The numerical value A indicates the average atomic weight and B is the chemical composition. For the proton, $\mathrm{B}_1 = 0.40 \pm 0.03$. The corresponding differential equation is given by,

$$\frac{dF}{dE_0} = \gamma F_0 E^{(\gamma+1)} \sum_A \left(1 + \delta_A \frac{E_0}{A}\right)^{-\alpha} \times \left(1 + \frac{\alpha \delta_A E_0/A}{\gamma(1 + \delta_A E_0/A)}\right) \tag{6.2}$$

The nuclear component of the primary spectrum is replaced with a superposition of free nucleons. Equation 6.2, transforming to the equivalent nucleon spectrum, yields the following differential energy spectra of protons,

$$\frac{dF_p}{dE_N} = D_1(E_N) + \frac{1}{2} \sum_{A \geq 4} D_A(E_N) \tag{6.3}$$

Here E$_N$ is the nucleon energy in GeV and

$$D_A(E_N) = \frac{C_A D_0 E_N^{-(\gamma+1)}}{(1 + \delta_A E_N)^{\alpha}} \left(1 + \frac{\alpha \delta_A E_N}{\gamma(1 + \delta_A E_N)}\right) \tag{6.4}$$

D$_0$ = γB$_1$F$_0$ = 0.75 cm^{-2}s^{-1}sr^{-1} (GeV/nucleon)$^{-1}$, and C$_A$ =A$^{1-\gamma}$B$_A$/B$_1$. Equation 6.3 is the best approximation for the primary proton flux distribution. Numerical iteration is used to generate the primary proton flux distribution based on the above theoretical approximation. In this numerical iteration, the proton distribution was calculated for two different energy regions as shown in Fig. 6.2. The combined results were used to generate the primary proton flux energy distribution from 10^8 to 10^{13} eV as shown in Fig. 6.3. This distribution is integrated over the selected energy intervals to calculate the number of proton particles required as an input for the ECRS simulation. The results and discussion of muon charge ratio at selected locations are described in the next section.

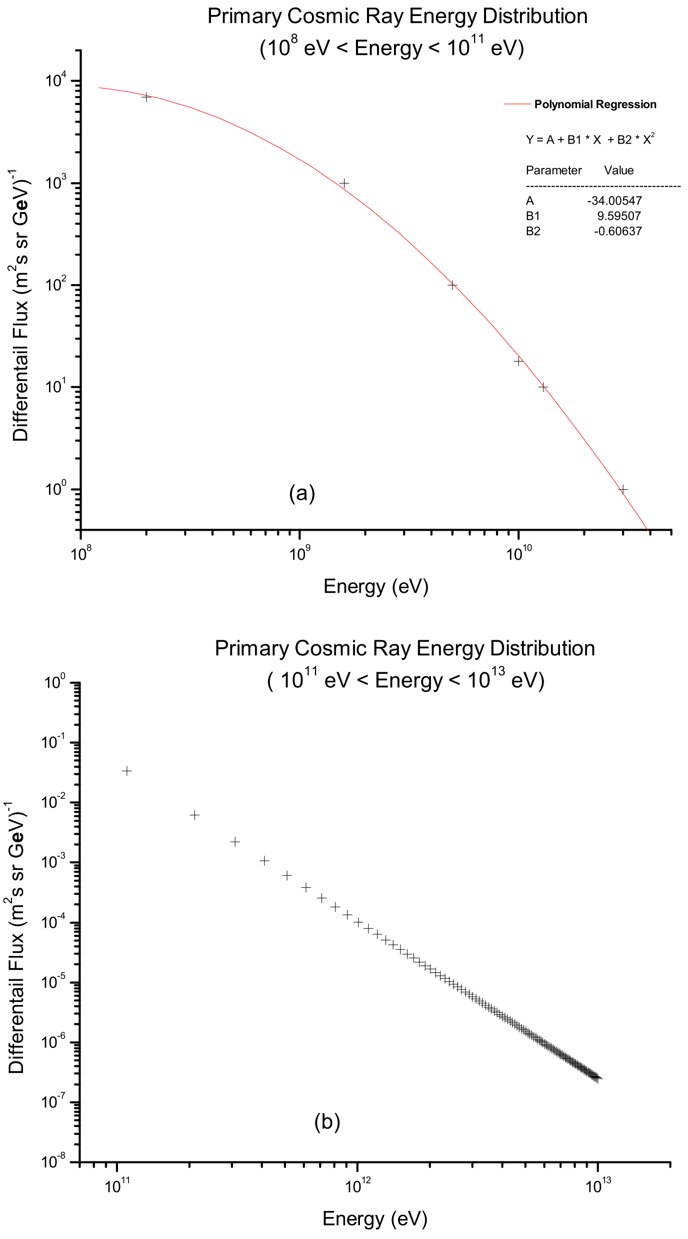

Figure 6.2: Generated primary proton flux distribution (a) for the energy from 10^8 to 10^{11} eV and (b) for the energy from 10^{11} to 10^{13} eV.

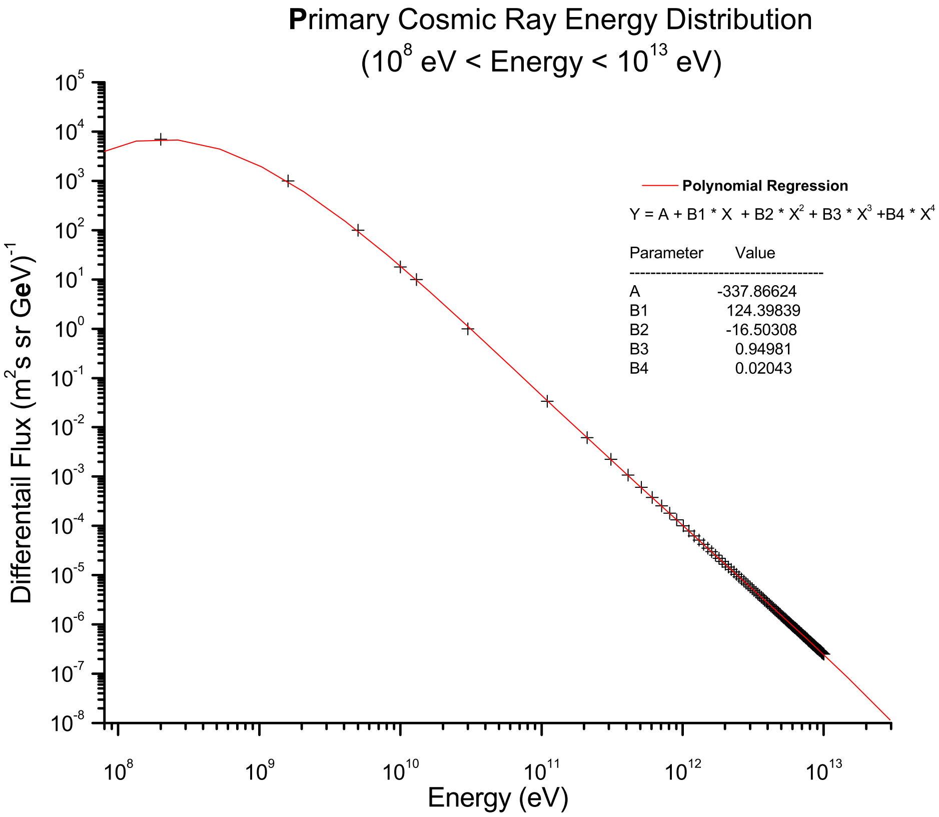

Figure 6.3: Generated primary proton distribution for energy from energy from 10^8 to 10^{13} eV by combining the results shown in Fig. 6.2.

6.3 ECRS Simulation Results: μ^+/μ^- Charge Ratio

Figure 6.4 (a) shows the moun charge ratio as a function of muon momentum obtained from the ECRS simulation with and without geomagnetic field for the Atlanta region. Figure 6.4 (b) shows the similar charge ratio from ECRS for two different geomagnetic locations, Atlanta and Lynn Lake with magnetic field. The vertical cutoff rigidity in Atlanta and Lynn Lake are 3.9 GV and 0.5 GV respectively. In each case, the effect on μ^+ is comparatively stronger than on μ^- which is clearly seen in the muon charge ratio in Fig. 6.4 (a) and (b). In the ECRS simulation, the muon charge ratios with and without geomagnetic field in Atlanta region show clear divergence below 1 GeV/c. Also, the muon charge ratio without magnetic field remains fairly constant except for the higher momentum. In addition, Fig. 6.4 (b) shows similar deviation below 1 GeV/c for the two different geomagnetic locations. According to these two simulation analyzes, low cutoff rigidity locations show higher charge ratio than the high cutoff rigidity locations for low momentum regions. Therefore it can be concluded that the divergence below 1 GeV/c comes from the effect of geomagnetic cutoff rigidity.

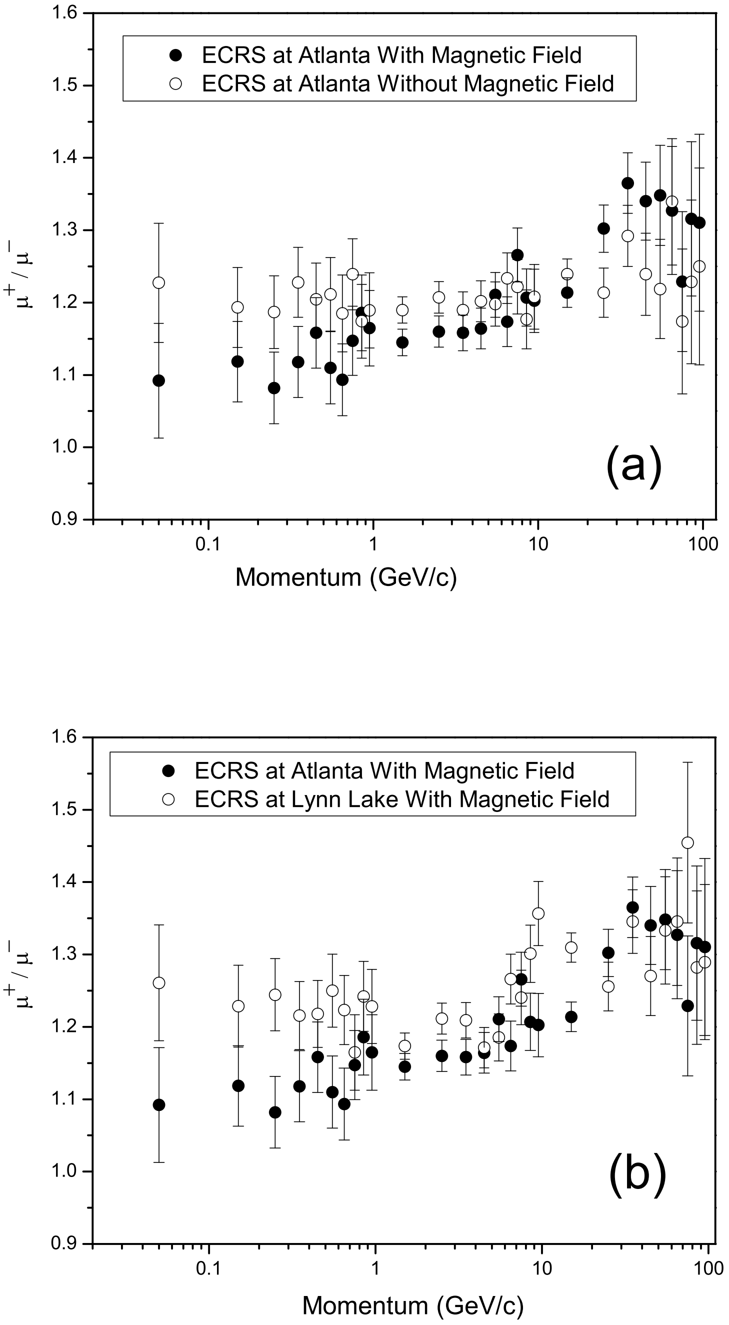

Figure 6.4: **(a)** The muon charge ratio at the Earth's surface as a function of muon momentum from ECRS simulation with and without geomagnetic field in the Atlanta region. **(b)** A comparison of the muon charge ratio at the Earth's surface as a function of muon momentum from ECRS between Atlanta and Lynn Lake.

Similar behaviors have been measured by both the NMSU-WIZARD/CAPRICE and BESS experiments. Figure 6.5 (a) shows the results measured by the CAPRICE spectrometer (J. Kremer et al., 1999) in Lynn Lake (CAPRIC97) compared with ECRS simulation for the same location in the muon momentum range from 0.2 to 100 GeV/c. Figure 6.5 (b) shows the muon charge ratio measured by the BARS spectrometer (M. Motoki et al., 2001), NMSU-WIZARD/CAPRICE and ECRS simulation in Lynn Lake for momentum range from 0.2 to 12.5 GeV/c. The mean values of the charge ratio between 0.1 and 12.5 GeV/c for the two geomagnetic locations are given in Table 6.1 for both experiments and ECRS simulation. The ECRS simulation results show good agreement in muon charge ratio with two sets experimental measurements at Lynn Lake.

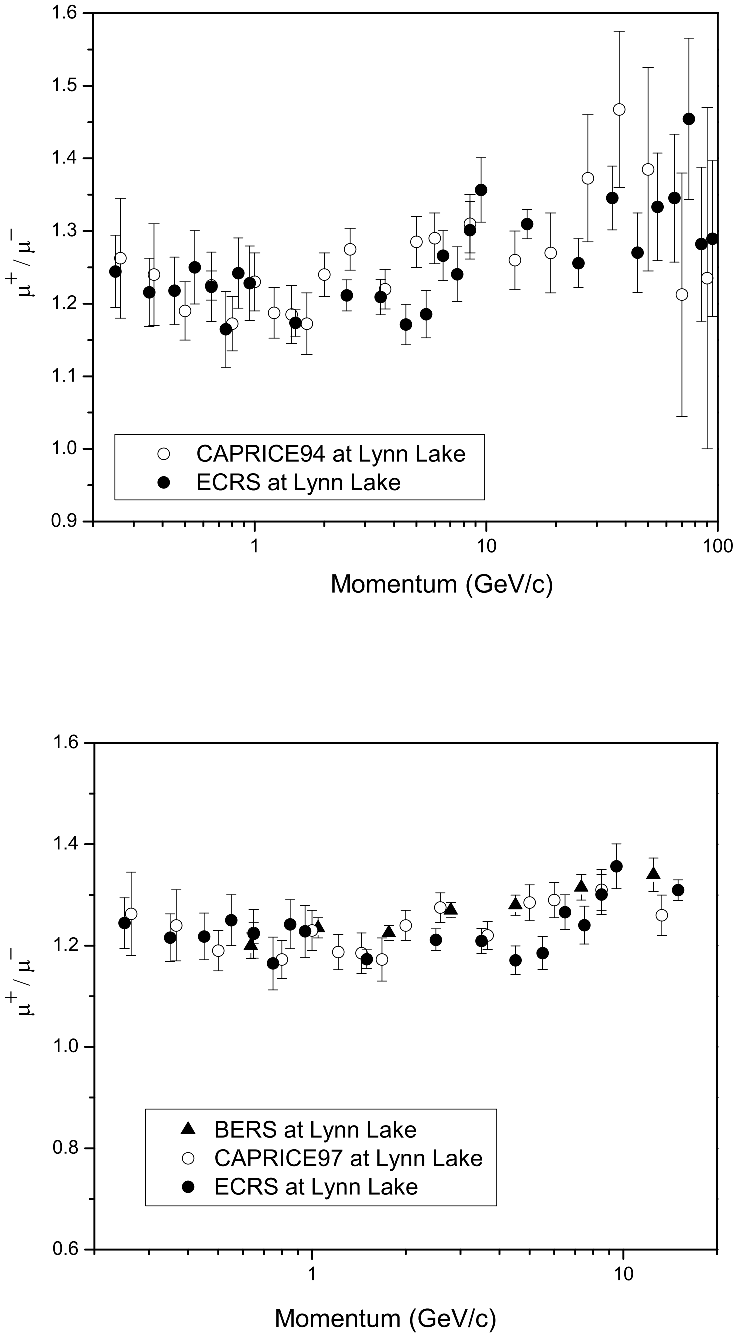

Figure 6.5: **(a)** The muon charge ratio as a function of muon momentum measured by NMSU-WIZARD/CAPRICE magnetic spectrometer (CAPRIC97) and calculated by ECRS simulation at Lynne Lake for momentum range from 0.2 GeV/c to 100 GeV/c. **(b)** The muon charge ratio as a function of muon momentum measured by NMSU-WIZARD/CAPRICE magnet spectrometer (open circle), NMSU-BARS spectrometer (filled triangle) for momentum range from 0.2 GeV/c to 12.5 GeV/c. The filled circle is from ECRS which shows a very good agreement with the experimental data.

Table 6.1: Experimental and simulation results of mean muon charge ratio between 0.1 and 12.5 GeV/c for two different geomagnetic locations.

	Location	Cutoff Rigidity (GV)	μ^+/μ^-
ECRS	Lynn Lake	0.5	1.22 ± 0.04
CAPRICE	Lynn Lake	0.5	1.23 ± 0.04
BESS	Lynn Lake	0.5	1.26 ± 0.02
ECRS	Atlanta	0	1.20 ± 0.04
ECRS	Atlanta	3.9	1.12 ± 0.04

To understand the muon charge ratio, a general theory of the mechanism for the production of hadrons, pionization and fragmentation are discussed below.

Pionization refers to the extensive production of secondary particles at low energy in the center of mass system. In this model assume that the primary particle spectrum consists of protons which interact once with the atmospheric nuclei, producing pions which all decay into muons. The single particle distribution for pions produced in a proton–proton collision ^{78},

$$f^{\pm}_{p\pi}(E_{\pi}, Ep) = \frac{E_{\pi}}{\sigma^{inel}_{pp}} \frac{d\sigma^{\pm}_{p\rightarrow\pi}}{dE_{\pi}} \tag{6.5}$$

where E_p and E_π are the energies of the primary proton and secondary pion, respectively, σ^{inel}_{pp} is the total inelastic proton-proton cross section, and ± refers to the charge of the observed pion.

From the equation 6.5, assume a primary proton spectrum given by,

$$\frac{dN}{dE} = N_0 E^{-(1+\gamma)} \tag{6.6}$$

where N_0 is a constant, and $\gamma \approx 1.7$. Then the pion spectrum in this model is given by,

$$\pi^{\pm}(E_{\pi}) = \frac{dn^{\pm}_{\pi}(E_{\pi})}{dE_{\pi}} = \frac{(const)}{E_{\pi}} \int_{E_{\pi}}^{\infty} dE E^{-(1+\gamma)} f^{\pm}_{p\pi}(E_{\pi}, E) \tag{6.7}$$

Applying the hypothesis of limiting fragmentation,

$$lim_{E\rightarrow\infty} \tilde{f}^{\pm}_{p\pi}(x) \tag{6.8}$$

where

$$x = \sqrt{2} \frac{P_{\pi L}}{[M_N(E + M_N)]^{1/2}} \tag{6.9}$$

is the usual Feynman scaled variable, $P^{c.m}_{\pi L}$ is the pion longitudinal momentum in the center-of-mass frame, and M_N is the nucleon mass. For E, $E_\pi \rightarrow \infty$, x $\approx E_\pi$/E. Equation 6.7 to the form,

$$\pi^{\pm}(E_\pi) = (const)E_\pi^{-(1+\gamma)} Z_{p\pi}^{\pm} \tag{6.10}$$

where

$$Z_{p\pi}^{\pm} = \int_0^1 \tilde{f}_{p\pi}^{\pm}(x) x^{\gamma-1} dx \tag{6.11}$$

Consequently, in this simple model, we can derived,

$$\frac{\mu^+(E_\mu)}{\mu^-(E_\mu)} = \frac{\pi^+(E_\pi)}{\pi^-(E_\pi)} = \frac{Z_{p\pi}^+}{Z_{p\pi}^-} \tag{6.12}$$

From this simple model, one can extract following features for muon charge ratio at the Earth's surface.

- From equation 6.10 – 6.12, the $\frac{\mu^+}{\mu^-}$ ratio is explicitly independent of the muon energy.

- Due to the $\pi^+ > \pi^-$ in the primary proton fragments, $\frac{\mu^+}{\mu^-} > 1$.

- The ratio $\frac{\mu^+}{\mu^-}$ depends on the power γ of the primary spectrum, and thus may have an implicit energy dependance through various in the power law of the primary spectrum.

- Pionization products have very little effects on $\frac{\mu^+}{\mu^-}$ ratio. The power law behavior of the primary spectrum, appearing as the weighting factor $x^{\gamma-1}$ in the integrand $Z_{p\pi}^{\pm}$, suppresses contributions from x ≈ 0, the pionization region. For x > 0 , it is expected that $\tilde{f}_{p\pi}^+ > \tilde{f}_{p\pi}^-$. Since the integrand in $Z_{p\pi}^{\pm}$ never becomes negative, it follows that $Z_{p\pi}^+ > Z_{p\pi}^-$, and that $\frac{\mu^+}{\mu^-} > 1$.

According to the theoretical explanation, the simulated $\frac{\mu^+}{\mu^-}$ is agreed with theoretical predictions .

Chapter 7

Conclusion and Future Studies

7.1 Conclusion

The precise knowledge of the flux of atmospheric particles induced by galactic and solar cosmic rays is important for number of applications. A detailed theoretical model of the shower development is required to analyze experimental data on extensive air shower or to plan future experiments. This can only be achieved by simulations, which take into account all knowledge of high energy hadronic and electromagnetic interactions. ECRS is a useful and flexible tool that can be used to study high energy cosmic ray interactions and support the interpretation of secondary cosmic ray shower measurements.

The ECRS simulation was successfully developed to study secondary cosmic ray showers in the Earth's atmosphere. Table 7.1 summarizes the main characteristics of the ECRS. It employs a number of theoretical models of high energy hadronic interactions as well as experimental data modules. Both internal and external magnetic fields were implemented.

Also, two different methods were used to verify the geomagnetic field as well as cutoff rigidity on the Earth's surface. In this thesis, the analysis of the data from the ECRS simulation was presented with air density effects and magnetic field effects for both lateral and longitudinal distribution. Simulated flux was shown good agreement with Toy's theoretical model.

The CAPRICE and BESS muon data was compared with results from the ECRS simulation that simulated extensive air shower for two different geomantic locations. The results were shown to have good agreement between the measured charged ratio and the ECRS simulated charged ratio, giving geomagnetic cutoff rigidity dependence for low energy muon particles.

7.2 Future Studies

In addition to the above simulation studies, simulated data was collected to study the atmospheric neutrinos. Detailed analysis of neutrinos will be one area of future studies based on the ECRS simulation.

The primary particle fluxes were simulated with different air densities that show an effect on lateral distribution at the Earth's surface. Cloud formation and air composition changes are directly linked with air density changes. Layers of cloud with appropriate composition can be employed in the ECRS to study the correlation between secondary flux and weather. Extensive analysis will be needed with a broad range of primary particle energies to achieve such a kind of experiment. ECRS simulated secondary flux could also be used to study the radiation effect on biological cells, and verify the detector data. The

ECRS simulation can be used to model cosmic ray flux variation due to the solar activity changes in a given period of time. More importantly, this simulation is an extremely useful toolkit for optimizing the design of detectors for cosmic ray studies.

Table 7.1: Main characteristics of the ECRS simulation program.

Main Characteristics of ECRS

Main observable	Energy deposited in the atmosphere, at different observing level. Particle lateral and longitudinal distribution. Number and energy of secondary particles, at different observing level. Detailed list of particles reaching ground, and/or crossing predetermined observing level.
Geometry	Real size of Earth with realistic atmospheric layers. Both internal and external magnetic field.
Primary particle	Mainly proton but any particle can be injected as a primary. Multiple particles can be also injected.
Propagated particle.	γ, $e^{\pm}$, $\mu^{\pm}$, π^0, $\pi^{\pm}$, $K^{\pm}$, K^0, η, the baryons p, n, K^{*0}, $K^{*\pm}$, $\Delta^{\pm}$, Δ^0, Δ^{++} the corresponding anti–baryons, and ν_e, ν_μ
Propagation General	Multiple Scattering. Scattering all charged particle. Ionization. Transition Radiation. Scintillation. Cerenkov Effect.
Propagation Muons	Bremsstrahlung. Decay and emission of knock–on electron. e^- - e^+ Pair Production.
Propagation Hadrons and nuclei	Hadronic cross sections are evaluated from fits to experimental data. Emission of knok–on electron. Decay of unsatiable hadron. QGSP, QGSP BERT, LHEP BERT and LHEP GN models for high energy.
Propagation Elections and gammas	Bremsstrahlung. Decay and emission of knock–on electron. e^- - e^+ Annihilation. Photoelectric and Photonuclear effect. Compton Scattering. Gamma Conversion into e^- - e^+. Emission of knok–on electrons.

Appendix A

Programs Used for ECRS

The following ECRS programs are written in C++. The ECRS program is a collection of classes that includes following classes.

- BEquation
- BIntegrator
- ECRSAtmosphereSD
- ECRSEventAction
- ECRSStackingMessenger
- ECRSVisManager
- ECRSDataArray
- ECRSEventActionMessenger
- ECRSPrimaryGeneratorAction

- ECRSSteppingAction
- ECRSDetectorConstruction
- ECRSFieldMessenger
- ECRSPrimaryGeneratorMessenger
- ECRSSteppingMessenger
- DateAndTime
- ECRSDetectorMessenger
- ECRSMagneticField
- ECRSRunAction
- ECRSTrackingAction
- SpaceCoordinateConvertor
- ECRSAtmoHit
- ECRSEquationOfMotion
- ECRSPhysicsList
- ECRSStackingAction
- ECRSTrackingMessenger

ECRS main program and construction codes are given below. Some main ECRS analysis codes are also included.

```
//ECRS main program

#include "G4UImanager.hh"
#include "G4UIterminal.hh"
#include "G4UIGAG.hh"
#include "G4UItcsh.hh"
#include "G4UIXm.hh"
#include "G4UIXaw.hh"
#include "G4RunManager.hh"
#include "Randomize.hh"
#include <time.h>
#include "ECRSMagneticField.hh"
#include "ECRSDetectorConstruction.hh"
#include "ECRSPhysicsList.hh"
#include "ECRSPrimaryGeneratorAction.hh"
#include "ECRSVisManager.hh"
#include "ECRSRunAction.hh"
#include "ECRSEventAction.hh"
#include "ECRSStackingAction.hh"
#include "ECRSTrackingAction.hh"
#include "ECRSSteppingAction.hh"
#include "ECRSSingleton.hh"

#ifdef G4VIS_USE
#include "ECRSVisManager.hh"
#include "G4OpenGLImmediateX.hh"
#include "G4OpenGLStoredX.hh"
#include "G4OpenGLImmediateXm.hh"
#include "G4OpenGLStoredXm.hh"
#include "G4DAWNFILE.hh"
#endif

#include "G4ios.hh"
#include "ECRSUnits.hh"
#include "G4UnitsTable.hh"
#include "SpaceCoordinateConvertor.hh"
#include "G4ProcessTable.hh"
#include <iostream>
#include <stdlib.h>

int main(int argc,char** argv) {
 // Definition of new units in the unit table should be defined  at
 the beginning    //before the instantiation of the runManager and
 should be followed by
  // G4UnitDefinition::BuildUnitsTable()

  new G4UnitDefinition("earth radii","re","Length",re);
  new G4UnitDefinition("earth radii 1","Re","Length",re);
  new G4UnitDefinition("earth radii 2","RE","Length",re);
  new G4UnitDefinition("hour","hour","Time",3600.*s);
  new G4UnitDefinition("minute","minute","Time",60.*s);
  new G4UnitDefinition("day","day","Time",24.*3600.*s);
  new G4UnitDefinition("nanotesla","nT","Magnetic flux density",nT);
  new G4UnitDefinition("gigavolt","GV","Electric
potential",1000.*megavolt);
```

```
  G4UnitDefinition::BuildUnitsTable();

  srand(time(NULL));
  ECRSSingleton* myOut = ECRSSingleton::instance();
  G4int seed_index;
  seed_index = (int)rand();
  myOut->Fopen("Cosmic_Output.dat");

 // Select the RanecuEngine random number generator with
//seeds defined above
  HepRandom::setTheEngine(new RanecuEngine);
  G4long Myseeds[2];
  Myseeds[0] = (int)rand();
  Myseeds[1] = (int)rand();
  HepRandom::setTheSeeds(Myseeds,seed_index);

  // Run manager
  G4RunManager * runManager = new G4RunManager;

  // Mandatory initialization classes
  ECRSDetectorConstruction* detector = new ECRSDetectorConstruction;
  runManager->SetUserInitialization(detector);
  runManager->SetUserInitialization(new ECRSPhysicsList);

#ifdef G4VIS_USE
  // Visualization, if you choose to have it!
  G4VisManager* visManager = new ECRSVisManager;
  visManager->Initialize();
#endif

  // set mandatory user action class
  runManager->SetUserAction(new
ECRSPrimaryGeneratorAction(detector));
  runManager->SetUserAction(new ECRSRunAction);
  runManager->SetUserAction(new ECRSEventAction);

  // set optional user action classes
  runManager->SetUserAction(new ECRSStackingAction);
  runManager->SetUserAction(new ECRSSteppingAction(detector));
  runManager->SetUserAction(new ECRSTrackingAction);

  G4UIsession* session=0;
  if (argc==1) session = new G4UIterminal;
  //Initialize G4 kernel
  runManager->Initialize();

  G4ProcessTable::GetProcessTable()-
>SetProcessActivation("MYTransportation",false);
  G4ProcessTable::GetProcessTable()-
>SetProcessActivation("Transportation",true);

  // User interactions

  G4UImanager* UI = G4UImanager::GetUIpointer();
  UI->ApplyCommand("/tracking/verbose 0");
  if (session)
    {
      session->SessionStart();
      delete session;
      G4cout << G4endl << "Interactive session ended." << G4endl;
    }
  else
    {
      G4String command = "/control/execute ";
      G4String fileName = argv[1];
      UI->ApplyCommand(command+fileName);
    }

#ifdef G4VIS_USE
  delete visManager;
#endif
  delete runManager;
  G4cout << "RunManager deleted." << G4endl;
  myOut->Fclose();
  return 0;
}
```

```
// ECRS Earth Construction Class

#include "ECRSDetectorConstruction.hh"
#include "ECRSDetectorMessenger.hh"
#include "ECRSAtmosphereSD.hh"
#include "ECRSMagneticField.hh"
#include "G4Box.hh"
#include "G4Sphere.hh"
#include "G4PVPlacement.hh"
#include "G4VisAttributes.hh"
#include "G4UnitsTable.hh"
#include "G4ThreeVector.hh"
#include "G4LogicalVolume.hh"
#include "G4RunManager.hh"
#include "G4ios.hh"
#include "G4SDManager.hh"
#include "G4MaterialTable.hh"

ECRSDetectorConstruction::ECRSDetectorConstruction()
  :air1(NULL),air2(NULL),air3(NULL),air4(NULL),air5(NULL),
   air6(NULL),air7(NULL),air8(NULL),air9(NULL),air10(NULL),
  air11(NULL),air12(NULL),air13(NULL),air14(NULL),air15(NULL),
  air16(NULL),air17(NULL),air18(NULL),air19(NULL),air20(NULL),
  air21(NULL),air22(NULL),air23(NULL),air24(NULL),air25(NULL),
  air26(NULL),air27(NULL),air28(NULL),air29(NULL),air30(NULL),
  air31(NULL),air32(NULL),air33(NULL),air34(NULL),air35(NULL),
  air36(NULL),air37(NULL),air38(NULL),

Si(NULL),space(NULL),elN(NULL),elO(NULL),elAr(NULL),earth(NULL),
elC(NULL),N2(NULL),O2(NULL),Ar(NULL),CO2(NULL),universe(NULL),
universe_log(NULL),universe_phys(NULL),ECRS_sphere(NULL),
ECRS_log(NULL),ECRS_phys(NULL),

atmo1(NULL),atmo2(NULL),atmo3(NULL),atmo4(NULL),atmo5(NULL),
atmo6(NULL),atmo7(NULL),atmo8(NULL),atmo9(NULL),atmo10(NULL),
atmo11(NULL),atmo12(NULL),atmo13(NULL),atmo14(NULL),atmo15(NULL),
atmo16(NULL),atmo17(NULL),atmo18(NULL),atmo19(NULL),atmo20(NULL),
atmo21(NULL),atmo22(NULL),atmo23(NULL),atmo24(NULL),atmo25(NULL),
atmo26(NULL),atmo27(NULL),atmo28(NULL),atmo29(NULL),atmo30(NULL),
atmo31(NULL),atmo32(NULL),atmo33(NULL),atmo34(NULL),atmo35(NULL),
atmo36(NULL),atmo37(NULL),atmo38(NULL),

atm1_log(NULL),atm2_log(NULL),atm3_log(NULL),atm4_log(NULL),
atm5_log(NULL),atm6_log(NULL),atm7_log(NULL),atm8_log(NULL),
atm9_log(NULL),atm10_log(NULL),atm11_log(NULL),atm12_log(NULL),
atm13_log(NULL),atm14_log(NULL),atm15_log(NULL),atm16_log(NULL),
atm17_log(NULL),atm18_log(NULL),atm19_log(NULL),atm20_log(NULL),
atm21_log(NULL),atm22_log(NULL),atm23_log(NULL),atm24_log(NULL),
atm25_log(NULL),atm26_log(NULL),atm27_log(NULL),atm28_log(NULL),
atm29_log(NULL),atm30_log(NULL),atm31_log(NULL),atm32_log(NULL),
atm33_log(NULL),atm34_log(NULL),atm35_log(NULL),atm36_log(NULL),
atm37_log(NULL),atm38_log(NULL),

atm1_phys(NULL),atm2_phys(NULL),atm3_phys(NULL),atm4_phys(NULL),
atm5_phys(NULL),atm6_phys(NULL),atm7_phys(NULL),atm8_phys(NULL),
atm9_phys(NULL),atm10_phys(NULL),atm11_phys(NULL),atm12_phys(NULL),
atm13_phys(NULL),atm14_phys(NULL),atm15_phys(NULL),atm16_phys(NULL),
atm17_phys(NULL),atm18_phys(NULL),atm19_phys(NULL),atm20_phys(NULL),
atm21_phys(NULL),atm22_phys(NULL),atm23_phys(NULL),atm24_phys(NULL),
atm25_phys(NULL),atm26_phys(NULL),atm27_phys(NULL),atm28_phys(NULL),
atm29_phys(NULL),atm30_phys(NULL),atm31_phys(NULL),atm32_phys(NULL),
atm33_phys(NULL),atm34_phys(NULL),atm35_phys(NULL),atm36_phys(NULL),
atm37_phys(NULL),atm38_phys(NULL),

Atm1Att(NULL),Atm2Att(NULL),Atm3Att(NULL),Atm4Att(NULL),Atm5Att(NULL)
,
Atm6Att(NULL),Atm7Att(NULL),Atm8Att(NULL),Atm9Att(NULL),Atm10Att(NULL
),
Atm11Att(NULL),Atm12Att(NULL),Atm13Att(NULL),Atm14Att(NULL),Atm15Att(
NULL),
Atm16Att(NULL),Atm17Att(NULL),Atm18Att(NULL),Atm19Att(NULL),Atm20Att(
NULL),
Atm21Att(NULL),Atm22Att(NULL),Atm23Att(NULL),Atm24Att(NULL),Atm25Att(
NULL),
Atm26Att(NULL),Atm27Att(NULL),Atm28Att(NULL),Atm29Att(NULL),Atm30Att(
NULL),
Atm31Att(NULL),Atm32Att(NULL),Atm33Att(NULL),Atm34Att(NULL),Atm35Att(
NULL),
Atm36Att(NULL),Atm37Att(NULL),Atm38Att(NULL),

   atmosphereSD(0)
{
  mult = 1.0;
  multDensity = 1.0;
  visibility = false;
  externalMag = true;

  // Set detector sizes
  universeSize = 19.1136e6*m;
  universe_x = universeSize;
  universe_y = universeSize;
  universe_z = universeSize;

  // set the radius of the Earth
  earthRadius = 6.3712e6*m;

  // set atmospheric layers height - Modified Sanjeewa sep/05
  G4double Height = 0.0000e3*m;

  // set 30 layers each 1 km interval from 0 - 30 km
  for (G4int i = 0 ; i < 31; i++)
    {
      atmHeight[i] = Height;
      Height  = Height + 1.0000e3*m;
    }

  // Set 7 layers each 10 km interval from 40 - 100 km
  for (G4int i = 31; i < 38; i++)
    {
      atmHeight[i] = atmHeight[i-1] + 10.000e3*m;
    }

  ECRSDetector = new ECRSDetectorMessenger(this);
  if (  externalMag = false )
    {
      G4cout << "External Magnetic Firld off " <<  externalMag <<
G4endl;
    }

  if ( externalMag = true){
    theMagneticField=new ECRSMagneticField();
    G4cout << "External Magnetic Firld on " <<  externalMag <<
G4endl;
  }
}

/////////////////////////////////////////////////////////////////

ECRSDetectorConstruction::~ECRSDetectorConstruction()
{
  delete ECRSDetector;
  delete theMagneticField;
}

/////////////////////////////////////////////////////////////////

G4VPhysicalVolume* ECRSDetectorConstruction::Construct()
{
  DefineMaterials();
  return ConstructWorld();
}

/////////////////////////////////////////////////////////////////

void ECRSDetectorConstruction::DefineMaterials()
{
  G4double a;  // atomic mass
  G4double z;  // atomic number

  //Define the ECRS to be made of solid silicone
  a = 28.086*g/mole;
  density = 2.4*g/cm3;
  //  density *= multDensity;
  Si = new G4Material(name="Silicone",z=14., a, density);

   //Define elements and gasses to compose atmosphere
  a = 14.007*g/mole;
  elN = new G4Element(name="Nitrogen",symbol="N",z=7.,a);

  a = 15.999*g/mole;
  elO = new G4Element(name="Oxygen",symbol="O",z=8.,a);

  a = 39.948*g/mole;
  elAr = new G4Element(name="Argon",symbol="Ar",z=18.,a);
  a = 12.011*g/mole;
  elC = new G4Element(name="Carbon",symbol="C",z=6.,a);

  density = 0.001251*g/cm3;
   N2 = new G4Material(name="n_gas",density,ncomponents=1);
  N2->AddElement(elN,natoms=2);

  density = 0.001429*g/cm3;
  O2 = new G4Material(name="o_gas",density,ncomponents=1);
  O2->AddElement(elO,natoms=2);

  density = 0.001784*g/cm3;
  Ar = new G4Material(name="Ar_gas",density,ncomponents=1);
  Ar->AddElement(elAr,natoms=1);

  density = 0.001965*g/cm3;
  CO2 = new G4Material(name="carbon_dioxide",density,ncomponents=2);
  CO2->AddElement(elC,natoms=1);
  CO2->AddElement(elO,natoms=2);

  //Make the air layer Density, Temperatur, and pressure array
  G4double DensityPro[38] = {0.000*g/cm3,
                             1.1673e-3*g/cm3, 1.0582e-3*g/cm3,
9.5695e-4*g/cm3, 8.6340e-4*g/cm3, 7.7704e-4*g/cm3, 6.9747e-4*g/cm3,
6.6431e-4*g/cm3, 5.5719e-4*g/cm3, 4.9576e-4*g/cm3, 4.3966e-4*g/cm3,
3.8857e-4*g/cm3, 3.3743e-4*g/cm3, 2.8838e-4*g/cm3, 2.4646e-4*g/cm3,
2.1066e-4*g/cm3, 1.6647e-4*g/cm3, 1.4230e-4*g/cm3, 1.2165e-4*g/cm3,
1.0400e-4*g/cm3, 8.8910e-5*g/cm3, 7.5715e-5*g/cm3, 6.4510e-5*g/cm3,
5.5006e-5*g/cm3, 4.6938e-5*g/cm3, 4.0084e-5*g/cm3, 3.4257e-5*g/cm3,
2.9298e-5*g/cm3, 2.5076e-5*g/cm3, 2.1478e-5*g/cm3, 1.8410e-5*g/cm3,
8.4634e-6*g/cm3 ,1.7142e-6*g/cm3, 5.0445e-7*g/cm3, 1.6321e-7*g/cm3,
3.9921e-8*g/cm3, 8.2196e-9*g/cm3, 1.3930e-9*g/cm3 };

  G4double TempPro[38] =    {0.000*kelvin, 284.900*kelvin,
278.402*kelvin, 271.906*kelvin, 265.413*kelvin, 258.921*kelvin,
252.432*kelvin, 245.943*kelvin, 239.457*kelvin, 232.974*kelvin,
226.942*kelvin, 220.013*kelvin, 216.650*kelvin, 216.650*kelvin,
216.650*kelvin, 216.650*kelvin, 216.650*kelvin, 216.650*kelvin,
216.650*kelvin, 216.650*kelvin, 216.650*kelvin, 217.581*kelvin,
218.574*kelvin, 219.567*kelvin, 220.560*kelvin, 221.552*kelvin,
222.554*kelvin, 223.536*kelvin, 224.527*kelvin, 225.518*kelvin,
226.509*kelvin, 236.513*kelvin, 266.925*kelvin, 258.019*kelvin,
233.292*kelvin, 208.399*kelvin, 188.893*kelvin, 188.420*kelvin };

  G4double PressurePro[38] ={0.000*pascal, 9.5461e+4*pascal,
8.9876e+4*pascal, 7.4691e+4*pascal, 6.5780e+4*pascal,
5.7752e+4*pascal, 5.0593e+4*pascal, 4.4075e+4*pascal,
3.8299e+4*pascal, 3.3154e+4*pascal, 2.8584e+4*pascal,
2.4540e+4*pascal, 2.0984e+4*pascal, 1.7934e+4*pascal,
1.5327e+4*pascal, 1.3100e+4*pascal, 1.0352e+4*pascal,
8.8497e+3*pascal, 7.5652e+3*pascal, 6.4674e+3*pascal,
5.5293e+3*pascal, 4.7289e+3*pascal, 4.0475e+3*pascal,
3.4668e+3*pascal, 2.9717e+3*pascal, 2.5492e+3*pascal,
2.1883e+3*pascal, 1.8799e+3*pascal, 1.6161e+3*pascal,
1.3904e+3*pascal, 1.1970e+3*pascal, 5.5749e+2*pascal,
1.3134e+2*pascal, 3.7362e+1*pascal,n1.0929e+1*pascal, 2.3881*pascal,
4.4568e-1*pascal, 7.5966e-2*pascal};

  //Make the air of the atmosphere
  density = DensityPro[1];    // 500m in CRC
  density *= multDensity;
  temperature = TempPro[1];
  pressure = PressurePro[1];
  air1 = new G4Material(name="air-1",density,ncomponents=4,
                kStateGas,temperature);
  air1->AddMaterial(N2,fractionmass=75.521*perCent);
  air1->AddMaterial(O2,fractionmass=23.143*perCent);
  air1->AddMaterial(Ar,fractionmass=1.288*perCent);
  air1->AddMaterial(CO2,fractionmass=0.048*perCent);

  density =  DensityPro[2];     // 1500m in CRC
  density *= multDensity;
  pressure = TempPro[2];
  temperature = PressurePro[2];
  air2 = new G4Material(name="air-2",density,ncomponents=4,
                kStateGas,temperature,pressure);
  air2->AddMaterial(N2,fractionmass=75.521*perCent);
  air2->AddMaterial(O2,fractionmass=23.143*perCent);
  air2->AddMaterial(Ar,fractionmass=1.288*perCent);
  air2->AddMaterial(CO2,fractionmass=0.048*perCent);

  density =  DensityPro[3];      // 2500m in CRC
  density *= multDensity;
  temperature = TempPro[3];
  pressure = PressurePro[3];
  air3 = new G4Material(name="air-3",density,ncomponents=4,
                kStateGas,temperature,pressure);
  air3->AddMaterial(N2,fractionmass=75.521*perCent);
  air3->AddMaterial(O2,fractionmass=23.143*perCent);
  air3->AddMaterial(Ar,fractionmass=1.288*perCent);
  air3->AddMaterial(CO2,fractionmass=0.048*perCent);

  density = DensityPro[4];      // 3500m in CRC
  density *= multDensity;
  temperature = TempPro[4];
  pressure = PressurePro[4];
  air4 = new G4Material(name="air-4",density,ncomponents=4,
                kStateGas,temperature,pressure);
  air4->AddMaterial(N2,fractionmass=75.521*perCent);
  air4->AddMaterial(O2,fractionmass=23.143*perCent);
  air4->AddMaterial(Ar,fractionmass=1.288*perCent);
  air4->AddMaterial(CO2,fractionmass=0.048*perCent);

  density = DensityPro[5];      // 4500m in CRC
  density *= multDensity;
  temperature = TempPro[5];
  pressure = PressurePro[5];
  air5 = new G4Material(name="air-5",density,ncomponents=4,
                kStateGas,temperature,pressure);
  air5->AddMaterial(N2,fractionmass=75.521*perCent);
```

```
air5->AddMaterial(O2,fractionmass=23.143*perCent);
air5->AddMaterial(Ar,fractionmass=1.288*perCent);
air5->AddMaterial(CO2,fractionmass=0.048*perCent);

density = DensityPro[6];      // 5500m in CRC
density *= multDensity;
temperature = TempPro[6];
pressure = PressurePro[6];
air6 = new G4Material(name="air-6",density,ncomponents=4,
                kStateGas,temperature,pressure);
air6->AddMaterial(N2,fractionmass=75.521*perCent);
air6->AddMaterial(O2,fractionmass=23.143*perCent);
air6->AddMaterial(Ar,fractionmass=1.288*perCent);
air6->AddMaterial(CO2,fractionmass=0.048*perCent);

density = DensityPro[7];      // 6500m in CRC
density *= multDensity;
temperature = TempPro[7];
pressure = PressurePro[7];
air7 = new G4Material(name="air-7",density,ncomponents=4,
                kStateGas,temperature,pressure);
air7->AddMaterial(N2,fractionmass=75.521*perCent);
air7->AddMaterial(O2,fractionmass=23.143*perCent);
air7->AddMaterial(Ar,fractionmass=1.288*perCent);
air7->AddMaterial(CO2,fractionmass=0.048*perCent);

density = DensityPro[8];      // 7500m in CRC
density *= multDensity;
temperature = TempPro[8];
pressure = PressurePro[8];
air8 = new G4Material(name="air-8",density,ncomponents=4,
                kStateGas,temperature,pressure);
air8->AddMaterial(N2,fractionmass=75.521*perCent);
air8->AddMaterial(O2,fractionmass=23.143*perCent);
air8->AddMaterial(Ar,fractionmass=1.288*perCent);
air8->AddMaterial(CO2,fractionmass=0.048*perCent);

density =  DensityPro[9];      // 8500m in CRC
density *= multDensity;
temperature = TempPro[9];
pressure = PressurePro[9];
air9 = new G4Material(name="air-9",density,ncomponents=4,
                kStateGas,temperature,pressure);
air9->AddMaterial(N2,fractionmass=75.521*perCent);
air9->AddMaterial(O2,fractionmass=23.143*perCent);
air9->AddMaterial(Ar,fractionmass=1.288*perCent);
air9->AddMaterial(CO2,fractionmass=0.048*perCent);

density = DensityPro[10];      // 9500m in CRC
density *= multDensity;
temperature = TempPro[10];
pressure = PressurePro[10];
air10 = new G4Material(name="air-10",density,ncomponents=4,
                 kStateGas,temperature,pressure);
air10->AddMaterial(N2,fractionmass=75.521*perCent);
```

```
air10->AddMaterial(O2,fractionmass=23.143*perCent);
air10->AddMaterial(Ar,fractionmass=1.288*perCent);
air10->AddMaterial(CO2,fractionmass=0.048*perCent);

density = DensityPro[11];      // 10500m in CRC
density *= multDensity;
temperature = TempPro[11];
pressure = PressurePro[11];
air11 = new G4Material(name="air-11",density,ncomponents=4,
                 kStateGas,temperature,pressure);
air11->AddMaterial(N2,fractionmass=75.521*perCent);
air11->AddMaterial(O2,fractionmass=23.143*perCent);
air11->AddMaterial(Ar,fractionmass=1.288*perCent);
air11->AddMaterial(CO2,fractionmass=0.048*perCent);

density = DensityPro[12];      // 11500m in CRC
density *= multDensity;
temperature = TempPro[12];
pressure = PressurePro[12];
air12 = new G4Material(name="air-12",density,ncomponents=4,
                 kStateGas,temperature,pressure);
air12->AddMaterial(N2,fractionmass=75.521*perCent);
air12->AddMaterial(O2,fractionmass=23.143*perCent);
air12->AddMaterial(Ar,fractionmass=1.288*perCent);
air12->AddMaterial(CO2,fractionmass=0.048*perCent);

density = DensityPro[13];      // 12500m in CRC
density *= multDensity;
temperature = TempPro[13];
pressure = PressurePro[13];
air13 = new G4Material(name="air-13",density,ncomponents=4,
                 kStateGas,temperature,pressure);
air13->AddMaterial(N2,fractionmass=75.521*perCent);
air13->AddMaterial(O2,fractionmass=23.143*perCent);
air13->AddMaterial(Ar,fractionmass=1.288*perCent);
air13->AddMaterial(CO2,fractionmass=0.048*perCent);

density = DensityPro[14];      // 13500m in CRC
density *= multDensity;
temperature = TempPro[14];
pressure = PressurePro[14];
air14 = new G4Material(name="air-14",density,ncomponents=4,
                 kStateGas,temperature,pressure);
air14->AddMaterial(N2,fractionmass=75.521*perCent);
air14->AddMaterial(O2,fractionmass=23.143*perCent);
air14->AddMaterial(Ar,fractionmass=1.288*perCent);
air14->AddMaterial(CO2,fractionmass=0.048*perCent);

density = DensityPro[15];      // 14500m in CRC
density *= multDensity;
temperature = TempPro[15];
pressure = PressurePro[15];
air15 = new G4Material(name="air-15",density,ncomponents=4,
                 kStateGas,temperature,pressure);
air15->AddMaterial(N2,fractionmass=75.521*perCent);
```

```
air15->AddMaterial(O2,fractionmass=23.143*perCent);
air15->AddMaterial(Ar,fractionmass=1.288*perCent);
air15->AddMaterial(CO2,fractionmass=0.048*perCent);

density =  DensityPro[16];      // 16000m in CRC
density *= multDensity;
temperature = TempPro[16];
pressure = PressurePro[16];
air16 = new G4Material(name="air-16",density,ncomponents=4,
                 kStateGas,temperature,pressure);
air16->AddMaterial(N2,fractionmass=75.521*perCent);
air16->AddMaterial(O2,fractionmass=23.143*perCent);
air16->AddMaterial(Ar,fractionmass=1.288*perCent);
air16->AddMaterial(CO2,fractionmass=0.048*perCent);

density = DensityPro[17];      // 17000m in CRC
density *= multDensity;
temperature = TempPro[17];
pressure = PressurePro[17];
air17 = new G4Material(name="air-17",density,ncomponents=4,
                 kStateGas,temperature,pressure);
air17->AddMaterial(N2,fractionmass=75.521*perCent);
air17->AddMaterial(O2,fractionmass=23.143*perCent);
air17->AddMaterial(Ar,fractionmass=1.288*perCent);
air17->AddMaterial(CO2,fractionmass=0.048*perCent);

density = DensityPro[18];      // 18000m in CRC
density *= multDensity;
temperature = TempPro[18];
pressure = PressurePro[18];
air18 = new G4Material(name="air-18",density,ncomponents=4,
                 kStateGas,temperature,pressure);
air18->AddMaterial(N2,fractionmass=75.521*perCent);
air18->AddMaterial(O2,fractionmass=23.143*perCent);
air18->AddMaterial(Ar,fractionmass=1.288*perCent);
air18->AddMaterial(CO2,fractionmass=0.048*perCent);

density = DensityPro[19];      // 19000m in CRC
density *= multDensity;
temperature = TempPro[19];
pressure = PressurePro[19];
air19 = new G4Material(name="air-19",density,ncomponents=4,
                 kStateGas,temperature,pressure);
air19->AddMaterial(N2,fractionmass=75.521*perCent);
air19->AddMaterial(O2,fractionmass=23.143*perCent);
air19->AddMaterial(Ar,fractionmass=1.288*perCent);
air19->AddMaterial(CO2,fractionmass=0.048*perCent);

density = DensityPro[20];      // 20000m in CRC
density *= multDensity;
temperature = TempPro[20];
pressure = PressurePro[20];
air20 = new G4Material(name="air-20",density,ncomponents=4,
                 kStateGas,temperature,pressure);
air20->AddMaterial(N2,fractionmass=75.521*perCent);
```

```
air20->AddMaterial(O2,fractionmass=23.143*perCent);
air20->AddMaterial(Ar,fractionmass=1.288*perCent);
air20->AddMaterial(CO2,fractionmass=0.048*perCent);

density = DensityPro[21];      // 21000m in CRC
density *= multDensity;
temperature = TempPro[21];
pressure = PressurePro[21];
air21 = new G4Material(name="air-21",density,ncomponents=4,
                 kStateGas,temperature,pressure);
air21->AddMaterial(N2,fractionmass=75.521*perCent);
air21->AddMaterial(O2,fractionmass=23.143*perCent);
air21->AddMaterial(Ar,fractionmass=1.288*perCent);
air21->AddMaterial(CO2,fractionmass=0.048*perCent);

density = DensityPro[22];      // 22000m in CRC
density *= multDensity;
temperature = TempPro[22];
pressure = PressurePro[22];
air22 = new G4Material(name="air-22",density,ncomponents=4,
                 kStateGas,temperature,pressure);
air22->AddMaterial(N2,fractionmass=75.521*perCent);
air22->AddMaterial(O2,fractionmass=23.143*perCent);
air22->AddMaterial(Ar,fractionmass=1.288*perCent);
air22->AddMaterial(CO2,fractionmass=0.048*perCent);

density = DensityPro[23];      // 23000m in CRC
density *= multDensity;
temperature = TempPro[23];
pressure = PressurePro[23];
air23 = new G4Material(name="air-23",density,ncomponents=4,
                 kStateGas,temperature,pressure);
air23->AddMaterial(N2,fractionmass=75.521*perCent);
air23->AddMaterial(O2,fractionmass=23.143*perCent);
air23->AddMaterial(Ar,fractionmass=1.288*perCent);
air23->AddMaterial(CO2,fractionmass=0.048*perCent);

density = DensityPro[24];      // 24000m in CRC
density *= multDensity;
temperature = TempPro[24];
pressure = PressurePro[24];
air24 = new G4Material(name="air-24",density,ncomponents=4,
                 kStateGas,temperature,pressure);
air24->AddMaterial(N2,fractionmass=75.521*perCent);
air24->AddMaterial(O2,fractionmass=23.143*perCent);
air24->AddMaterial(Ar,fractionmass=1.288*perCent);
air24->AddMaterial(CO2,fractionmass=0.048*perCent);

density = DensityPro[25];      // 25000m in CRC
density *= multDensity;
temperature = TempPro[25];
pressure = PressurePro[25];
air25 = new G4Material(name="air-25",density,ncomponents=4,
                 kStateGas,temperature,pressure);
air25->AddMaterial(N2,fractionmass=75.521*perCent);
```

```
air25->AddMaterial(O2,fractionmass=23.143*perCent);
air25->AddMaterial(Ar,fractionmass=1.288*perCent);
air25->AddMaterial(CO2,fractionmass=0.048*perCent);

density = DensityPro[26];      // 26000m in CRC
density *= multDensity;
temperature = TempPro[26];
pressure = PressurePro[26];
air26 = new G4Material(name="air-26",density,ncomponents=4,
                 kStateGas,temperature,pressure);
air26->AddMaterial(N2,fractionmass=75.521*perCent);
air26->AddMaterial(O2,fractionmass=23.143*perCent);
air26->AddMaterial(Ar,fractionmass=1.288*perCent);
air26->AddMaterial(CO2,fractionmass=0.048*perCent);

density = DensityPro[27];      // 27000m in CRC
density *= multDensity;
temperature = TempPro[27];
pressure = PressurePro[27];
air27 = new G4Material(name="air-27",density,ncomponents=4,
                 kStateGas,temperature,pressure);
air27->AddMaterial(N2,fractionmass=75.521*perCent);
air27->AddMaterial(O2,fractionmass=23.143*perCent);
air27->AddMaterial(Ar,fractionmass=1.288*perCent);
air27->AddMaterial(CO2,fractionmass=0.048*perCent);

density = DensityPro[28];      // 28000m in CRC
density *= multDensity;
temperature = TempPro[28];
pressure = PressurePro[28];
air28 = new G4Material(name="air-28",density,ncomponents=4,
                 kStateGas,temperature,pressure);
air28->AddMaterial(N2,fractionmass=75.521*perCent);
air28->AddMaterial(O2,fractionmass=23.143*perCent);
air28->AddMaterial(Ar,fractionmass=1.288*perCent);
air28->AddMaterial(CO2,fractionmass=0.048*perCent);

density = DensityPro[29];      // 29000m in CRC
density *= multDensity;
temperature = TempPro[29];
pressure = PressurePro[29];
air29 = new G4Material(name="air-29",density,ncomponents=4,
                 kStateGas,temperature,pressure);
air29->AddMaterial(N2,fractionmass=75.521*perCent);
air29->AddMaterial(O2,fractionmass=23.143*perCent);
air29->AddMaterial(Ar,fractionmass=1.288*perCent);
air29->AddMaterial(CO2,fractionmass=0.048*perCent);

density = DensityPro[30];      // 30000m in CRC
density *= multDensity;
temperature = TempPro[30];
pressure = PressurePro[30];
air30 = new G4Material(name="air-30",density,ncomponents=4,
                 kStateGas,temperature,pressure);
air30->AddMaterial(N2,fractionmass=75.521*perCent);
```

```
air30->AddMaterial(O2,fractionmass=23.143*perCent);
air30->AddMaterial(Ar,fractionmass=1.288*perCent);
air30->AddMaterial(CO2,fractionmass=0.048*perCent);

density = DensityPro[31];      // 36000m in CRC
density *= multDensity;
temperature = TempPro[31];
pressure = PressurePro[31];
air31 = new G4Material(name="air-31",density,ncomponents=4,
                 kStateGas,temperature,pressure);
air31->AddMaterial(N2,fractionmass=75.521*perCent);
air31->AddMaterial(O2,fractionmass=23.143*perCent);
air31->AddMaterial(Ar,fractionmass=1.288*perCent);
air31->AddMaterial(CO2,fractionmass=0.048*perCent);

density = DensityPro[32];      // 46000m in CRC
density *= multDensity;
temperature = TempPro[32];
pressure = PressurePro[32];
air32 = new G4Material(name="air-32",density,ncomponents=4,
                 kStateGas,temperature,pressure);
air32->AddMaterial(N2,fractionmass=75.521*perCent);
air32->AddMaterial(O2,fractionmass=23.143*perCent);
air32->AddMaterial(Ar,fractionmass=1.288*perCent);
air32->AddMaterial(CO2,fractionmass=0.048*perCent);

density = DensityPro[33];      // 56000m in CRC
density *= multDensity;
temperature = TempPro[33];
pressure = PressurePro[33];
air33 = new G4Material(name="air-33",density,ncomponents=4,
                 kStateGas,temperature,pressure);
air33->AddMaterial(N2,fractionmass=75.521*perCent);
air33->AddMaterial(O2,fractionmass=23.143*perCent);
air33->AddMaterial(Ar,fractionmass=1.288*perCent);
air33->AddMaterial(CO2,fractionmass=0.048*perCent);

density = DensityPro[34];      // 65000m in CRC
density *= multDensity;
temperature = TempPro[34];
pressure = PressurePro[34];
air34 = new G4Material(name="air-34",density,ncomponents=4,
                 kStateGas,temperature,pressure);
air34->AddMaterial(N2,fractionmass=75.521*perCent);
air34->AddMaterial(O2,fractionmass=23.143*perCent);
air34->AddMaterial(Ar,fractionmass=1.288*perCent);
air34->AddMaterial(CO2,fractionmass=0.048*perCent);

density = DensityPro[35];      // 75000m in CRC
density *= multDensity;
temperature = TempPro[35];
pressure = PressurePro[35];
air35 = new G4Material(name="air-35",density,ncomponents=4,
                 kStateGas,temperature,pressure);
air35->AddMaterial(N2,fractionmass=75.521*perCent);
```

```
  air35->AddMaterial(O2,fractionmass=23.143*perCent);
  air35->AddMaterial(Ar,fractionmass=1.288*perCent);
  air35->AddMaterial(CO2,fractionmass=0.048*perCent);

  density = DensityPro[36];      // 85000m in CRC
  density *= multDensity;
  temperature = TempPro[36];
  pressure = PressurePro[36];
  air36 = new G4Material(name="air-36",density,ncomponents=4,
                   kStateGas,temperature,pressure);
  air36->AddMaterial(N2,fractionmass=75.521*perCent);
  air36->AddMaterial(O2,fractionmass=23.143*perCent);
  air36->AddMaterial(Ar,fractionmass=1.288*perCent);
  air36->AddMaterial(CO2,fractionmass=0.048*perCent);

  density = DensityPro[37];      // 95000m in CRC
  density *= multDensity;
  temperature = TempPro[37];
  pressure = PressurePro[37];
  air37 = new G4Material(name="air-37",density,ncomponents=4,
                   kStateGas,temperature,pressure);
  air37->AddMaterial(N2,fractionmass=75.521*perCent);
  air37->AddMaterial(O2,fractionmass=23.143*perCent);
  air37->AddMaterial(Ar,fractionmass=1.288*perCent);
  air37->AddMaterial(CO2,fractionmass=0.048*perCent);

  //Make the vacuum of space
  density = universe_mean_density;  //included from
PhysicalConstants.h
  pressure = 1.0E-19*pascal;
  temperature = 2.74*kelvin;
  space = new G4Material(name="space",z=1.,a=1.01*g/mole,
                   density,kStateGas,temperature,pressure);

  //Make the Univers
  density = .0012250*g/cm3;  //included from PhysicalConstants.h
  pressure = 1.01325E5*pascal;
  temperature = 288.150*kelvin;
  earth = new G4Material(name="earth",density,ncomponents=4,
                   kStateGas,temperature,pressure);
  earth->AddMaterial(N2,fractionmass=75.521*perCent);
  earth->AddMaterial(O2,fractionmass=23.143*perCent);
  earth->AddMaterial(Ar,fractionmass=1.288*perCent);
  earth->AddMaterial(CO2,fractionmass=0.048*perCent);
}

/////////////////////////////////////////////////////////////////

G4VPhysicalVolume* ECRSDetectorConstruction::ConstructWorld()
{
  // THE UNIVERSE

  universe_x = mult*universeSize;
  universe_y = mult*universeSize;
  universe_z = mult*universeSize;
  universe = new G4Box("universe",universe_x,universe_y,universe_z);
  universe_log = new
G4LogicalVolume(universe,space,"universe_log",0,0,0);
  universe_phys
    = new
G4PVPlacement(0,G4ThreeVector(),"universe",universe_log,0,false,0);
  G4VisAttributes* UniverseAtt = new G4VisAttributes();
  UniverseAtt->SetVisibility(true);
  universe_log->SetVisAttributes(UniverseAtt);

  // Define the angles for spheres of atmosphere and ECRS
  G4double startAnglePhi = 0.0*deg;
  G4double spanningAnglePhi = 360.0*deg;
  G4double startAngleTheta = 0.0*deg;
  G4double spanningAngleTheta = 180.0*deg; //90

  G4int layers = 37;
  G4int n = layers;
  G4double max = 0.75;
  G4double color = 0;

  // THE ATMOSPHERE - LAYER 37
  G4double innerRadiusAtm37 = mult*0.0*m;
  G4double outerRadiusAtm37 = mult*(earthRadius + atmHeight[37]);
  atmo37 = new G4Sphere("atm37",innerRadiusAtm37,outerRadiusAtm37,

      startAnglePhi,spanningAnglePhi,startAngleTheta,spanningAngleThe
ta);
  atm37_log = new G4LogicalVolume(atmo37,air37,"atm37_log");
  atm37_phys = new G4PVPlacement(0,
                       G4ThreeVector(0,0,0),
                       atm37_log,"Atm37",universe_log,false,0);
  Atm37Att = new G4VisAttributes();
  color = max/layers*n;
  n--;
  Atm37Att->SetColour(G4Colour(color,color,color));
  Atm37Att->SetForceWireframe(true);
  Atm37Att->SetVisibility(visibility);
  atm37_log->SetVisAttributes(Atm37Att);

  // THE ATMOSPHERE - LAYER 36
  G4double innerRadiusAtm36 = mult*0.0*m;
  G4double outerRadiusAtm36 = mult*(earthRadius + atmHeight[36]);
  atmo36 = new G4Sphere("atm36",innerRadiusAtm36,outerRadiusAtm36,

      startAnglePhi,spanningAnglePhi,startAngleTheta,spanningAngleThe
ta);
  atm36_log = new G4LogicalVolume(atmo36,air36,"atm36_log");
  atm36_phys = new G4PVPlacement(0,
                       G4ThreeVector(0,0,0),
                       atm36_log,"Atm36",atm37_log,false,0);
  Atm36Att = new G4VisAttributes();
  color = max/layers*n;
  n--;
  Atm36Att->SetColour(G4Colour(color,color,color));
  Atm36Att->SetForceWireframe(true);
  Atm36Att->SetVisibility(visibility);
  atm36_log->SetVisAttributes(Atm36Att);

  // THE ATMOSPHERE - LAYER 35
  G4double innerRadiusAtm35 = mult*0.0*m;
  G4double outerRadiusAtm35 = mult*(earthRadius + atmHeight[35]);
  atmo35 = new G4Sphere("atm35",innerRadiusAtm35,outerRadiusAtm35,

      startAnglePhi,spanningAnglePhi,startAngleTheta,spanningAngleThe
ta);
  atm35_log = new G4LogicalVolume(atmo35,air35,"atm35_log");
  atm35_phys = new G4PVPlacement(0,
                         G4ThreeVector(0,0,0),
                         atm35_log,"Atm35",atm36_log,false,0);
  Atm35Att = new G4VisAttributes();
  color = max/layers*n;
  n--;
  Atm35Att->SetColour(G4Colour(color,color,color));
  Atm35Att->SetForceWireframe(true);
  Atm35Att->SetVisibility(visibility);
  atm35_log->SetVisAttributes(Atm35Att);

  // THE ATMOSPHERE - LAYER 34
  G4double innerRadiusAtm34 = mult*0.0*m;
  G4double outerRadiusAtm34 = mult*(earthRadius + atmHeight[34]);
  atmo34 = new G4Sphere("atm34",innerRadiusAtm34,outerRadiusAtm34,

      startAnglePhi,spanningAnglePhi,startAngleTheta,spanningAngleThe
ta);
  atm34_log = new G4LogicalVolume(atmo34,air34,"atm34_log");
  atm34_phys = new G4PVPlacement(0,
                         G4ThreeVector(0,0,0),
                         atm34_log,"Atm34",atm35_log,false,0);
  Atm34Att = new G4VisAttributes();
  color = max/layers*n;
  n--;
  Atm34Att->SetColour(G4Colour(color,color,color));
  Atm34Att->SetForceWireframe(true);
  Atm34Att->SetVisibility(visibility);
  atm34_log->SetVisAttributes(Atm34Att);

  // THE ATMOSPHERE - LAYER 33

  G4double innerRadiusAtm33 = mult*0.0*m;
  G4double outerRadiusAtm33 = mult*(earthRadius + atmHeight[33]);
  atmo33 = new G4Sphere("atm33",innerRadiusAtm33,outerRadiusAtm33,

      startAnglePhi,spanningAnglePhi,startAngleTheta,spanningAngleThe
ta);
  atm33_log = new G4LogicalVolume(atmo33,air33,"atm33_log");
  atm33_phys = new G4PVPlacement(0,
                         G4ThreeVector(0,0,0),
                         atm33_log,"Atm33",atm34_log,false,0);
  Atm33Att = new G4VisAttributes();
  color = max/layers*n;
  n--;
  Atm33Att->SetColour(G4Colour(color,color,color));
  Atm33Att->SetForceWireframe(true);
  Atm33Att->SetVisibility(visibility);
  atm33_log->SetVisAttributes(Atm33Att);

  // THE ATMOSPHERE - LAYER 32
   G4double innerRadiusAtm32 = mult*0.0*m;
   G4double outerRadiusAtm32 = mult*(earthRadius + atmHeight[32]);
   atmo32 = new G4Sphere("atm32",innerRadiusAtm32,outerRadiusAtm32,

startAnglePhi,spanningAnglePhi,startAngleTheta,spanningAngleTheta);
   atm32_log = new G4LogicalVolume(atmo32,air32,"atm32_log");
   atm32_phys = new G4PVPlacement(0,
                          G4ThreeVector(0,0,0),
                          atm32_log,"Atm32",atm33_log,false,0);
   Atm32Att = new G4VisAttributes();
   color = max/layers*n;
   n--;
   Atm32Att->SetColour(G4Colour(color,color,color));
   Atm32Att->SetForceWireframe(true);
   Atm32Att->SetVisibility(visibility);
   atm32_log->SetVisAttributes(Atm32Att);

   // THE ATMOSPHERE - LAYER 31
   G4double innerRadiusAtm31 = mult*0.0*m;
   G4double outerRadiusAtm31 = mult*(earthRadius + atmHeight[31]);
   atmo31 = new G4Sphere("atm31",innerRadiusAtm31,outerRadiusAtm31,

startAnglePhi,spanningAnglePhi,startAngleTheta,spanningAngleTheta);
   atm31_log = new G4LogicalVolume(atmo31,air31,"atm31_log");
   atm31_phys = new G4PVPlacement(0,
                          G4ThreeVector(0,0,0),
                          atm31_log,"Atm31",atm32_log,false,0);
   Atm31Att = new G4VisAttributes();
   color = max/layers*n;
   n--;
   Atm31Att->SetColour(G4Colour(color,color,color));
   Atm31Att->SetForceWireframe(true);
   Atm31Att->SetVisibility(visibility);
   atm31_log->SetVisAttributes(Atm31Att);

   // THE ATMOSPHERE - LAYER 30
   G4double innerRadiusAtm30 = mult*0.0*m;
   G4double outerRadiusAtm30 = mult*(earthRadius + atmHeight[30]);
   atmo30 = new G4Sphere("atm30",innerRadiusAtm30,outerRadiusAtm30,

startAnglePhi,spanningAnglePhi,startAngleTheta,spanningAngleTheta);
   atm30_log = new G4LogicalVolume(atmo30,air30,"atm30_log");
   atm30_phys = new G4PVPlacement(0,
                          G4ThreeVector(0,0,0),
                          atm30_log,"Atm30",atm31_log,false,0);
   Atm30Att = new G4VisAttributes();
   color = max/layers*n;
   n--;
   Atm30Att->SetColour(G4Colour(color,color,color));
   Atm30Att->SetForceWireframe(true);
   Atm30Att->SetVisibility(visibility);
   atm30_log->SetVisAttributes(Atm30Att);

   // THE ATMOSPHERE - LAYER 29
   G4double innerRadiusAtm29 = mult*0.0*m;
   G4double outerRadiusAtm29 = mult*(earthRadius + atmHeight[29]);
   atmo29 = new G4Sphere("atm29",innerRadiusAtm29,outerRadiusAtm29,

startAnglePhi,spanningAnglePhi,startAngleTheta,spanningAngleTheta);
   atm29_log = new G4LogicalVolume(atmo29,air29,"atm29_log");
   atm29_phys = new G4PVPlacement(0,
                          G4ThreeVector(0,0,0),
                          atm29_log,"Atm29",atm30_log,false,0);
   Atm29Att = new G4VisAttributes();
   color = max/layers*n;
   n--;
   Atm29Att->SetColour(G4Colour(color,color,color));
   Atm29Att->SetForceWireframe(true);
   Atm29Att->SetVisibility(visibility);
   atm29_log->SetVisAttributes(Atm29Att);

   // THE ATMOSPHERE - LAYER 28
   G4double innerRadiusAtm28 = mult*0.0*m;
   G4double outerRadiusAtm28 = mult*(earthRadius + atmHeight[28]);
   atmo28 = new G4Sphere("atm28",innerRadiusAtm28,outerRadiusAtm28,

startAnglePhi,spanningAnglePhi,startAngleTheta,spanningAngleTheta);
   atm28_log = new G4LogicalVolume(atmo28,air28,"atm28_log");
   atm28_phys = new G4PVPlacement(0,
                          G4ThreeVector(0,0,0),
                          atm28_log,"Atm28",atm29_log,false,0);
   Atm28Att = new G4VisAttributes();
   color = max/layers*n;
   n--;
   Atm28Att->SetColour(G4Colour(color,color,color));
   Atm28Att->SetForceWireframe(true);
   Atm28Att->SetVisibility(visibility);
   atm28_log->SetVisAttributes(Atm28Att);

   // THE ATMOSPHERE - LAYER 27
   G4double innerRadiusAtm27 = mult*0.0*m;
   G4double outerRadiusAtm27 = mult*(earthRadius + atmHeight[27]);
   atmo27 = new G4Sphere("atm27",innerRadiusAtm27,outerRadiusAtm27,

startAnglePhi,spanningAnglePhi,startAngleTheta,spanningAngleTheta);
   atm27_log = new G4LogicalVolume(atmo27,air27,"atm27_log");
   atm27_phys = new G4PVPlacement(0,
                          G4ThreeVector(0,0,0),
                          atm27_log,"Atm27",atm28_log,false,0);
   Atm27Att = new G4VisAttributes();
   color = max/layers*n;
   n--;
   Atm27Att->SetColour(G4Colour(color,color,color));
   Atm27Att->SetForceWireframe(true);
   Atm27Att->SetVisibility(visibility);
   atm27_log->SetVisAttributes(Atm27Att);

   // THE ATMOSPHERE - LAYER 26
   G4double innerRadiusAtm26 = mult*0.0*m;
   G4double outerRadiusAtm26 = mult*(earthRadius + atmHeight[26]);
   atmo26 = new G4Sphere("atm26",innerRadiusAtm26,outerRadiusAtm26,

startAnglePhi,spanningAnglePhi,startAngleTheta,spanningAngleTheta);
   atm26_log = new G4LogicalVolume(atmo26,air26,"atm26_log");
   atm26_phys = new G4PVPlacement(0,
                       G4ThreeVector(0,0,0),
                       atm26_log,"Atm26",atm27_log,false,0);
   Atm26Att = new G4VisAttributes();
   color = max/layers*n;
   n--;
   Atm26Att->SetColour(G4Colour(color,color,color));
   Atm26Att->SetForceWireframe(true);
   Atm26Att->SetVisibility(visibility);
   atm26_log->SetVisAttributes(Atm26Att);

   // THE ATMOSPHERE - LAYER 25
   G4double innerRadiusAtm25 = mult*0.0*m;
   G4double outerRadiusAtm25 = mult*(earthRadius + atmHeight[25]);
   atmo25 = new G4Sphere("atm25",innerRadiusAtm25,outerRadiusAtm25,

startAnglePhi,spanningAnglePhi,startAngleTheta,spanningAngleTheta);
   atm25_log = new G4LogicalVolume(atmo25,air25,"atm25_log");
   atm25_phys = new G4PVPlacement(0,
                       G4ThreeVector(0,0,0),
                       atm25_log,"Atm25",atm26_log,false,0);
   Atm25Att = new G4VisAttributes();
   color = max/layers*n;
   n--;
   Atm25Att->SetColour(G4Colour(color,color,color));
   Atm25Att->SetForceWireframe(true);
   Atm25Att->SetVisibility(visibility);
   atm25_log->SetVisAttributes(Atm25Att);

   // THE ATMOSPHERE - LAYER 24
   G4double innerRadiusAtm24 = mult*0.0*m;
   G4double outerRadiusAtm24 = mult*(earthRadius + atmHeight[24]);
   atmo24 = new G4Sphere("atm24",innerRadiusAtm24,outerRadiusAtm24,

startAnglePhi,spanningAnglePhi,startAngleTheta,spanningAngleTheta);
   atm24_log = new G4LogicalVolume(atmo24,air24,"atm24_log");
   atm24_phys = new G4PVPlacement(0,
                       G4ThreeVector(0,0,0),
                       atm24_log,"Atm24",atm25_log,false,0);
   Atm24Att = new G4VisAttributes();
   color = max/layers*n;
   n--;
   Atm24Att->SetColour(G4Colour(color,color,color));
   Atm24Att->SetForceWireframe(true);
   Atm24Att->SetVisibility(visibility);
   atm24_log->SetVisAttributes(Atm24Att);

   // THE ATMOSPHERE - LAYER 23
   G4double innerRadiusAtm23 = mult*0.0*m;
   G4double outerRadiusAtm23 = mult*(earthRadius + atmHeight[23]);
   atmo23 = new G4Sphere("atm23",innerRadiusAtm23,outerRadiusAtm23,

startAnglePhi,spanningAnglePhi,startAngleTheta,spanningAngleTheta);
   atm23_log = new G4LogicalVolume(atmo23,air23,"atm23_log");
   atm23_phys = new G4PVPlacement(0,
                       G4ThreeVector(0,0,0),
                       atm23_log,"Atm23",atm24_log,false,0);
   Atm23Att = new G4VisAttributes();
   color = max/layers*n;
   n--;
   Atm23Att->SetColour(G4Colour(color,color,color));
   Atm23Att->SetForceWireframe(true);
   Atm23Att->SetVisibility(visibility);
   atm23_log->SetVisAttributes(Atm23Att);

   // THE ATMOSPHERE - LAYER 22
   G4double innerRadiusAtm22 = mult*0.0*m;
   G4double outerRadiusAtm22 = mult*(earthRadius + atmHeight[22]);
   atmo22 = new G4Sphere("atm22",innerRadiusAtm22,outerRadiusAtm22,

startAnglePhi,spanningAnglePhi,startAngleTheta,spanningAngleTheta);
   atm22_log = new G4LogicalVolume(atmo22,air22,"atm22_log");
   atm22_phys = new G4PVPlacement(0,
                       G4ThreeVector(0,0,0),
                       atm22_log,"Atm22",atm23_log,false,0);
   Atm22Att = new G4VisAttributes();
   color = max/layers*n;
   n--;
   Atm22Att->SetColour(G4Colour(color,color,color));
   Atm22Att->SetForceWireframe(true);
   Atm22Att->SetVisibility(visibility);
   atm22_log->SetVisAttributes(Atm22Att);

   // THE ATMOSPHERE - LAYER 21
   G4double innerRadiusAtm21 = mult*0.0*m;
   G4double outerRadiusAtm21 = mult*(earthRadius + atmHeight[21]);
   atmo21 = new G4Sphere("atm21",innerRadiusAtm21,outerRadiusAtm21,

startAnglePhi,spanningAnglePhi,startAngleTheta,spanningAngleTheta);
   atm21_log = new G4LogicalVolume(atmo21,air21,"atm21_log");
   atm21_phys = new G4PVPlacement(0,
                       G4ThreeVector(0,0,0),
                       atm21_log,"Atm21",atm22_log,false,0);
   Atm21Att = new G4VisAttributes();
   color = max/layers*n;
   n--;
   Atm21Att->SetColour(G4Colour(color,color,color));
   Atm21Att->SetForceWireframe(true);
   Atm21Att->SetVisibility(visibility);
   atm21_log->SetVisAttributes(Atm21Att);

   // THE ATMOSPHERE - LAYER 20
   G4double innerRadiusAtm20 = mult*0.0*m;
   G4double outerRadiusAtm20 = mult*(earthRadius + atmHeight[20]);
   atmo20 = new G4Sphere("atm20",innerRadiusAtm20,outerRadiusAtm20,

startAnglePhi,spanningAnglePhi,startAngleTheta,spanningAngleTheta);
   atm20_log = new G4LogicalVolume(atmo20,air20,"atm20_log");
   atm20_phys = new G4PVPlacement(0,
                       G4ThreeVector(0,0,0),
                       atm20_log,"Atm20",atm21_log,false,0);
   Atm20Att = new G4VisAttributes();
   color = max/layers*n;
   n--;
   Atm20Att->SetColour(G4Colour(color,color,color));
   Atm20Att->SetForceWireframe(true);
   Atm20Att->SetVisibility(visibility);
   atm20_log->SetVisAttributes(Atm20Att);

   // THE ATMOSPHERE - LAYER 19
   G4double innerRadiusAtm19 = mult*0.0*m;
   G4double outerRadiusAtm19 = mult*(earthRadius + atmHeight[19]);
   atmo19 = new G4Sphere("atm19",innerRadiusAtm19,outerRadiusAtm19,

startAnglePhi,spanningAnglePhi,startAngleTheta,spanningAngleTheta);
   atm19_log = new G4LogicalVolume(atmo19,air19,"atm19_log");
   atm19_phys = new G4PVPlacement(0,
                       G4ThreeVector(0,0,0),
                       atm19_log,"Atm19",atm20_log,false,0);
   Atm19Att = new G4VisAttributes();
   color = max/layers*n;
   n--;
   Atm19Att->SetColour(G4Colour(color,color,color));
   Atm19Att->SetForceWireframe(true);
   Atm19Att->SetVisibility(visibility);
   atm19_log->SetVisAttributes(Atm19Att);

   // THE ATMOSPHERE - LAYER 18
   G4double innerRadiusAtm18 = mult*0.0*m;
   G4double outerRadiusAtm18 = mult*(earthRadius + atmHeight[18]);
   atmo18 = new G4Sphere("atm18",innerRadiusAtm18,outerRadiusAtm18,

startAnglePhi,spanningAnglePhi,startAngleTheta,spanningAngleTheta);
   atm18_log = new G4LogicalVolume(atmo18,air18,"atm18_log");
   atm18_phys = new G4PVPlacement(0,
                       G4ThreeVector(0,0,0),
                       atm18_log,"Atm18",atm19_log,false,0);
   Atm18Att = new G4VisAttributes();
   color = max/layers*n;
   n--;
   Atm18Att->SetColour(G4Colour(color,color,color));
   Atm18Att->SetForceWireframe(true);
   Atm18Att->SetVisibility(visibility);
   atm18_log->SetVisAttributes(Atm18Att);

   // THE ATMOSPHERE - LAYER 17
   G4double innerRadiusAtm17 = mult*0.0*m;
   G4double outerRadiusAtm17 = mult*(earthRadius + atmHeight[17]);
   atmo17 = new G4Sphere("atm17",innerRadiusAtm17,outerRadiusAtm17,

startAnglePhi,spanningAnglePhi,startAngleTheta,spanningAngleTheta);
   atm17_log = new G4LogicalVolume(atmo17,air17,"atm17_log");
   atm17_phys = new G4PVPlacement(0,
                          G4ThreeVector(0,0,0),
                          atm17_log,"Atm17",atm18_log,false,0);
   Atm17Att = new G4VisAttributes();
   color = max/layers*n;
   n--;
   Atm17Att->SetColour(G4Colour(color,color,color));
   Atm17Att->SetForceWireframe(true);
   Atm17Att->SetVisibility(visibility);
   atm17_log->SetVisAttributes(Atm17Att);

   // THE ATMOSPHERE - LAYER 16
   G4double innerRadiusAtm16 = mult*0.0*m;
   G4double outerRadiusAtm16 = mult*(earthRadius + atmHeight[16]);
   atmo16 = new G4Sphere("atm16",innerRadiusAtm16,outerRadiusAtm16,

startAnglePhi,spanningAnglePhi,startAngleTheta,spanningAngleTheta);
   atm16_log = new G4LogicalVolume(atmo16,air16,"atm16_log");
   atm16_phys = new G4PVPlacement(0,
                          G4ThreeVector(0,0,0),
                          atm16_log,"Atm16",atm17_log,false,0);
   Atm16Att = new G4VisAttributes();
   color = max/layers*n;
   n--;
   Atm16Att->SetColour(G4Colour(color,color,color));
   Atm16Att->SetForceWireframe(true);
   Atm16Att->SetVisibility(visibility);
   atm16_log->SetVisAttributes(Atm16Att);

   // THE ATMOSPHERE - LAYER 15
   G4double innerRadiusAtm15 = mult*0.0*m;
   G4double outerRadiusAtm15 = mult*(earthRadius + atmHeight[15]);
   atmo15 = new G4Sphere("atm15",innerRadiusAtm15,outerRadiusAtm15,

startAnglePhi,spanningAnglePhi,startAngleTheta,spanningAngleTheta);
   atm15_log = new G4LogicalVolume(atmo15,air15,"atm15_log");
   atm15_phys = new G4PVPlacement(0,
                          G4ThreeVector(0,0,0),
                          atm15_log,"Atm15",atm16_log,false,0);
   Atm15Att = new G4VisAttributes();
   color = max/layers*n;
   n--;
   Atm15Att->SetColour(G4Colour(color,color,color));
   Atm15Att->SetForceWireframe(true);
   Atm15Att->SetVisibility(visibility);
   atm15_log->SetVisAttributes(Atm15Att);
   // THE ATMOSPHERE - LAYER 14
   G4double innerRadiusAtm14 = mult*0.0*m;
   G4double outerRadiusAtm14 = mult*(earthRadius + atmHeight[14]);
   atmo14 = new G4Sphere("atm14",innerRadiusAtm14,outerRadiusAtm14,

startAnglePhi,spanningAnglePhi,startAngleTheta,spanningAngleTheta);
   atm14_log = new G4LogicalVolume(atmo14,air14,"atm14_log");
   atm14_phys = new G4PVPlacement(0,
                       G4ThreeVector(0,0,0),
                       atm14_log,"Atm14",atm15_log,false,0);
   Atm14Att = new G4VisAttributes();
   color = max/layers*n;
   n--;
   Atm14Att->SetColour(G4Colour(color,color,color));
   Atm14Att->SetForceWireframe(true);
   Atm14Att->SetVisibility(visibility);
   atm14_log->SetVisAttributes(Atm14Att);

   // THE ATMOSPHERE - LAYER 13
   G4double innerRadiusAtm13 = mult*0.0*m;
   G4double outerRadiusAtm13 = mult*(earthRadius + atmHeight[13]);
   atmo13 = new G4Sphere("atm13",innerRadiusAtm13,outerRadiusAtm13,

startAnglePhi,spanningAnglePhi,startAngleTheta,spanningAngleTheta);
   atm13_log = new G4LogicalVolume(atmo13,air13,"atm13_log");
   atm13_phys = new G4PVPlacement(0,
                       G4ThreeVector(0,0,0),
                       atm13_log,"Atm13",atm14_log,false,0);
   Atm13Att = new G4VisAttributes();
   color = max/layers*n;
   n--;
   Atm13Att->SetColour(G4Colour(color,color,color));
   Atm13Att->SetForceWireframe(true);
   Atm13Att->SetVisibility(visibility);
   atm13_log->SetVisAttributes(Atm13Att);

   // THE ATMOSPHERE - LAYER 12
   G4double innerRadiusAtm12 = mult*0.0*m;
   G4double outerRadiusAtm12 = mult*(earthRadius + atmHeight[12]);
   atmo12 = new G4Sphere("atm12",innerRadiusAtm12,outerRadiusAtm12,

startAnglePhi,spanningAnglePhi,startAngleTheta,spanningAngleTheta);
   atm12_log = new G4LogicalVolume(atmo12,air12,"atm12_log");
   atm12_phys = new G4PVPlacement(0,
                       G4ThreeVector(0,0,0),
                       atm12_log,"Atm12",atm13_log,false,0);
   Atm12Att = new G4VisAttributes();
   color = max/layers*n;
   n--;
   Atm12Att->SetColour(G4Colour(color,color,color));
   Atm12Att->SetForceWireframe(true);
   Atm12Att->SetVisibility(visibility);
   atm12_log->SetVisAttributes(Atm12Att);
   // THE ATMOSPHERE - LAYER 11
   G4double innerRadiusAtm11 = mult*0.0*m;
   G4double outerRadiusAtm11 = mult*(earthRadius + atmHeight[11]);
   atmo11 = new G4Sphere("atm11",innerRadiusAtm11,outerRadiusAtm11,

startAnglePhi,spanningAnglePhi,startAngleTheta,spanningAngleTheta);
   atm11_log = new G4LogicalVolume(atmo11,air11,"atm11_log");
   atm11_phys = new G4PVPlacement(0,
                       G4ThreeVector(0,0,0),
                       atm11_log,"Atm11",atm12_log,false,0);
   Atm11Att = new G4VisAttributes();
   color = max/layers*n;
   n--;
   Atm11Att->SetColour(G4Colour(color,color,color));
   Atm11Att->SetForceWireframe(true);
   Atm11Att->SetVisibility(visibility);
   atm11_log->SetVisAttributes(Atm11Att);

   // THE ATMOSPHERE - LAYER 10
   G4double innerRadiusAtm10 = mult*0.0*m;
   G4double outerRadiusAtm10 = mult*(earthRadius + atmHeight[10]);
   atmo10 = new G4Sphere("atm10",innerRadiusAtm10,outerRadiusAtm10,

startAnglePhi,spanningAnglePhi,startAngleTheta,spanningAngleTheta);
   atm10_log = new G4LogicalVolume(atmo10,air10,"atm10_log");
   atm10_phys = new G4PVPlacement(0,
                       G4ThreeVector(0,0,0),
                       atm10_log,"Atm10",atm11_log,false,0);
   Atm10Att = new G4VisAttributes();
   color = max/layers*n;
   n--;
   Atm10Att->SetColour(G4Colour(color,color,color));
   Atm10Att->SetForceWireframe(true);
   Atm10Att->SetVisibility(visibility);
   atm10_log->SetVisAttributes(Atm10Att);

   // THE ATMOSPHERE - LAYER 9

   G4double innerRadiusAtm9 = mult*0.0*m;
   G4double outerRadiusAtm9 = mult*(earthRadius + atmHeight[9]);
   atmo9 = new G4Sphere("atm9",innerRadiusAtm9,outerRadiusAtm9,

      startAnglePhi,spanningAnglePhi,startAngleTheta,spanningAngleThe
ta);
   atm9_log = new G4LogicalVolume(atmo9,air9,"atm9_log");
   atm9_phys = new G4PVPlacement(0,
                      G4ThreeVector(0,0,0),
                      atm9_log,"Atm9",atm10_log,false,0);
   Atm9Att = new G4VisAttributes();
   color = max/layers*n;
   n--;
   Atm9Att->SetColour(G4Colour(color,color,color));
   Atm9Att->SetForceWireframe(true);
   Atm9Att->SetVisibility(visibility);
   atm9_log->SetVisAttributes(Atm9Att);
   // THE ATMOSPHERE - LAYER 8
   G4double innerRadiusAtm8 = mult*0.0*m;
   G4double outerRadiusAtm8 = mult*(earthRadius + atmHeight[8]);
   atmo8 = new G4Sphere("atm8",innerRadiusAtm8,outerRadiusAtm8,

      startAnglePhi,spanningAnglePhi,startAngleTheta,spanningAngleThe
ta);
   atm8_log = new G4LogicalVolume(atmo8,air8,"atm8_log");
   atm8_phys = new G4PVPlacement(0,
                         G4ThreeVector(0,0,0),
                         atm8_log,"Atm8",atm9_log,false,0);
   Atm8Att = new G4VisAttributes();
   color = max/layers*n;
   n--;
   Atm8Att->SetColour(G4Colour(color,color,color));
   Atm8Att->SetForceWireframe(true);
   Atm8Att->SetVisibility(visibility);
   atm8_log->SetVisAttributes(Atm8Att);

   // THE ATMOSPHERE - LAYER 7
   G4double innerRadiusAtm7 = mult*0.0*m;
   G4double outerRadiusAtm7 = mult*(earthRadius + atmHeight[7]);
   atmo7 = new G4Sphere("atm7",innerRadiusAtm7,outerRadiusAtm7,

      startAnglePhi,spanningAnglePhi,startAngleTheta,spanningAngleThe
ta);
   atm7_log = new G4LogicalVolume(atmo7,air7,"atm7_log");
   atm7_phys = new G4PVPlacement(0,
                         G4ThreeVector(0,0,0),
                         atm7_log,"Atm7",atm8_log,false,0);
   Atm7Att = new G4VisAttributes();
   color = max/layers*n;
   n--;
   Atm7Att->SetColour(G4Colour(color,color,color));
   Atm7Att->SetForceWireframe(true);
   Atm7Att->SetVisibility(visibility);
   atm7_log->SetVisAttributes(Atm7Att);

   // THE ATMOSPHERE - LAYER 6
   G4double innerRadiusAtm6 = mult*0.0*m;
   G4double outerRadiusAtm6 = mult*(earthRadius + atmHeight[6]);
   atmo6 = new G4Sphere("atm6",innerRadiusAtm6,outerRadiusAtm6,

      startAnglePhi,spanningAnglePhi,startAngleTheta,spanningAngleThe
ta);
   atm6_log = new G4LogicalVolume(atmo6,air6,"atm6_log");
   atm6_phys = new G4PVPlacement(0,
                         G4ThreeVector(0,0,0),
                         atm6_log,"Atm6",atm7_log,false,0);
   Atm6Att = new G4VisAttributes();
   color = max/layers*n;
   n--;
   Atm6Att->SetColour(G4Colour(color,color,color));
   Atm6Att->SetForceWireframe(true);
   Atm6Att->SetVisibility(visibility);
   atm6_log->SetVisAttributes(Atm6Att);

   // THE ATMOSPHERE - LAYER 5
   G4double innerRadiusAtm5 = mult*0.0*m;
   G4double outerRadiusAtm5 = mult*(earthRadius + atmHeight[5]);
   atmo5 = new G4Sphere("atm5",innerRadiusAtm5,outerRadiusAtm5,

      startAnglePhi,spanningAnglePhi,startAngleTheta,spanningAngleThe
ta);
   atm5_log = new G4LogicalVolume(atmo5,air5,"atm5_log");
   atm5_phys = new G4PVPlacement(0,
                      G4ThreeVector(0,0,0),
                      atm5_log,"Atm5",atm6_log,false,0);
   Atm5Att = new G4VisAttributes();
   color = max/layers*n;
   n--;
   Atm5Att->SetColour(G4Colour(color,color,color));
   Atm5Att->SetForceWireframe(true);
   Atm5Att->SetVisibility(visibility);
   atm5_log->SetVisAttributes(Atm5Att);

   // THE ATMOSPHERE - LAYER 4
   G4double innerRadiusAtm4 = mult*0.0*m;
   G4double outerRadiusAtm4 = mult*(earthRadius + atmHeight[4]);
   atmo4 = new G4Sphere("atm4",innerRadiusAtm4,outerRadiusAtm4,

      startAnglePhi,spanningAnglePhi,startAngleTheta,spanningAngleThe
ta);
   atm4_log = new G4LogicalVolume(atmo4,air4,"atm4_log");
   atm4_phys = new G4PVPlacement(0,
                      G4ThreeVector(0,0,0),
                      atm4_log,"Atm4",atm5_log,false,0);
   Atm4Att = new G4VisAttributes();
   color = max/layers*n;
   n--;
   Atm4Att->SetColour(G4Colour(color,color,color));
   Atm4Att->SetForceWireframe(true);
   Atm4Att->SetVisibility(visibility);
   atm4_log->SetVisAttributes(Atm4Att);

   // THE ATMOSPHERE - LAYER 3
   G4double innerRadiusAtm3 = mult*0.0*m;
   G4double outerRadiusAtm3 = mult*(earthRadius + atmHeight[3]);
   atmo3 = new G4Sphere("atm3",innerRadiusAtm3,outerRadiusAtm3,

      startAnglePhi,spanningAnglePhi,startAngleTheta,spanningAngleThe
ta);
   atm3_log = new G4LogicalVolume(atmo3,air3,"atm3_log");
   atm3_phys = new G4PVPlacement(0,
                      G4ThreeVector(0,0,0),
                      atm3_log,"Atm3",atm4_log,false,0);
   Atm3Att = new G4VisAttributes();
   color = max/layers*n;
   n--;
   Atm3Att->SetColour(G4Colour(color,color,color));
   Atm3Att->SetForceWireframe(true);
   Atm3Att->SetVisibility(visibility);
   atm3_log->SetVisAttributes(Atm3Att);

   // THE ATMOSPHERE - LAYER 2

   G4double innerRadiusAtm2 = mult*0.0*m;
   G4double outerRadiusAtm2 = mult*(earthRadius + atmHeight[2]);
   atmo2 = new G4Sphere("atm2",innerRadiusAtm2,outerRadiusAtm2,

      startAnglePhi,spanningAnglePhi,startAngleTheta,spanningAngleThe
ta);
   atm2_log = new G4LogicalVolume(atmo2,air2,"atm2_log");
   atm2_phys = new G4PVPlacement(0,
                         G4ThreeVector(0,0,0),
                         atm2_log,"Atm2",atm3_log,false,0);
   Atm2Att = new G4VisAttributes();
   color = max/layers*n;
   n--;
   Atm2Att->SetColour(G4Colour(color,color,color));
   Atm2Att->SetForceWireframe(true);
   Atm2Att->SetVisibility(visibility);
   atm2_log->SetVisAttributes(Atm2Att);

   // THE ATMOSPHERE - LAYER 1
   G4double innerRadiusAtm1 = mult*0.0*m;
   G4double outerRadiusAtm1 = mult*(earthRadius + atmHeight[1]);
   atmo1 = new G4Sphere("atm1",innerRadiusAtm1,outerRadiusAtm1,

      startAnglePhi,spanningAnglePhi,startAngleTheta,spanningAngleThe
ta);
   atm1_log = new G4LogicalVolume(atmo1,air1,"atm1_log");
   atm1_phys = new G4PVPlacement(0,
                         G4ThreeVector(0,0,0),
                         atm1_log,"Atm1",atm2_log,false,0);
   Atm1Att = new G4VisAttributes();
   color = max/layers*n;
   n--;
   Atm1Att->SetColour(G4Colour(color,color,color));
   Atm1Att->SetForceWireframe(true);
   Atm1Att->SetVisibility(visibility);
   atm1_log->SetVisAttributes(Atm1Att);

   // THE EARTH
   G4double innerRadiusECRS = mult*0.0*m;
   G4double outerRadiusECRS = mult*earthRadius;
   ECRS_sphere = new G4Sphere("ECRS_sphere",
                        innerRadiusECRS,outerRadiusECRS,

startAnglePhi,spanningAnglePhi,startAngleTheta,spanningAngleTheta);
   ECRS_log = new G4LogicalVolume(ECRS_sphere,earth,"ECRS_log");
   ECRS_phys = new G4PVPlacement(0,
                         G4ThreeVector(0,0,0),
                         ECRS_log,"ECRS",atm1_log,false,0);
   G4VisAttributes* ECRSAtt = new G4VisAttributes(G4Colour(0,0,1));
   ECRSAtt->SetForceSolid(true);
   ECRS_log->SetVisAttributes(ECRSAtt);

   // Set the atmosphere as the Sensitive Detector

   G4SDManager* SDman = G4SDManager::GetSDMpointer();

   if (!atmosphereSD)
     {
       atmosphereSD = new ECRSAtmosphereSD("AtmoSD",this);
       SDman->AddNewDetector(atmosphereSD);
     }

   if (atm1_log) atm1_log->SetSensitiveDetector(atmosphereSD);
   if (atm5_log) atm5_log->SetSensitiveDetector(atmosphereSD);
   if (atm10_log) atm10_log->SetSensitiveDetector(atmosphereSD);
   if (atm15_log) atm15_log->SetSensitiveDetector(atmosphereSD);

   PrintUniverseParameters();
   G4cout << *(G4Material::GetMaterialTable()) << G4endl;

   return universe_phys;
}

//////////////////////////////////////////////////////////////

void ECRSDetectorConstruction::PrintUniverseParameters()
{
  G4cout << G4endl;
  G4cout << "Universe XY half dimensions: ";
  G4cout << G4BestUnit(universe_x,"Length") << G4endl;
  G4cout << "Universe Z half dimension: ";
  G4cout << G4BestUnit(universe_z,"Length") << G4endl;
  G4cout << "ECRS Radius: ";
  G4cout << G4BestUnit(mult*earthRadius,"Length") << G4endl;
}

//////////////////////////////////////////////////////////////

void ECRSDetectorConstruction::SetMultiplier(G4double newValue)
{
  mult = newValue;
  G4cout << "You must use the command: /geometry/update before
beamOn";
  G4cout << G4endl;
}

//////////////////////////////////////////////////////////////

void ECRSDetectorConstruction::SetDensityMult(G4double newValue)
{
  multDensity = newValue;
  G4cout << "You must use the command: /geometry/update before
beamOn";
```

```
  G4cout << G4endl;
}

//////////////////////////////////////////////////////////////

void ECRSDetectorConstruction::SetVisFlag(G4bool newValue)
{
  Atm1Att->SetVisibility(newValue);
  atm1_log->SetVisAttributes(Atm1Att);

  Atm2Att->SetVisibility(newValue);
  atm2_log->SetVisAttributes(Atm2Att);

  Atm3Att->SetVisibility(newValue);
  atm3_log->SetVisAttributes(Atm3Att);

  Atm4Att->SetVisibility(newValue);
  atm4_log->SetVisAttributes(Atm4Att);

  Atm5Att->SetVisibility(newValue);
  atm5_log->SetVisAttributes(Atm5Att);

  Atm6Att->SetVisibility(newValue);
  atm6_log->SetVisAttributes(Atm6Att);

  Atm7Att->SetVisibility(newValue);
  atm7_log->SetVisAttributes(Atm7Att);

  Atm8Att->SetVisibility(newValue);
  atm8_log->SetVisAttributes(Atm8Att);

  Atm9Att->SetVisibility(newValue);
  atm9_log->SetVisAttributes(Atm9Att);

  Atm10Att->SetVisibility(newValue);
  atm10_log->SetVisAttributes(Atm10Att);

  Atm11Att->SetVisibility(newValue);
  atm11_log->SetVisAttributes(Atm11Att);

  Atm12Att->SetVisibility(newValue);
  atm12_log->SetVisAttributes(Atm1Att);

  Atm13Att->SetVisibility(newValue);
  atm13_log->SetVisAttributes(Atm2Att);

  Atm14Att->SetVisibility(newValue);
  atm14_log->SetVisAttributes(Atm3Att);

  Atm15Att->SetVisibility(newValue);
  atm15_log->SetVisAttributes(Atm4Att);

  Atm16Att->SetVisibility(newValue);
  atm16_log->SetVisAttributes(Atm5Att);
```

```
Atm17Att->SetVisibility(newValue);
atm17_log->SetVisAttributes(Atm6Att);

Atm18Att->SetVisibility(newValue);
atm18_log->SetVisAttributes(Atm7Att);

Atm19Att->SetVisibility(newValue);
atm19_log->SetVisAttributes(Atm8Att);

Atm20Att->SetVisibility(newValue);
atm20_log->SetVisAttributes(Atm9Att);

Atm21Att->SetVisibility(newValue);
atm21_log->SetVisAttributes(Atm10Att);

Atm22Att->SetVisibility(newValue);
atm22_log->SetVisAttributes(Atm11Att);

Atm23Att->SetVisibility(newValue);
atm23_log->SetVisAttributes(Atm1Att);

Atm24Att->SetVisibility(newValue);
atm24_log->SetVisAttributes(Atm2Att);

Atm25Att->SetVisibility(newValue);
atm25_log->SetVisAttributes(Atm3Att);

Atm26Att->SetVisibility(newValue);
atm26_log->SetVisAttributes(Atm4Att);

Atm27Att->SetVisibility(newValue);
atm27_log->SetVisAttributes(Atm5Att);

Atm28Att->SetVisibility(newValue);
atm28_log->SetVisAttributes(Atm6Att);

Atm29Att->SetVisibility(newValue);
atm29_log->SetVisAttributes(Atm7Att);

Atm30Att->SetVisibility(newValue);
atm30_log->SetVisAttributes(Atm8Att);

Atm31Att->SetVisibility(newValue);
atm31_log->SetVisAttributes(Atm9Att);

Atm32Att->SetVisibility(newValue);
atm32_log->SetVisAttributes(Atm10Att);

Atm33Att->SetVisibility(newValue);
atm33_log->SetVisAttributes(Atm11Att);

Atm34Att->SetVisibility(newValue);
atm34_log->SetVisAttributes(Atm8Att);

Atm35Att->SetVisibility(newValue);
```

```
  atm35_log->SetVisAttributes(Atm9Att);

  Atm36Att->SetVisibility(newValue);
  atm36_log->SetVisAttributes(Atm10Att);

  Atm37Att->SetVisibility(newValue);
  atm37_log->SetVisAttributes(Atm11Att);

  G4cout << "You may have to use: /vis/viewer/refresh to see changes"
       << " in visualization.";
  G4cout << G4endl;
}

/////////////////////////////////////////////////////////////////

void ECRSDetectorConstruction::UpdateGeometry()
{
  G4RunManager::GetRunManager()->DefineWorldVolume(ConstructWorld());
}

/////////////////////////////////////////////////////////////////

void ECRSDetectorConstruction::UpdateMaterials()
{
  DefineMaterials();
  G4RunManager::GetRunManager()->DefineWorldVolume(ConstructWorld());
}

////////////////////////////////////////////////////////////////

void ECRSDetectorConstruction::PrintAllParameters()
{
  PrintUniverseParameters();
  G4cout << G4endl;
  G4cout << "Atmospheric Layer:\tHeight Above Surface:\tHeight From
Center:\n";
  G4cout << "Atm1\t\t\t" << G4BestUnit(mult*atmHeight[1],"Length") <<
"\t\t\t";
  G4cout << G4BestUnit(mult*(atmHeight[1]+earthRadius),"Length") <<
G4endl;
  G4cout << "Atm2\t\t\t" << G4BestUnit(mult*atmHeight[2],"Length") <<
"\t\t\t";
  G4cout << G4BestUnit(mult*(atmHeight[2]+earthRadius),"Length") <<
G4endl;
  G4cout << "Atm3\t\t\t" << G4BestUnit(mult*atmHeight[3],"Length") <<
"\t\t\t";
  G4cout << G4BestUnit(mult*(atmHeight[3]+earthRadius),"Length") <<
G4endl;
  G4cout << "Atm4\t\t\t" << G4BestUnit(mult*atmHeight[4],"Length") <<
"\t\t\t";
  G4cout << G4BestUnit(mult*(atmHeight[4]+earthRadius),"Length") <<
G4endl;
  G4cout << "Atm5\t\t\t" << G4BestUnit(mult*atmHeight[5],"Length") <<
"\t\t\t";
```

```
  G4cout << G4BestUnit(mult*(atmHeight[5]+earthRadius),"Length") <<
G4endl;
  G4cout << "Atm6\t\t\t" << G4BestUnit(mult*atmHeight[6],"Length") <<
"\t\t\t";
  G4cout << G4BestUnit(mult*(atmHeight[6]+earthRadius),"Length") <<
G4endl;
  G4cout << "Atm7\t\t\t" << G4BestUnit(mult*atmHeight[7],"Length") <<
"\t\t\t";
  G4cout << G4BestUnit(mult*(atmHeight[7]+earthRadius),"Length") <<
G4endl;
  G4cout << "Atm8\t\t\t" << G4BestUnit(mult*atmHeight[8],"Length") <<
"\t\t\t";
  G4cout << G4BestUnit(mult*(atmHeight[8]+earthRadius),"Length") <<
G4endl;
  G4cout << "Atm9\t\t\t" << G4BestUnit(mult*atmHeight[9],"Length") <<
"\t\t\t";
  G4cout << G4BestUnit(mult*(atmHeight[9]+earthRadius),"Length") <<
G4endl;
  G4cout << "Atm10\t\t\t" << G4BestUnit(mult*atmHeight[10],"Length")
<< "\t\t\t";
  G4cout << G4BestUnit(mult*(atmHeight[10]+earthRadius),"Length") <<
G4endl;
  G4cout << "Atm11\t\t\t" << G4BestUnit(mult*atmHeight[11],"Length")
<< "\t\t\t";
  G4cout << G4BestUnit(mult*(atmHeight[11]+earthRadius),"Length") <<
G4endl;
  G4cout << "Atm12\t\t\t" << G4BestUnit(mult*atmHeight[12],"Length")
<< "\t\t\t";
  G4cout << G4BestUnit(mult*(atmHeight[12]+earthRadius),"Length") <<
G4endl;
  G4cout << "Atm13\t\t\t" << G4BestUnit(mult*atmHeight[13],"Length")
<< "\t\t\t";
  G4cout << G4BestUnit(mult*(atmHeight[13]+earthRadius),"Length") <<
G4endl;
  G4cout << "Atm14\t\t\t" << G4BestUnit(mult*atmHeight[14],"Length")
<< "\t\t\t";
  G4cout << G4BestUnit(mult*(atmHeight[14]+earthRadius),"Length") <<
G4endl;
  G4cout << "Atm15\t\t\t" << G4BestUnit(mult*atmHeight[15],"Length")
<< "\t\t\t";
  G4cout << G4BestUnit(mult*(atmHeight[15]+earthRadius),"Length") <<
G4endl;
  G4cout << "Atm16\t\t\t" << G4BestUnit(mult*atmHeight[16],"Length")
<< "\t\t\t";
  G4cout << G4BestUnit(mult*(atmHeight[16]+earthRadius),"Length") <<
G4endl;
  G4cout << "Atm17\t\t\t" << G4BestUnit(mult*atmHeight[17],"Length")
<< "\t\t\t";
  G4cout << G4BestUnit(mult*(atmHeight[17]+earthRadius),"Length") <<
G4endl;
  G4cout << "Atm18\t\t\t" << G4BestUnit(mult*atmHeight[18],"Length")
<< "\t\t\t";
  G4cout << G4BestUnit(mult*(atmHeight[18]+earthRadius),"Length") <<
G4endl;
```

```
  G4cout << "Atm19\t\t\t" << G4BestUnit(mult*atmHeight[19],"Length")
<< "\t\t\t";
  G4cout << G4BestUnit(mult*(atmHeight[19]+earthRadius),"Length") <<
G4endl;
  G4cout << "Atm20\t\t\t" << G4BestUnit(mult*atmHeight[20],"Length")
<< "\t\t\t";
  G4cout << G4BestUnit(mult*(atmHeight[20]+earthRadius),"Length") <<
G4endl;
  G4cout << "Atm21\t\t\t" << G4BestUnit(mult*atmHeight[21],"Length")
<< "\t\t\t";
  G4cout << G4BestUnit(mult*(atmHeight[21]+earthRadius),"Length") <<
G4endl;
  G4cout << "Atm22\t\t\t" << G4BestUnit(mult*atmHeight[22],"Length")
<< "\t\t\t";
  G4cout << G4BestUnit(mult*(atmHeight[22]+earthRadius),"Length") <<
G4endl;
  G4cout << "Atm23\t\t\t" << G4BestUnit(mult*atmHeight[23],"Length")
<< "\t\t\t";
  G4cout << G4BestUnit(mult*(atmHeight[23]+earthRadius),"Length") <<
G4endl;
  G4cout << "Atm24\t\t\t" << G4BestUnit(mult*atmHeight[24],"Length")
<< "\t\t\t";
  G4cout << G4BestUnit(mult*(atmHeight[24]+earthRadius),"Length") <<
G4endl;
  G4cout << "Atm25\t\t\t" << G4BestUnit(mult*atmHeight[25],"Length")
<< "\t\t\t";
  G4cout << G4BestUnit(mult*(atmHeight[25]+earthRadius),"Length") <<
G4endl;
  G4cout << "Atm26\t\t\t" << G4BestUnit(mult*atmHeight[26],"Length")
<< "\t\t\t";
  G4cout << G4BestUnit(mult*(atmHeight[26]+earthRadius),"Length") <<
G4endl;
  G4cout << "Atm27\t\t\t" << G4BestUnit(mult*atmHeight[27],"Length")
<< "\t\t\t";
  G4cout << G4BestUnit(mult*(atmHeight[27]+earthRadius),"Length") <<
G4endl;
  G4cout << "Atm28\t\t\t" << G4BestUnit(mult*atmHeight[28],"Length")
<< "\t\t\t";
  G4cout << G4BestUnit(mult*(atmHeight[28]+earthRadius),"Length") <<
G4endl;
  G4cout << "Atm29\t\t\t" << G4BestUnit(mult*atmHeight[29],"Length")
<< "\t\t\t";
  G4cout << G4BestUnit(mult*(atmHeight[29]+earthRadius),"Length") <<
G4endl;
  G4cout << "Atm30\t\t\t" << G4BestUnit(mult*atmHeight[30],"Length")
<< "\t\t\t";
  G4cout << G4BestUnit(mult*(atmHeight[30]+earthRadius),"Length") <<
G4endl;
  G4cout << "Atm31\t\t\t" << G4BestUnit(mult*atmHeight[31],"Length")
<< "\t\t\t";
  G4cout << G4BestUnit(mult*(atmHeight[31]+earthRadius),"Length") <<
G4endl;
  G4cout << "Atm32\t\t\t" << G4BestUnit(mult*atmHeight[32],"Length")
<< "\t\t\t";
```

```
  G4cout << G4BestUnit(mult*(atmHeight[32]+earthRadius),"Length") <<
G4endl;
  G4cout << "Atm33\t\t\t" << G4BestUnit(mult*atmHeight[33],"Length")
<< "\t\t\t";
  G4cout << G4BestUnit(mult*(atmHeight[33]+earthRadius),"Length") <<
G4endl;
  G4cout << "Atm34\t\t\t" << G4BestUnit(mult*atmHeight[34],"Length")
<< "\t\t\t";
  G4cout << G4BestUnit(mult*(atmHeight[34]+earthRadius),"Length") <<
G4endl;
  G4cout << "Atm35\t\t\t" << G4BestUnit(mult*atmHeight[35],"Length")
<< "\t\t\t";
  G4cout << G4BestUnit(mult*(atmHeight[35]+earthRadius),"Length") <<
G4endl;
  G4cout << "Atm36\t\t\t" << G4BestUnit(mult*atmHeight[36],"Length")
<< "\t\t\t";
  G4cout << G4BestUnit(mult*(atmHeight[36]+earthRadius),"Length") <<
G4endl;
  G4cout << "Atm37\t\t\t" << G4BestUnit(mult*atmHeight[37],"Length")
<< "\t\t\t";
  G4cout << G4BestUnit(mult*(atmHeight[37]+earthRadius),"Length") <<
G4endl;
}
```

Bibliography

[1] K. Greisen, “End to the Cosmic-Ray Spectrum?”, *Phys. Rev. Lett.* Vol. 16, P. 748750, 1966.

[2] T. O’Gorman, “Field Testing for Cosmic Ray Soft Errors in Semiconductor Memorie”, *IBM Research and Devolopment* Vol. 40, 1996.

[3] L. Schultz, “Image Reconstruction and Material Z Discrimination via Cosmic ray Muon Radiography”, *Nuclear Instruments and Methods in Physics Research Section*, Vol. 519, P. 687-694, 2004.

[4] H. Mavromichalaki, “Space Weather Prediction by Cosmic Rays”, *Advances in Space Research*, Vol. 37, P. 1141-1147, 2006.

[5] V. F. Hess, *Phys. Zeits.* Vol. 12, P. 998, 1911.

[6] G. Pfotzer, *Phys. Zeits.* Vol. 102, P. 23, 1936.

[7] R. Mewaldt, “Anomalous cosmic rays: Interstellar interlopers in the heliosphere and magnetosphere”, *EOS* Vol. 75, 1994.

[8] R. D. Blandford, “Particle acceleration by astrophysical shocks”, *The Astrophysical Journal* Vol. 221, P. L29, 1978.

[9] A. Simpson, “Elemental and isotopic composition of the galactic cosmic rays”, *Rev. Nucl. and Part. Sci.* Vol. 33, P. 323, 1983.

[10] M. Nagano, “Observations and implications of the ultrahigh-energy cosmic rays”, *Rev. Mod. Phys.* Vol. 72, P. 689732, 2000.

[11] T. Abu-Zayyad, “Evidence for Changing of Cosmic Ray Composition between 10^{17} and 10^{18} eV from Multicomponent Measurements”, *Phys. Rev. Lett.* Vol. 84, P. 42764279, 2000.

[12] M. Longair , “Particles, photons and their detection”, *PHigh Energy Astrophysics* Vol. 1, 1992.

[13] J. Cronin , “Cosmic rays: the most energetic particles in the universe”, *Rev. Mod. Phys* Vol. 171, P. s165-s172, 1999.

[14] A. Olinto, “Ultra high energy cosmic rays: the theoretical challenge”, *RPhysics Reports* Vol. 333, P. 329-348, 2000.

[15] M. Ryan, “Cosmic-Ray Proton and Helium Spectra above 50 GeV”,*Phys.Rev. Letters* Vol. 28, P. 985, 1972.

[16] N. Grigorov, *12th ICRC* , P. 1746, 1972.

[17] T. Burnett, “Extremely High Multiplicities in High-Energy Nucleus-Nucleus Collisions”,*Phys.Rev. Letters* Vol. 50, P. 2062, 1983.

[18] H. Makoto , "The ATIC Science Flight in 2002-03: Description and Preliminary Results", *28th International Cosmic Ray Conference* P. 1873, 2003.

[19] J. Wefel, "Primary Proton and Helium Spectra Observed by RUNJOB Collaboration", *28th International Cosmic Ray Conference* P. 1849, 2003.

[20] D. Muller, "Measurements with TRACER: Discussion of Results and Future Prospects", *29th International Cosmic Ray Conference* P. 101, 2005.

[21] T. Burnett, "Measurement of the Cosmic-Ray Energy Spectrum and Composition Using a Hybrid Technique", *Phys.Rev. Letters* Vol. 50, P. 2062, 1983.

[22] T. Abu-Zayyad , "Measurements with TRACER: Discussion of Results and Future Prospects",*Astrophysicsal Journal* V. 557, P. 686, 2001.

[23] K. Boothby, "The Dual Imaging Cerenkov Experiment (DICE) and the elemental composition in the Knee Region",*Nucl. Phys.* V. 60B, P. 1246, 1998.

[24] J. Fowler, "A Measurement of the Cosmic Ray Spectrum and Composition at the Knee",*Astropart. Phy.* V. 15, P. 49, 2001.

[25] G. Yodh, "Composition near the knee: results from the CACTI experiment",*Nuclear Physics B* V. 122, P. 239, 2003.

[26] T. Antoni, "Measurements of Attenuation and Absorption Lengths with the KASCADE Experiment",*Nucl. Inst.and Methods* V. A513, P. 490, 2003.

[27] M. Amenomori, "Multi-TeV Gamma-Ray Emission from the Carb Nebula Observed

with the new TIBET-III Air Shower Array",*27th International Cosmic Ray Conference* , P. 4, 2001.

28 M. Nagano, "The Knee: Theory and Experiment",*J.Phys.G*,V. 10, P. 1295, 1982.

29 K. Amako, "Present status of Geant4", *Nuclear Instruments and Methods in Physics Research A*, Vol. 453, P. 455, 2000.

30 "GEANTDetector description and simulation tool", *CERN Program Library Long Write-up W5013, CERN*

31 K. Amako, "GEANT4 and its validation", *Nuclear physics. B*, Vol. 150, P. 44-49, 2006.

32 K. Amako, "The GEANT4 toolkit capability in the hadron therapy field: simulation of a transport beam line", *Nuclear physics. B*, Vol. 150, P. 54-57, 2006.

33 M. J. Ryan, "Cosmic-Ray Proton and Helium Spectra above 50 GeV", *Phys. Rev. Lett*, Vol. 28, P. 985988 , 1972.

34 H. Burkhardt, "Monte Carlo Generator for Muon Pair Production", *CERN-SL-2002-016*, Note. 511 , 2002.

35 J. Apostolakis, "An algorithm to optimize the tracking of ionizing particles" , *CERN-OPEN-99-299*, 1999.

36 "Reference Manual at the Geant4"

37 M. A. Reading, "The Inventor Mentor: Programimng Object-Oriented 3D Graphics with Open Inventer", Addison-Wesley, Reading, MA, 1996.

[38] S. Tanaka, “Dawn for Geant4 Visualization”, *Proceesings of the CHEP ’97 Conferance*, Berlin, Germany, 1997.

[39] Bird, D.J. et al., “Direction of a Cosmic-Ray with Measured Energy Well Beyond the Expected Spectral Cutoff Due to Cosmic Microwave Radiation ”, *Astrophysical Journal*, Vol. 441, P. 144, 1995.

[40] A.M. Hillas, “Shower simulation: lessons from MOCCA”, *Nuclear Physics B*, Vol. 52, P. 29-42, 1997.

[41] J. Knapp, “Extensive air shower simulations at the highest energies”, *Astroparticle Physiscs*, Vol. 19, P. 77-79, 2003.

[42] D. Heck, “Recent Extensions to the Air Shower Simulation Program CORSIKA”, *Proc. 25th Int. Cosmic Ray Conference, Durban (RSA)*, Vol. 6, P. 245, 1997.

[43] S. Agostinelli, “Geant4a simulation toolkit”, *Nuclear Instruments and Methods in Physics Research A*, Vol. 506, P. 250, 2003.

[44] D. R. Lide, “Handbook of Chemistry and Physics”, 76 th edition.

[45] M. C. Coperchio, *Comp. Phy. Commun*, Vol. 110, P. 155, 1998.

[46] International Geomagnetic Reference Field (IGRF), URL http://www.ngdc.noaa.gov/IAGA/vmod/igrf.html.

[47] R. A. Wolf, “Magnetospheric configuration”, Introduction to Space Physics ed. M. G. Kivelson and C. T. Russell, Cambridge Univ. Press, p. 288-329.

[48] N. A. Tsyganenko, “Tsyganenko Magnetic Field Model and GEOPACK”. URL http://modelweb.gsfc.nasa.gov/magnetos/tsygan.html.

[49] N. A. Tsyganenko, “Global quantitative models of the geomagnetic field in the cislunar magnetosphere for different disturbance levels”, *Planet. Space Science*, Vol. 35, P. 1347, 1987

[50] N. A. Tsyganenko, “Modeling the Earth’s magnetospheric magnetic field confined within a realistic magnetopause”, *J. Geophys. Res.*, Vol. 100, P. 5599, 1995

[51] N. A. Tsyganenko, “Effects of the solar wind conditions on the global magnetospheric configuration as deduced from data-based field models”, *Eur. Space Agency Spec. Publ.*, ESA SP-389, P. 181, 1996

[52] W. H. Campbell, “Introduction to Geomagnetic Field“, second edition, Cambridge University press, 2003.

[53] A. M. Hillas, “Cosmic Ray”, *Pergamon Press*, 1972.

[54] M. A. Shea, “Cosmic Ray and Possible Climate Changes”, *28 th International Cosmic Ray Conference*

[55] H. Svensmark, “Influnce of Cosmic Rays on Earth Climate”, *Physical Review Letters*, Vol. 94, P 5030-5098, 1998.

[56] N. Marsh, “Cosmic Rays, Clouds, and Climate”, *Space Science Review*, Vol. 81, P 215-230, 2000.

[57] N. Marsh, “Solar Influnce on Earth’s Climate”, *Space Science Review*, Vol. 107, P 317-325, 2003.

[58] “Root” url http://root.cern.ch/

[59] “Condor” url http://www.cs.wisc.edu/condor/

[60] M. Risse, “Properties of Extensive Air Shower”, *Epiphany Conference on Astroparticle Physics*, 2004.

[61] W. Heitler, “Quntum Theory of Radiation”, *Oxford University Press, Oxford*, 2nd Ed., 1944.

[62] S. Khaled, “The Effect of Air Density on Wind Power Flux”, *"Wind Engineering"*, Vol 28, P 305-310, 2004

[63] Y. Fukuda, “Measurement of a small atmospheric Neutrino ratio”, *Physics Letter B*, Vol. 433, P 9-18, 1998.

[64] Y. Fukuda, “Evidence for Oscillation of Atmospheric Neutrinos”, *Physical Review Letters*, Vol. 81(8), P 1562, 1988.

[65] M. Honda, “Atmospheric Neutrino Fuxes”, *Physics Letters B*, Vol. 248, P 193-198, 1990.

[66] G. Barr, “Flux of Atmospheric Neutrions” *Physical Review Letters D*, Vol. 39(3), 1989.

[67] T.K. Gaisser, “Comparison of atmospheric neutrino flux calculations at low energies”, *Phys. Rev. D*, Vol. 54, P 5578, 1996.

[68] M. Honda, “Calculation of the flux of atmospheric neutrinos”, *Phys. Rev. D*, Vol. 52, P 4985, 1995.

[69] J. Kremer, “Measurements of Ground-Level Muons at Two Geomagnetic Locations”, *Physical Review Letters*, Vol 83(21), 1999.

[70] M. Motoki, “Precise Measurement of Atmospheric Muon Fluxes at Sea Level”, *Proceedings of ICRC 2001*, P. 927, 2001

[71] V.B. Anikeev, “Pair meter technique measurements of horizontal muon energy spectrum”, *Proceedings of ICRC 2001*, P 958, 2001

[72] S. Tsuji, “Atmospheric muon measurements I: Vertical measurements”, *Proceedings of ICRC 2001*, P. 931, 2001

[73] W.R. Frazer, “Limiting Fragmentation the Charge Ratio of Cosmic Ray Muons”, *Physcial Review D*, Vol.5, P. 1653-1657, 1972

[74] R.I. Golden, “Performance of a Balloon-born Magnet Spectrometer for Cosmic Ray Studies”, *Nuclear Instrument in Physics Research*, Vol A306, P. 366-377, 1991.

[75] T. Sanuki, “Precise Measurement of Cosmic Ray Proton and Helium Spectra with the BESS Spectrometer”, *Astrophysical Journal*, Vol. 545, P. 1135, 2000.

[76] M.J. Rayan, “Cosmic-Ray Proton and Helium Spectra abobe 50 GeV”, *Phy. Rev. Lett.*, Vol 28, P. 985-988, 1972.

[77] S.N. Nikolsky, *Sov. Phys. JETP*, Vol. 60. P. 10, 1984.

[78] J.D. Bjorken, “Some Implications of a New Source of Cosmic-Ray Muons”, *Phys. Rev. Lett.*, Vol. 184, P. 1345, 1969.

[79] L. Mikaelyan, “Status of three-neutrino oscillations after the SNO-salt data”, *Nuclear Physics B*, Vol. 91, P. 120, 2001.

[80] M. Maltoni, “Future reactor neutrino oscillation experiments at Krasnoyarsk“, *Phys. Rev.*, Vol. D68, P. 113010, 2003.

[81] F. Juget, “Magnetic moment measurement of electron type neutino from the MUNU experiment”, *Nuclear Physics B*, Vol. 138, P. 337, 2005.

[82] B. Neganov, “Status of the Experiment on the Laboratory Search for the Electron Antineutrino Magnetic Moment”, *Phys.Atom.Nucl.*, Vol. 64, P. 2033, 2001.

[83] J. Abraham , “Properties and performance of the prototype instrument for the Pierre Auger Observatory”, *Nuclear Instruments and Methods*, Vol. A523, P. 50, 2004.