TESTING AND VALIDATION OF THE MAGNETOSPHERIC SPECIFICATION MODEL

THESIS

Clark M. Groves, Captain, USAF

AFIT/GAP/ENP/95-06

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY
AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

UNABLE TO GET PAGES 60 THRU 92. COULD NOT LOCATE CAPT CLARK M. GROVES, PER THE POC FOR AFIT STUDENTS. POC: MARY ELLEN OGLE 1-800-543-3577, EXT 2021 AIR UNIVERSITY AT AFIT, WRIGHT-PATTERSON AIR FORCE BASE, OHIO

MAY 31, 1996

AFIT/GAP/ENP/95D-06

TESTING AND VALIDATION OF THE MAGNETOSPHERIC SPECIFICATION MODEL

THESIS

Clark M. Groves, Captain, USAF

AFIT/GAP/ENP/95-06

"The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government."

AFIT/GAP/ENP/95D-06

TESTING AND VALIDATION OF THE MAGNETOSPHERIC SPECIFICATION MODEL

THESIS

Presented to the Faculty of the Graduate School of Engineering

of the Air Force Institute of Technology

Air Education and Training Command in Partial Fulfillment of the

Requirements for the Degree of Master of Science

Clark M. Groves, B.S.

Captain, USAF

December 1995

AFIT/GAP/ENP/95D-06

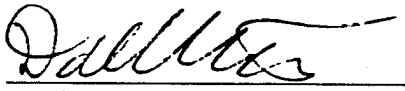
TESTING AND VALIDATION

OF THE

MAGNETOSPHERIC SPECIFICATION MODEL

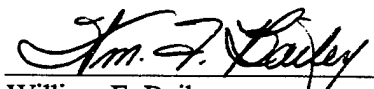
Clark M. Groves, B.S.
Captain, USAF

Approved:



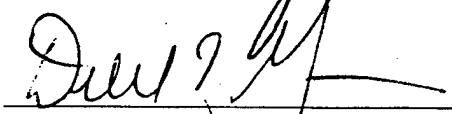
6 Dec 95

David E. Weeks
Chairman, Advisory Committee



6 Dec 95

William F. Bailey
Member, Advisory Committee



6 Dec 95

Derrill T. Goldizen
Member, Advisory Committee

Preface

The goal of this work was to establish a valid, independent Air Force site for accuracy testing of the Magnetospheric Specification Model (MSM). This report is limited in scope to a discussion of the construction and authentication of the testing platform at the Air Force Institute of Technology, along with presentation of an initial, limited accuracy study.

It is my experience that the details and difficulties associated with MSM research are overwhelming to the beginner. This paper is intended to document in one place some of the foundational information necessary to begin MSM research. It is written from the perspective of a complete novice, someone who began knowing nothing at all about the subject. It is written is a style, and includes details, that will hopefully make it possible for another novice to attain a working knowledge of the model in short order. Those who already understand MSM may find it simplistic. They, however, will be able to determine whether all the requisite issues have been documented, and are cheerfully invited to supply constructive criticism.

The bulk of my four months of research time was invested in contacting scientists with MSM expertise, gathering and documenting information, installing and testing software, and acquiring and validating databases. In short, I had to establish a credible testing platform for the model at AFIT. The bulk of model validation remains to be done. I feel very strongly that the potential exists for another AFIT student to significantly extend the analysis of MSM beyond the accuracy study presented in Chapter 5.

I would like to thank my advisor Dr. David Weeks for his unfailing optimism and zeal for learning which he displayed during the course of my thesis work. Special thanks also go

to Dr. Robert Hilmer, Dr. Bonnie Hausman, and Capt. Tom Smith for prolonged patience and expert assistance in answering hundreds of questions. The members of my committee, Dr. William Bailey and Capt. Derrill Goldizen provided needed corrections, guidance and encouragement, for which I am grateful. The AFIT Physics Department generously allowed for TDY travel to Falcon AFB and Phillips Geophysics Laboratory, as well as to an Air Force Space Model Review Meeting. Without that support this research effort would not have be possible. The many scientists and officers whom I met at these places all added to my understanding of the complex task of mastering MSM.

At the outset of this effort, my computer knowledge was infamously poor. Through the assistance of persevering friends such as Charlie Brennan, Roy Calfas, Matt Smitham and Karyl Davis, I was able to come up to speed quickly. I owe them many thanks.

Capt. Cliff Dungey devised the original idea of MSM research at AFIT. For that, and the many other things he has done to support the AFIT Space Physics program, a big thanks.

Many people in the Dayton community have been warm hosts, especially the members of Believers Assembly. Eric and Naomi Herup are one of a kind friends. Thank you.

My wife Linda deserves the warmest and most sincere thanks of all. She managed to support me in grueling times during our 18 month stay at AFIT, while loving and caring for our six children, one on whom was born during my thesis quarter. Each of our children have been mighty good troopers during the whole ordeal too. I love them all dearly.

Finally, I give thanks to Almighty God for His sustaining grace, in every circumstance.

Clark M. Groves

Table of Contents

Page

Page

List of Figures

List of Tables

Page

Abstract

The Magnetospheric Specification Model (MSM) is a computer simulation which specifies energetic charged particle fluxes in the earth's magnetosphere. Developed by Rice University for the United States Air Force, it is a first of its kind model. Earth's magnetosphere is of enormous size and complexity, and is the hostile operational arena for a wide variety of critical military hardware. Satellites are especially subject to environmental damage by peaks in energetic charged particle flux induced by solar wind dynamics. MSM receives ground based magnetometer data, solar wind data, and direct and modified in situ data as inputs. It then simulates the magnetohydrodynamics of particles and fields and computes the flux of particles between 2 and 10 Earth radii in the energy range 1 to 100 keV. Comparison of model flux output with in situ particle flux measurements yields an error estimate of the model's simulation. Selective variation of input data quantifies model sensitivity to data availability. A study of model error as a function of time reveals a diurnal error peak, indicating a model weakness in a particular magnetic local time zone. Differentiation of error as a function of magnetometer indices reveals model accuracy sensitivity with respect to geomagnetic activity.

TESTING AND VALIDATION OF THE MAGNETOSPHERIC SPECIFICATION MODEL

1. INTRODUCTION

The Magnetospheric Specification Model (MSM) is a computer simulation which specifies energetic charged particle fluxes in the earth's magnetosphere. It was the first operational computer simulation of the space environment in history. MSM was developed in response to a vital and growing military mission need and represents the genesis of a revolution in spacecraft mission support.

1.1 Air Force Mission Need

Over the past 30 years, many crucial military systems have been developed which operate in the near-earth space realm. This charged particle environment impacts the lifetime and reliability of satellites through radiation effects and spacecraft charging. Many instrument anomalies occur, and spacecraft operators and systems engineers must be able to rapidly isolate causes and implement corrective actions. In addition, military satellites may someday be targets during wartime, even though forbidden by international treaty. A warfighting commander must be able to rule in or rule out, with a high degree of confidence, any possible adversarial strikes against our assets. Delineating satellite damage due to enemy countermeasures from environmentally induced damage is imperative. However, these actions are possible only if there exists a means to specify the satellite operation environment.

During the rapid development era of space operations, there has been an absence of analysis capability for the space environment. There are very few platforms devoted to

providing measurements of Earth's space plasma environs, especially relative to the size of the system. The cost of significantly increasing this number is prohibitive. The only realistic option for specifying and eventually forecasting space conditions is via computer modelling. The need for an operational suite of space models as a force enhancer is immediate.

1.2 Solar-Terrestrial Environment

A basic understanding of the solar-terrestrial environment is necessary on the part of the reader. Its detailed description is outside the scope of this document. An overview of the subject, limited to a phenomenological discussion of aspects relevant to understanding charged particle flux patterns in the magnetosphere, is provided in the Appendix.

1.3 Space Model Development History

Air Force space model research is in its infancy. The Magnetospheric Specification Model (MSM), the focus of this study, became operational in July 1995 as the first operational computer simulation of the space environment in history. A brief overview of its programmatic development is instructive as a background to the current research effort.

US Space Command was created in 1985. The need for models to specify the space environment quickly came to the fore. In response, Air Weather Service (AWS) in 1986 initiated a research and development project known as the Space Environmental Technology Transition (SETT) program under the tutelage of Col. Thomas Tascione. This is an incremental plan to develop seven independently operating "building block" models of the space environment from the sun to the earth. A second plan exists, the Integrated Space Environmental Model (ISEM), to develop an executive system which fully couples all the models and integrates them with the 50th Weather Squadron's (50WS - formerly Air Force

Space Forecast Center) real-time space and geophysical database at Falcon AFB, CO. 50WS will be the end-user of the models in service to Space Command operations.

SETT program management at AWS falls under the office of Directorate of Technology, Plans and Programs, HQ AWS/XOX. From the outset, responsibility fell to Phillips Laboratory Geophysics Directorate, Hanscom AFB (PL/GP), for basic research and development of the SETT suite. Oversight of the MSM at PL fell under Dr. Mike Heinemann (PL/GPSG). In 1986, PL contracted Rice University to design the model in accordance with Air Force Needs Statement 0586. Dr John W. Freeman, et. al., of the Department of Space Physics and Astronomy undertook the obligation. The code was subsequently delivered by Rice, and MSM became operational at 50WS in July 1995, after user interface software development by Hughes STX Corporation, Colorado Springs, under the direction of Dr. Verne Patterson.

The Rice developers conducted some testing of the model as an integral part of R&D. 50WS desired independent testing. From 1992-1994, the United States Air Force Environmental Technical Applications Center (ETAC), tasked by Phillips Lab, undertook a limited sensitivity test on MSM. Responsibility fell to Capt. Tom Smith of the Simulations and Techniques Branch (SYT) . Although significant work was done, the study was never completed and a final report was never published. Tasking for ETAC analysis of SETT models has since been terminated. The progress made served as the foundation for this current work, as discussed later in this report.

50WS analysts, under the supervision of Mr. Kevin Scro, subsequently attempted to accomplish limited studies of their own on MSM. This proved to be an untenable option due

to the pressures of 50WS's operational mission and manning cuts for technical officers.

MSM development has spanned many years. Expertise and resources relating to the model exist in a complex network of scientists and officers at many locations. Comprehension and compilation of this knowledge was a major effort at the outset of this project. It is useful for a follow-on researcher to understand the chronology of MSM evolution and form working contacts with the sites and participants involved. AWS/XOX convenes a semi-annual Space Model Review Meeting, to provide the scientists and agencies within the Air Force space modelling community with a forum for face to face interaction.

1.4 Research Objective and Scope

The goal of this work was to establish a valid, independent Air Force site for accuracy testing of the Magnetospheric Specification Model. This report is limited in scope to a discussion of the construction and authentication of the testing platform at the Air Force Institute of Technology, along with presentation of an initial, limited accuracy study, with recommendations for further analysis.

This work serves as the basis for a complete analysis of the MSM. The report is intended to document foundational information necessary to achieve credible research. In addition, it is strongly hoped that this project inaugurates the official sanction and funding of independent beta testing of other SETT models at the Air Force Institute of Technology.

1.5 Air Force Impact

One weak link surfaces in the Air Force space model formation process. Currently, no designated site exists for independent validation studies. Creating a SETT model test site is a monumental task. In the final analysis, however, independent, operations oriented feedback

is essential to the development of space models which will best accomplish the Air Force mission. Since MSM is the most developmentally mature of the SETT suite, it was chosen as the vehicle for establishing the pattern of AFIT / SETT model testing.

The in-residence AFIT graduate space physics program is uniquely positioned to join the Air Force space model development team with exceptional value added capacity. Officers engaging in model related thesis work foster invaluable networks in the research community and gain detailed knowledge of state-of-the-art models. Upon graduation, these officers go to Space Command assignments to become operational users of the models. These dynamics promise to promote synergism and clarity between research centers and operations centers rather than the proverbial strain. Moreover, AFIT provides AWS a beta test site without the overhead of subcontractors. Faculty/scientists and student/researchers are already in place.

An AFIT / SETT beta test site scenario is a win-win paradigm for every Air Force member on the team. AWS gains independent validation of their models at minimal cost; Air Force Space Command gains improved, tested models while simultaneously having its future officers trained as experts in new operations tools; research labs gain a unique and lasting link to operations feedback; AFIT expands its singular role of mission relevant research and training; AFIT officers maximize their mission tailored graduate education.

1.6 Scientific Overview of MSM (1)

MSM simulates Earth's magnetosphere and provides a basis for determining energetic particle fluxes at arbitrary positions and times. This simulation includes a representation of the electric and magnetic fields, as well as the analysis of particle fluxes in the magnetic equal-trial plane from 2 to 10 Earth radii (Re) for particle energy ranges between 1 and 100 keV.

Particles are moved according to bounce-averaged equations of adiabatic drift in the electric field. They can be mapped along equipotential magnetic field lines on or off of a coordinate grid in the magnetospheric equatorial plane. The electric field is determined by merging a convective model for the inner magnetosphere with a parameterized model of the ionospheric electrostatic potential distribution. The magnetic field representation used is a parameterized current-driven model that determines the magnetic field from the superposition of the earth's magnetic dipole moment and the magnetic fields caused by the major current sources: Chapman-Ferraro current, equatorial ring current, and the cross-tail current sheet.

MSM is designed to exploit all available environmental data resources at 50WS, where it operates. It accesses seven environmental input parameters: Kp index, Dst index, low latitude midnight boundary of the auroral zone, polar cap potential drop, polar cap potential pattern, solar wind velocity, and solar wind density.

MSM outputs magnetospheric electron and ion differential fluxes as functions of energy and position. In addition, the model provides precipitating electron fluxes in the auroral zone, as well as the ionospheric electrostatic potential.

1.7 Sequence of Presentation

Chapter 2 will present a review of a previous MSM validation study, followed by an overview of the general research approach adopted for this project. Chapter 3 will present concepts for understanding MSM code, related data and corollary software. Chapter 4 will discuss the issues which surfaced in constructing a valid research platform for the model. Chapter 5 highlights accuracy studies accomplished. Conclusions and recommendations for further study complete the report.

2. PAST AND PRESENT VALIDATION EFFORTS

2.1 Overview of Previous Validation Effort

The draft document Validation of the MSM, and some undocumented work done subsequently (acquired by personal communication), represent the uncompleted ETAC study (2). Those analyses have been presented orally at Space Model Review Meetings.

The preface of the ETAC draft identifies the tasking of the project as received from Phillips Laboratory: 1) establish the operating reliability of the MSM, 2) determine the model's accuracy, and 3) investigate the usefulness of the MSM as a satellite anomaly analysis tool for use at 50WS (2:iv). ETAC's draft report includes a chapter on each of these topics.

Private communication with the author indicates that tasking one was completed (3). Personal experience in the AFIT study also bears this out. ETAC used MSM version 3.0 and reported numerous run time problems. The present project used MSM version 5.0 and experienced none of the reliability problems related to version 3.0. ETAC's feedback to model developers led to the addition of useful error traps and improved operating reliability in subsequent MSM versions. This highlights the value of independent feedback. Operating reliability has improved to the degree that the AFIT research effort did not uncover any additional items in this area.

ETAC's efforts on tasking two were extensive in scope. However, difficulties were encountered in ETAC's initial execution of their accuracy studies. The accuracy results in the draft were questionable due to an inadvertent mishandling of input data. In addition, subsequent to ETAC's research, certain software routines used to produce input files from satellite data had undergone revision, requiring use of updated files in future studies.

These facts motivated AFIT researchers to choose accuracy testing as a primary, initial focus in the study of MSM. In order to shorten the learning curve for AFIT' research, ETAC graciously provided the components of their testing platform as a starting framework for this study. This included the input data files, MSM source code, in situ comparison data files, and post-processing analysis software they used. (Chapter 3 of this report defines these items.)

The chapter in ETAC's draft report on the usefulness of MSM as a satellite anomaly analysis tool indicates that they made some progress on tasking three. This is a critical area for further research. However, it was not possible to extend the initial research efforts at AFIT to include this tasking.

2.2 Assumptions About Previous Validation Effort

Since the current study was to be an independent AFIT accuracy analysis, nothing was taken for granted regarding the items received from ETAC. Instead, a series of reviews was undertaken to establish the validity of each component of the testing platform. Data files were updated, corrected, or replaced as necessary. Original sources were reacquired whenever possible. Post-processing software was also scrutinized. Programs were corrected as necessary, or in some cases, original software was written to accomplish a needed task. All of these issues are discussed in detail in Chapter 4.

2.3 Current Research Approach

The goal of this research effort was to establish an independent, up-to-date testing platform for MSM, unambiguously documenting the process undertaken, and ultimately performing credible accuracy studies on the model. This complex task had to be broken down into three manageable partitions as a general research approach: 1) conceptually understand

MSM code, along with supporting data and software, 2) identify and solve issues, problem areas and pitfalls related to establishing a valid testing platform, and 3) adopt and execute an accuracy testing strategy. Each of the next three chapters reports, in order, on how the triad of steps in the research approach was carried out. Before proceeding to cover those topics, Table 1 on page 10 is provided to prime the reader's understanding of the research approach.

TABLE 1
QUESTIONS AND ISSUES UNDERLYING THE RESEARCH APPROACH

QUESTIONS on MSM	ISSUES
How is MSM organized?	- Understanding a large, complex code.
What are model inputs?	- Where did input data come from? Are these reliable sources? - Is this raw environmental data? If not, what modifying algorithms were employed to create the input files? - Did the algorithms employed to make changes undergo revision, creating the need for updated input files? - Is data properly formatted? Proper units? Data gaps a factor?
How does one get the model to successfully execute?	- Does one have to modify the code to get it to execute? - Is the version of MSM code used a factor? - Are there any Fortran compiler or debugging issues? - Are there data storage, operability or automation issues? - How about a run-time comparison of computer platforms?
What are model outputs?	- What are the format and units of the output files? - Which ones are needed for accuracy studies? - Is any post-processing software needed? What kind?
TEST STRATEGY	**ISSUES**
What comparison data set is used to test the model's accuracy?	- Where did comparison data come from? Reliable sources? - Is this raw environmental data? If not, what modifying algorithms were employed to create the comparison files? - Were the techniques employed correctly? - Is data properly formatted? Proper units? Data gaps a factor? - Are issues related to the sensors gathering the data a factor?
What comparison methods are relevant?	- Which comparisons yield statistically relevant errors? - Which comparisons reveal operationally significant insights?
What software tools are needed to carry out the strategy?	- How do MSM output and comparison data get into equivalent units and format, and become equivalently interpolated in space and time? - Is the documentation related to the software complete and accurate?

3. UNDERSTANDING MSM (4)

3.1 Background

The Rice developers produced the MSM Scientific Description & Software Documentation manual as the exhaustive, general description of the model (4:2-2). The next two chapters of this report draw heavily from that document. However, an attempt has been made to condense and synthesize the material found there, based on lessons learned in this study. Some undocumented material obtained directly from experienced MSM scientists has been added. The result is intended to be a concise guide for a new MSM researcher on the specific issue of understanding how to work with MSM in the research setting.

The goal of this chapter is to present an overview of the primary components of the MSM testing platform. The focus is on functional and conceptual insights. The following topics will be covered: 1) understanding input data, 2) understanding MSM code, 3) understanding output data, 4) understanding satellite comparison data, 5) understanding post-processing analysis software, and 6) understanding miscellaneous utility software.

For clarity, it is important to state three things at the outset of this chapter. First, each of the six areas mentioned above is highly interconnected with the other areas. This means, for example, that in order to understand input data, you must first know something about MSM code. But in order to understand the code you need to know certain things about the input. The result is that ideas or terms introduced in one section may not be fully defined until a later section, where their description more naturally fits into the flow of the document.

Second, it is important to repeat that this chapter focuses primarily on a conceptual understanding of the components of the testing platform. Some technical questions raised in

the reader's mind by this chapter will be not be answered until the following chapter. Chapter 4 covers the same topics as Chapter 3, but with the emphasis on technical matters.

The third issue is a consequence of the first two, and relates to the use of tables to present what is essentially narrative material. The information needed to understand MSM is lengthy, interconnected, and technically detailed. A table format with bullet statements allows an MSM novice to quickly cross reference information, and review material at a glance. Therefore, it was often selected as the format of choice for material in Chapters 3 and 4.

3.2 Understanding Input

A natural starting point for understanding MSM is to look at model input. Functionally, three categories of input data exist: 1) operator specified input, 2) environmental input, and 3) static input. The functional distinction between the categories has to do with who or what determines the contents of the files in each group. For category 1, the user determines the content. For category 2, the environment dictates the content. For category 3, the model itself provides the file contents. Specifics on each of the three input types follows.

3.2.1 Operator Specified Input Category

Table 2 provides an overall description of the operator specified category of inputs. Table 3 follows with detailed information on the individual files within the category.

TABLE 2
OVERALL DESCRIPTION OF OPERATOR SPECIFIED INPUT CATEGORY

File names: *msmin, enchan*
- these files must be created by the user, and are the main operator interface to the model - these files dictate to the model certain bookkeeping and controlling information

TABLE 3
DETAILS ON OPERATOR SPECIFIED INPUT FILES

msmin file	- dictates to the model the time window of interest for model simulation - tells the model whether the run is an initial start (cold start) , or a continuation run (hot restart) - provides input vehicle for identification headers and file prefix designations for each run -- helps keep multiple runs organized - provides the daily sun spot number - allows user to toggle on/off screen printout of results - provides a correction factor to the *pcp* input file - allows user to toggle on/off between Kp only environmental data mode, and full use of all available environmental data mode
enchan file	- dictates to the model the type of charged particle simulation desired, as well as the energy channels of interest for each particle type

3.2.2 Environmental Input Category

The environmental category of input data includes eight files. Table 4 names the files and gives an overall description of the input category, using bullet statements which apply to all files. Tables 5 and 6 then give specific details of each of the individual files.

TABLE 4
OVERALL DESCRIPTION OF ENVIRONMENTAL INPUT CATEGORY

File names: *fkp , dst , eqedge , pcp , xipatt , swden , swvel , sumkp*
- the contents of these files must be acquired by the user from various national data archival sources or scientific centers (named in Table 6) - the data must be properly formatted by the user and placed into files for use by MSM - the files contain the environmental input data needed by the model as parameters for the simulation of the magnetosphere - the data in each file must span the entire time window of interest for the model run - gaps in the data stream are allowed, intermittent or extended, except for the *fkp* file - in the Rice manual these are called "event specific" inputs (4:2-4), i.e. they often are data records related to specific environmental events, such as geomagnetic storms - at 50WS, MSM will access their Environmental Data Base, real time, for these parameters

TABLE 5
DETAILS ON ENVIRONMENTAL INPUT FILES

FILE NAME	INPUT PARAMETER	DESCRIPTION OF PARAMETER	UNITS	TIME TAG	SENSOR
fkp	Kp	- average global magnetic activity index (Göttingen)	none	every 3 hours	magneto-meters
dst	Dst	- Disturbance Storm Time Index, or equatorial ring current activity index	nano-tesla	every 1 hour	magneto-meters
eqedge	Equatorward Auroral Edge	- low latitude midnight boundary of the aurora	degrees latitude	~ every 1 hour	DMSP * SSJ/4
pcp	Polar Cap Potential	- electric field potential drop across the polar cap	kvolts	~ every 1 hour	DMSP * SSIES
xipatt	Cross Polar Cap Potential Pattern	- Heppner-Maynard Model polar cap pattern (4:2-18)	none	~ every 1 hour	DMSP * SSIES
swden	Solar Wind Density	- solar wind density	protons / cm^3	hourly average	IMP-8 *
swvel	Solar Wind Velocity	- solar wind velocity	km / sec	hourly average	IMP-8 *
sumkp	Sum of Kp's	- sum all the Kp values for a day; list 10 day's worth	none	every 1 day	derived from Kp

* DMSP and IMP-8 are names of satellites. SSJ/4 and SSIES are sensors onboard DMSP. No details on these items are included in this report. Further information is available in the MSM Scientific Description & Software Documentation manual (4:2-4) as well as in the ETAC draft document (2:3) and a paper by Heelis and Hairston (10).

3.2.2.1 Environmental Input Time Tagging

All data points in the environmental input files are marked by their time of occurrence (they are said to be 'time tagged'). Time tagging in each file is independent from that in other files. The model also keeps track of its time independently of any of these time tags. It begins

TABLE 6
FURTHER DETAILS ON ENVIRONMENTAL INPUT FILES

FILE NAME	INPUT PARAMETER	USED IN THE MODEL FOR THE PURPOSE OF:	ORIGINAL SOURCE FOR DATA IN FILE
fkp	Kp	- establishing initial and ongoing boundary conditions for the particle distribution functions - generating proxies for missing data in other files using front-end models - generating replacement flux values using default model (optional)	- National Geophysical Data Center (NGDC) via WWW*
dst	Dst	- determining ring current strength in the magnetic field model	- NGDC via WWW
eqedge	Equatorward Auroral Edge	- determining auroral zone boundary in the electric field model - constraining the magnetic field mapping	- Phillips Lab (PL)
pcp	Polar Cap Potential	- input to electric field model	- University of Texas at Dallas (10)
xipatt	Cross Polar Cap Potential Pattern	- input to electric field model	- University of Texas at Dallas (10)
swden	Solar Wind Density	- determining standoff distance in the magnetic field model	- National Space Science Data Center (NSSDC) via WWW
swvel	Solar Wind Velocity	- determining standoff distance in the magnetic field model	- NSSDC via WWW
sumkp	Sum of Kp's	- determining high energy fluxes	- derived from Kp

* WWW stand for World Wide Web.

its simulation at the time tag declared by the user in the *msmin* file to be the model 'start time', then moves forward in 15 minute intervals. Environmental input data are required at each model time step. However, input file time tags are generally not simultaneous with the model time steps. Therefore, subroutine *pargen* within the MSM code performs a linear interpolation between nearest neighbor data points in the input files, and those are the actual values used by MSM at each simulation time step.

3.2.2.2 Environmental Input Operating Modes

Before proceeding to section 3.2.3 and the static category of input files, it is important to cover a confusing issue regarding the model's use of the environmental input files. It would be advantageous to the reader at this juncture to refer to Table 15 in Chapter 4. That table and the narrative which follows it, include a description of line 9 of input file *msmin*, which is pertinent to this discussion.

Table 15 reveals that MSM is capable of operating in three input modes. Logically, it makes sense that the optimal mode is that one which provides the model with all available environmental data parameters. However, MSM is capable of executing with data gaps in individual files, or with one or more entire files missing, to the point of using Kp as the single, default input. There are significant differences in the way in which the model functions internally in the three input modes. The next three sections attempt to make sense of this issue. The reader should pay particular attention to what the Kp parameter is used for in the model in each case.

3.2.2.2.1 Full Data Mode

If the user directs MSM to access all inputs, the model employs a subroutine called

indata to read all Kp values into memory arrays, along with every other available environmental inputs. In the perfect scenario, all environmental input files would be available to the model, with no data gaps in any of the files. In this case, MSM would use Kp inputs for only two purposes. First, it would initialize the particle distribution conditions across the model grid using an empirically based Kp algorithm. Then, as the MSM run progressed, the model would use this algorithm to establish flux conditions at the model boundary at each new time step. Only non-Kp data would be used as input parameters for the electric and magnetic field models, so that the plasma distribution algorithms could continuously update the time dependent particle flux values across the model grid.

The perfect scenario is usually modified to some degree. It might happen that all input files are present, but some of them have data gaps. It might also happen that one or more files are completely missing (other than *fkp*), with those remaining containing some data gaps. Whatever the case, as long as the user has directed MSM to access all inputs, everything available will be used by the model as described in the previous paragraph. The model handles the data gaps as described in the next paragraph.

In full data mode, input data files are tested for gaps in the data stream by a subroutine called *pargen*. If data are missing from a given parameter, the model responds in one of two ways. If the time gap is small, values are assigned by simply extending the linear interpolation between neighboring values in the data file. However, if values are missing for a time gap in excess of a set length, the model employs empirical 'front-end models' based on Kp alone to generate proxies for that parameter during those time steps. These values are fed to the electric and magnetic field models to continue the determination of time dependent particle

flux distributions. Everything else in the model run functions as before.

3.2.2.2.2 Kp Only Mode Using Front-End Models

The most extreme modification of the perfect scenario above would be to remove all input files from the run directory except *fkp*, and yet direct the model to access all input data. In this case, the complete lack of data would simply be treated as multiple extended data gaps, and the front-end models would compute empirical proxies for all the missing data as described above. The electric and magnetic field routines and particle distribution algorithms would operate exactly as before. The only difference would be that they would be using only proxy values, derived empirically from Kp.

Rice provided a specific operational mode to allow for this option without having to actually remove any environmental data files from the run directory. Option 1 in line 9 of Table 15 is this option. It essentially turns off the read statements in the *indata* subroutine for everything but the *fkp* file. This was done because one of the original Air Force design specifications for MSM was that it had to be robust , i.e. it had to be capable of operation even with a catastrophic loss of data to the point of being left with Kp alone (4:2-7). This option allows easy testing of the model in this mode.

3.2.2.2.3 Kp Only Mode Using Default Model

In the event of catastrophic data loss for the entire simulation period, another alternative exists in MSM. It is option 2 in line 9 of Table 15. This option employs 'default models' (4:2-7). Although these models are based on Kp only, they must not be confused with the front-end routines described above. Default models completely bypass the processes described above in which the front-end models play a role. No proxies are determined for

missing parameter values. No electric and magnetic field algorithms are used. No particle distribution functions are employed. Instead, flux outputs are determined empirically, directly from Kp. These default models are average models and contain no intrinsic time dependence other than that derived from the variation in Kp. They totally replace MSM time dependent computed flux output.

3.2.3 Static Input Category

There are six files in the static input category. Table 7 names the files and gives an overall description of the input category, with bullets which apply to all files. Since the model user is not required to manipulate these files in any way, no table is presented with detailed information on the individual files.

TABLE 7
OVERALL DESCRIPTION OF STATIC INPUT CATEGORY

File names: *coord , dktable , efcoef , hardy , ioneng , ionnum*
- these files are supplied with the code and do not change (static) - they contain coefficient data used to set up the coordinate grid of the model or they contain coefficient data used by the front-end and default models when gaps exist in the environmental data files - they can be considered an integral part of the model

3.3 Understanding MSM Code

It is not feasible to present the details of MSM code in this report. Instead, coverage is limited to providing an overview of the topic. The only details discussed in this chapter are those needed as background information for understanding the issues and problems raised in Chapter 4 or the accuracy results presented in Chapter 5.

3.3.1 Overview of MSM Code

MSM source code is written in Fortran 77. It consists of a highly modularized aggregate of more than 10,000 lines, with 90 program subroutines. Seven major functional areas exist. As a reference, Table 8 lists the seven functional areas and the subroutines associated with each one. That is followed by Figure 1, which provides a simplified flow chart of the structure of MSM.

TABLE 8
CATEGORIZATION OF MSM SUBROUTINES

MAJOR FUNCTIONAL AREA	SUBROUTINE NAMES
Control Routines	*msmcon , pptcon , ebcon , hiepar*
Magnetic Field Model	*btrace , bfgyro , fndbrk , getmat , loadbm , mexist , rmvbsh , zerobm*
Electric Field Model	*efield , aurora , efbndy , efloc , emodel , epot , fndbnd , low , regl , thet*
Particle Tracer and Loss Algorithms	*pptm , bndchk , bndset , cexrat , dvefdi ,rk4 , dvefdj flxcal , flxnit , flxtrp , flxval , inital , mover , opterr , rlyons , setalm , setref ,tchk , setsct , thrcal , vlocty , vmlset ,vmnom ,wkrate*
Input and Output Data Handling Routines	*dtachk , dtntrp , dtxipt , indata , output , pargen , smooth , stndof*
Utility Routines	*blockdata , et2flx , flx2et , fndbi , g3ntrp , g3trpa , grid , mltset , modflx , outp , pfix , rdhdr , rdstfl , read3d , tconv2 , tconv3 , timinc , tnorml , usrtim , wrtvec*
Front-End and Default Models	*addhdy , aurl1 , aurl2s , clpdfl , dstdfl , efun , enet, eqtdfl , flxdfl , fsum , patdfl , pcpdfl , regen , stndfl , tilt*

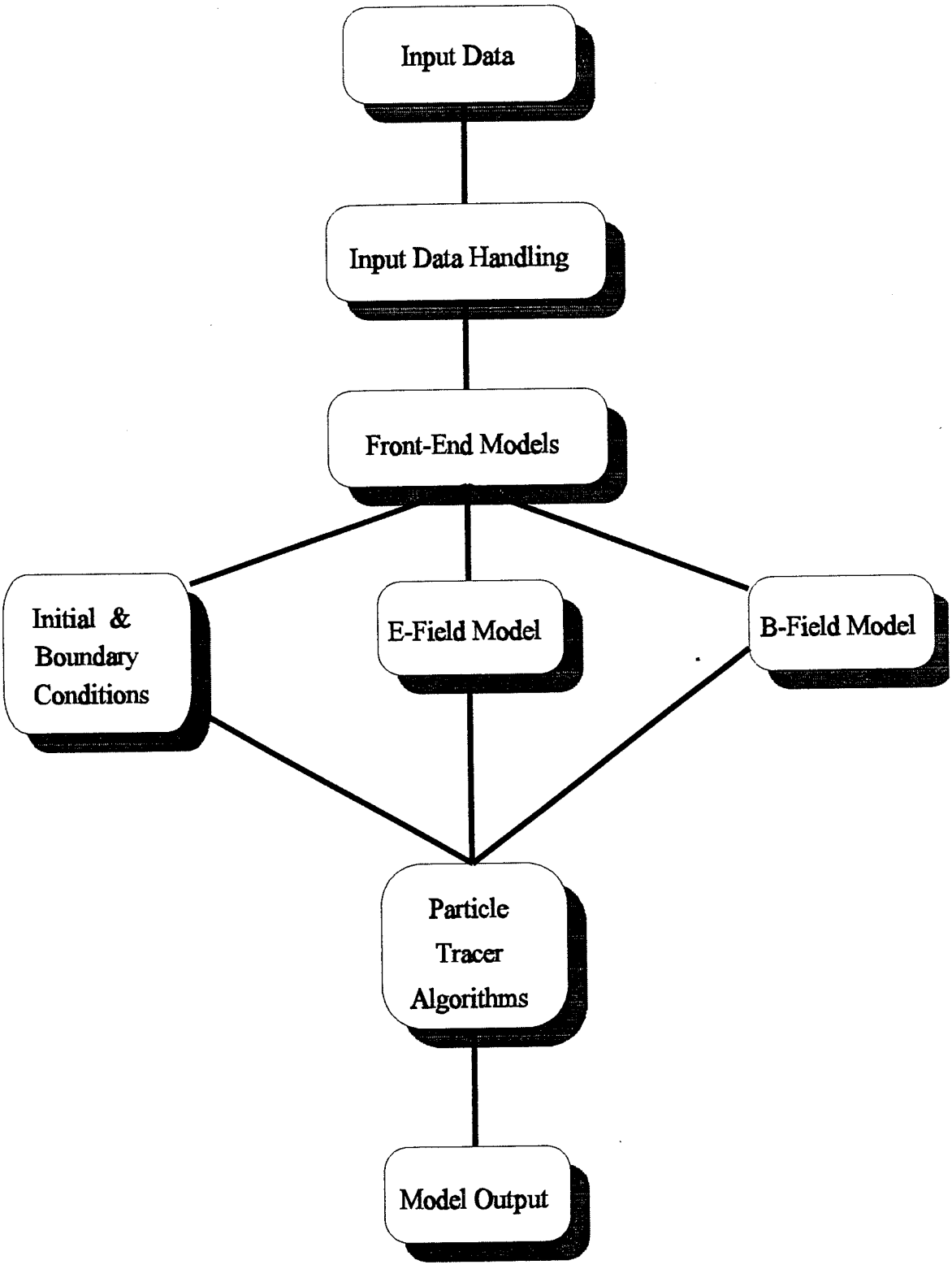


Figure 1. MSM Simplified Flowchart (4:2-59)

3.3.2 Primary Control Routine

The main control routine in the code is named *msmcon*. Here MSM reads the *msmin* and *enchan* files to set its control parameters, including reading in the desired times for starting and stopping model simulation. As mentioned earlier, the model has a nominal fifteen minute time step. The highest number of consecutive simulations allowed in any contiguous computer run is fixed in *msmcon* at fifty. Hence, a maximum limit of twelve hours of model simulation is possible for any single computer run.

At the start time tag, the model establishes the initial particle distribution function using the default models. This is termed a cold start. During a contiguous computer run, the particle flux simulations at each of the subsequent steps use the particle distribution results of the previous step as beginning conditions. The original conditions set at the start time tag by the default models are not extremely accurate, and it takes several consecutive time steps to wash out the effect of the cold start. For this reason, the first few hours of model simulation after a cold start are not considered reliable (5).

During a contiguous computer run, the model automatically accesses the required results of the previous time step, proceeding through each fifteen minute time tag in the data stream until the one declared for stopping the simulation is reached. At this point, model execution ceases. To continue an uninterrupted analysis without the introduction of a new set of cold start errors, a 'hot restart' routine must be accomplished. Details on this procedure will be explained in the model setup section, 4.2.4.

3.3.3 Magnetic Field Model

The magnetic field routine in the model is an algorithm flexible enough to represent a

wide variety of magnetospheric conditions. Nearly 1000 permutations of magnetic field configuration exist in a 350 Mb lookup table. The model uses three independent environmental input parameters to determine the choice of magnetic field configuration: 1) magnetopause standoff distance, 2) Dst, and 3) Equatorward edge of the auroral boundary.

MSM does not include the effect of the earth's magnetic dipole tilt. This means that the model has built into it the assumption that the magnetic and geographic equatorial planes always coincide. This will become a factor when accuracy testing is discussed.

3.4 Understanding Output

MSM produces a set of outputs at each fifteen minute model time step. Over the course of a run, they accumulate in twenty one files, mostly in a binary output format. Table 9 lists the names of the output files according to category.

TABLE 9
CATEGORIZATION OF MSM OUTPUT FILES

CATEGORY	FILENAMES
Particle Output	*bndloc , flux , flxbnd , hieout , ishift , vm*
Magnetic Field Output	*bmin , xmin , ymin , zmin*
Electric Field Output	*v , vnorth , vsouth*
Auroral Precipitation Output	*eavg , flxsum , ipiflx , ipieng*
Utility Output	*aloct , augpar , colat , errfile*

Many of the files above contain output results which were not needed in this study. Therefore, the only files examined in detail in this chapter are those which require background

discussion for understanding issues raised in Chapter 4, or accuracy results presented in Chapter 5.

3.4.1 Output File *errfile*

An important file for the beginning user is the *errfile* in the utility category. MSM has a significant array of error trapping routines. The *errfile* file is the one to which the model writes diagnostic and informational output as it executes, always in ASCII format.

3.4.2 Output Files Used for Flux Studies

Only six of the output files listed in Table 9 were needed for the accuracy studies performed. Model flux outputs are compared with in situ, geosynchronous satellite flux readings. The primary model output file needed for this is the *flux* file, since it contains differential flux values for all model time steps (in units of particles / cm^2-sec-ster-eV). MSM determines results on a grid in the magnetic equatorial plane. The grid has 62 latitudinal grid lines and 51 longitudinal grid lines. In order to compare model outputs with in situ satellite readings, post-processing software is needed to interpolate MSM results to points in space along a geosynchronous orbit. The model must thus provide additional information to the post-processing software to allow for this interpolation. The software (*flxintrp.f*) and procedure for doing this are discussed in Chapter 4. The six files needed by the interpolation software are listed in Table 10.

TABLE 10
FILES NEEDED FOR GEOSYNCHRONOUS FLUX DETERMINATIONS

aloct , bndloc , flux , xmin , ymin , zmin

3.5 Understanding Satellite Comparison Data

MSM satellite comparison data for this study came from in situ readings on two DoD geosynchronous earth orbiting spacecraft, located at 15 and 70 degrees east longitude. Roughly speaking, geosynchronous satellites orbit at 6.6 Re with the same period as the earth, remaining in the geographic equatorial plane over a single location on the equator. Thus, such a satellite can be considered to be fixed to a given meridian as the earth rotates. Every 15 degrees of longitude equates to one hour of planetary rotation.

As mentioned above, accuracy testing in this study was performed through post-processing of output files. It should be mentioned that in situ geosynchronous satellite data can be optionally specified as input data to MSM at run time. The two files provided to the model must be named *epsat*, for energetic electron fluxes, and *epions*, for energetic proton fluxes. If included at model execution time, MSM uses the data in subroutine *opterr* to perform a simple error analysis of model output. However, the nature of this analysis was not sufficient for the accuracy testing desired in this study. Rather, *epsat* and *epions* were employed with post-processing software discussed in the next section.

The raw geosynchronous satellite data used in this study to verify MSM output fluxes was originally acquired by ETAC, through the Los Alamos National Laboratory and Air Force Global Weather Central. AFIT acquired it from the ETAC database.

The ETAC draft report indicates that the data are five minute average particle counts of electrons and protons over broad kinetic energy bands. The sensor look angle selected for both particle species was that perpendicular to the magnetic field lines, i.e. the satellite equatorial direction (2:7).

3.6 Understanding Post-Processing Analysis Software

The satellite data must be converted from particle counts to differential fluxes to make it comparable with MSM outputs. The last section mentioned that model output must be interpolated to geosynchronous satellite locations. It is also necessary to perform temporal interpolations on the model output. In addition, both sets may require quality control to eliminate anomalous values.

Each such step is manifested in a piece of software. Since all these activities must be accomplished subsequent to, and exterior to the MSM run, this report coins the phrase "post-processing software" to describe them.

Tables 11, 12 and 13 list and define the concepts behind the post-processing software used in this study. Most of the software was developed by the Rice scientists, or in some cases, the concepts behind a piece of code were developed by them and coded by someone else. AFIT acquired the software from ETAC, where minor modifications had been made. Some programs were written at ETAC. One of the programs is original to AFIT, along with minor modifications of other programs.

Table 11 provides a synopsis of software used for post-processing of model output. Table 12 examines the software used to process satellite comparison data from its raw form to one which is comparable to model output. Table 13 explains the concepts behind the software used to actually calculate the RMS error of the model. The detailed description of these algorithms and the issues related to them are explained in Chapter 4.

TABLE 11
POST-PROCESSING SOFTWARE USED WITH MSM OUTPUT

PROGRAM NAME	DESCRIPTION	INPUTS NEEDED	OUTPUTS YIELDED
flxintrp.f	-this software takes six binary output files from the MSM run and interpolates spatially between model grid points to generate flux values at geosynchronous satellite positions for every fifteen minute model time step -output is in ASCII format -output is differential particle flux in the form: $\log_{10}$ (particles / cm^2-sec-ster-keV) -each output column identifies a particle type (chemical species and energy) -each output row identifies a time step -original program was *calflx.f*, developed by Rice, then modified and renamed by ETAC	*aloct,* *bndloc,* *flux,* *xmin,* *ymin,* *zmin*	*modflux15** *modflux70*
qcmsm.f	-during certain environmental conditions, the model's boundary can contract to within that of a geosynchronous orbit -when *flxintrp.f* is executed, interpolated points falling outside the model boundary are given values of 9.999 -this software removes the 9.999 values -this software divides the model data into separate electron and proton files and formats it for use by the *compare.f* software -this program was written by ETAC, modified by AFIT	*modflux15* *modflux70*	*xXflux15.ele*** *xXflux15.ion* *xXflux70.ele* *xXflux70.ion*

* A '15' or '70' in a filename indicates the file contains model output interpolated to the orbit positions of geosynchronous satellites located at 15 and 70 degrees east longitude, respectively.

** A lowercase 'x' in a filename refers to a generic run prefix identifier, and an uppercase 'X' refers to the Kp data mode, both declared in the *msmin* file (see Table 15). An 'ele' indicates the file contains electron output, and an 'ion' indicates it contains proton output.

TABLE 12
POST-PROCESSING SOFTWARE USED WITH SATELLITE COMPARISON DATA

PROGRAM NAME	DESCRIPTION	INPUTS NEEDED	OUTPUTS YIELDED
epform.f	-this software takes original geosynchronous satellite files and reformats appropriately for use by the *epconv.f* software -units of the data at this point are (particles / cm^2-sec-ster-MeV) -this program was written by ETAC	*original geosynchronous satellite flux files*	*epsat.orig* epions.orig*
epconv.f	-this software takes the satellite data files and performs five operations: 1) initial quality control 2) sensor deadtime correction 3) differential energy channel calculations (convert particle counts to differential flux) 4) applies channel correction factors 5) reformats to Rice format *** -units of the data at this point: $\log_{10}$ (particles / cm^2-sec-ster-keV) -Rice developed the algorithms, the program was coded by ETAC, minor modifications and corrections by AFIT	*epsat.orig epions.orig*	*epsat.rev** epions.rev*
epqc.f	-this software takes the Rice format satellite data and performs three operations: 1) final quality control 2) divides the data into separate files for each satellite location and particle type 3) reformats data for use by *compare.f* -this program was written by ETAC, modified by AFIT	*epsat.rev epion.rev*	*epqcele15**** epqcion15 epqcele70 epqcion70*

* An 'ep' in the filename stands for 'energetic particles', while 'orig' stands for 'original data.'
** A 'sat' in the filename indicates the file contains electron data, while 'ions' indicates the file contains proton data, and 'rev' stands for 'revised data.'
*** A Rice format indicates that the files are ready for direct model input, for use in *opterr* subroutine error analysis (see Section 3.5). The *opterr* method was not used at AFIT.
**** A 'qc' in the filename indicates the data in the files is in its final quality controlled form. A '15' or '70' in these filenames indicates that the data originated from geosynchronous satellites located at 15 and 70 degrees east longitude, respectively.

TABLE 13
POST-PROCESSING SOFTWARE USED TO PERFORM RMS ERROR ANALYSIS

PROGRAM NAME	DESCRIPTION	INPUTS NEEDED	OUTPUTS YIELDED
compare.f	-this software compares modified MSM output fluxes with the modified in situ particle fluxes and calculates the squared difference between the two -this program was written at AFIT Basic Methodology** -in situ readings are at five minute intervals and MSM output is at fifteen minute intervals -temporal interpolations are performed between neighboring MSM flux values to create values with time tags which match the in situ flux time tags -the squared error is calculated between the two values at each time step***	*xXflux15.ele* *xXflux15.ion* *xXflux70.ele* *xXflux70.ion* *epqcele15* *epqcion15* *epqcele70* *epqcion70*	*xXerrele.15** *xXerrion.15* *xXerrele.70* *xXerrion.70*

* The 'err' in the filename indicates that these files contain the error calculations for the model run, as compared to in situ satellite data.
** The methods employed in all the software listed in Tables 11, 12 and 13 will be revisited, and fully explained in Chapter 4.
*** A separate file is output for each particle species and satellite location. Every file contains the results for all three energies studied for each particle species. For each energy studied, model flux, in situ flux and squared error are listed for each five minute time step.

3.7 Understanding Miscellaneous Utility Software

Several small computer programs are needed for various utility purposes in the complex process of completing an MSM accuracy analysis. These are in addition to the pieces of software already mentioned in this report. Table 14 below summarizes them. Each piece of miscellaneous utility software is completely separate from the MSM code, and must not be confused with the utility subroutines listed in Table 8.

TABLE 14
MISCELLANEOUS UTILITY SOFTWARE

PROGRAM NAME	DESCRIPTION
fixfiles.f	-reads in the MSM output *flux* file and rewrites only the most recent record for use in a hot restart (to save disk space) -resets *msmin* file times to run the model for the next 12 hour period -updates the *sumkp* file and sun spot number every 24 hours -written by ETAC using Rice concepts, minor modifications by AFIT
runmsm	-automatically executes the MSM model in background mode for a declared number of consecutive, contiguous runs, initiating the necessary post-processing programs, utility programs, and hot restart procedures required between runs -written by ETAC
convert.f	-takes *pcp* and *xipatt* input data files received from the University of Texas at Dallas and reformats them for proper reading by MSM -written by AFIT
dstmsm.f	-takes *dst* input data as received from national archival source (particular format) and reformats it for proper reading by MSM*
kpmsm.f	-takes *fkp* input data as received from national archival source (particular format) and reformats it for proper reading by MSM*

* These programs were written by R. Hilmer, a former Rice doctoral student, now at PL.

3.8 Chapter Summary

This chapter was written to facilitate a working understanding of the MSM code and its related inputs, outputs, and corollary software. A myriad of names and concepts have been introduced. In the complex process of setting up a testing platform for the model, it is evident that many opportunities would arise for introducing errors. The next chapter thoroughly explores such issues and presents the solutions implemented in this study to resolve them.

4. ISSUES CONFRONTED IN BUILDING AN MSM TESTING PLATFORM

4.1 Background

The last chapter presented information designed to develop a conceptual, working understanding of the Magnetospheric Specification Model. This chapter presents potential sources of error, a host of technical issues encountered, which must be laid to rest before one can unambiguously claim to have established a valid platform for accuracy testing of the model. Only then are the accuracy studies presented in Chapter 5 credible. In the end, given the limits of time and resources, certain assumptions are made. The goal is to limit these to defensible assumptions. All assumptions remaining as a part of this study are clearly stated.

The order of presentation in this chapter will roughly follow that of the last chapter. Topics covered will be the issues, problems and solutions surrounding the following: 1) input data, 2) MSM code, 3) output data, 4) satellite comparison data, and 5) post-processing analysis software. However, before proceeding to those topics, it is essential to provide the technical information needed to set up and execute a model run.

4.2 Model Setup and Execution

The MSM source code was given the name *msmwork.f* in this study. This obviously must be compiled to obtain executable code, called *msmwork* in this study. Actually, due to the length of the code and the number of subroutines, it is highly recommended that the source code be split into all its various parts, each part taking a *subroutine.f* name convention, with compiling accomplished using a Makefile. In the SUN UNIX environment, such splitting is trivial using a built in 'split' command. This procedure results in reduced time spent compiling, when small adjustments are made to particular parts of the source code.

All input files must be placed in the same directory as the executable code. This includes all three types referred to previously: operator specified, environmental, and static inputs.

It is useful to review the details of Table 3 at this juncture. Two operator specified input files introduced there must be created by the user: the *msmin* file and the *enchan* file. The next two sections present the mechanics of constructing them.

4.2.1 Setup of Operator Specified File *msmin*

The *msmin* file has nine lines. Table 15 below gives an example of the content of such a file, and describes each line. The narrative in this section provides the detailed information

TABLE 15
SAMPLE *msmin* FILE

LINE #	SAMPLE CONTENT	DESCRIPTION
1	1989 244 86400	model start time: year, Julian day, time (seconds)
2	1989 245 43200	model stop time: year, Julian day, time (seconds)
3	1	cold start or hot restart toggle: 0=cold, 1=hot
4	' MSM run one '	character identification for this run, up to 80 characters
5	145.0	daily sunspot number
6	' a '	file prefix identification for this run
7	0	ASCII printout control toggle: 0=no print, 1=print
8	1.00	*pcp* correction factor: 1.34=old SSIES algorithm, 1.0=new SSIES algorithm (see reference 10)
9	0	environmental data input mode: 0=use Kp plus all data available, 1=use Kp only with front-end models, 2=use Kp only with default models (see Section 3.2.2.2)

needed to understand and setup this file.

Lines 1 and 2 are such that a 12 hour run will be accomplished (49 time steps including the initial one). This is standard. After each run this must be incremented by 12 hours (43200 seconds) to set up for the next contiguous run. This change is one of the automation details accomplished by the *fixfiles.f* program between runs.

Line 3 is set to 0 for the initial run (the cold start). The *fixfiles.f* program changes this to 1 (and maintains it at 1) for all subsequent runs (hot restarts). This tells the model to read record number 1 of the *flux* file as the particle distribution starting point for the next run. The next paragraph explains how this works.

First, after completion of a 12 hour run, the *flxintrp.f* program reads in the values at all grid points at the all 49 time steps in the *flux* file. It also reads in the five other necessary binary output files discussed in Table 11. Using this information, *flxintrp* is able to determine the spatially interpolated flux values along the geosynchronous orbit path for the 12 hour period. These results are written to the *modflux* files. The information in the output files is no longer needed, except for the values of the last time step in the *flux* file, which are necessary for beginning conditions for the next run. Therefore, the *fixfiles.f* program reads the last record of the *flux* file and saves it to a file called *tempflux*. Then the *runmsm* program erases all the output files, including the *flux* file.

Finally, the *runmsm* program renames *tempflux* to *flux*. The *flux* file now has only one record in it. On the hot restart, line 3 of the *msmin* file instructs the model to use this record as the particle distribution starting point file for that run.

Line 4 can be any character string the user chooses for identification of file headers, up

to 80 characters in length. This remains static throughout a series of contiguous runs.

Line 5 is the daily sunspot number (SSN). The issue of sunspot number as an input requires some explanation. Note that SSN was not listed as an input file in the section on inputs earlier in this report. This is because it is provided to the model in the *msmin* file rather than via the *indata* subroutine like all other input. Two methods exist for handling SSN. The first method is simply to take the arithmetic mean of the daily SSNs for the period of interest and enter it here on line 5 as a static value. This was the standard practice of the Rice developers (5). Alternately, a file of daily sunspot numbers for the period of interest can be created and placed in the directory with the executable code. The user can then modify the *fixfiles* program to update line 5 with the passage of each 24 hour period in the input data set. This was the approach adopted in this study. It allows for flexible choice of data periods.

SSN is a required input. It is available in archive form from the Space Environmental Services Center, Boulder, Colorado. However, it should be stated that it is used by the model only in the *hiepar* routine. This routine gives a crude estimate of high energy particle fluxes (greater than 3 MeV). The methodology of this routine is entirely different from that used to determine particle fluxes in the 1-100 keV energy range. It is a simple, empirical method rather than a physically based method (4:2-2). This study did not look at *hiepar* outputs.

Line 6 of the *msmin* file simply assists the user in bookkeeping. All output files produced during a model run are actually named as follows: the file name, preceded by the prefix letter found on line 6. Thus, the model wouldn't produce a file named *flux,* but rather a file named *aflux*, given the case of the example *msmin* file above. This becomes an issue when multiple independent runs are performed.

Line 7 is the toggle for instructing the model whether or not to exercise an option to have all output printed to the screen (or routed to a print file) in ASCII format as the model executes. This produces one long ASCII file as well as all the individual files named in Table 9. Since this option slows down the model run and requires additional memory, the no print option was the default in this study.

Issues regarding line 8 will be discussed later in the chapter. At this juncture, it is sufficient to understand that a value of 1.00 or 1.34 is needed on for the model to execute.

Line 9 provides the toggle for instructing the model on how to handle the available environmental input data. The issues involved here were fully explained in Section 3.2.2.2.

4.2.2 Setup of the Operator Specified File *enchan*

The other user specified input file besides *msmin* is the *enchan* file. The mechanics of setting up the *enchan* file are quite simple, although certain issues regarding choice of entries into the file are more complex. The mechanics will be described here, the issues will be dealt with later in the chapter.

Table 16 below gives an example of the content of an *enchan* file, and describes each line. The narrative in this section provides the detailed information needed to understand and setup the file.

The model user determines the size of the *enchan* file. Except for Line 1, each line entered in the *enchan* file contains two numbers. The first is an integer and the second is a floating point number. The integer number defines the chemical species the user desires for the model to simulate. The number 1 is used for electrons, the number 2 for protons, and the number 3 for oxygen ions. The floating point number in each entry represents the

TABLE 16
SAMPLE *enchan* FILE

LINE #	SAMPLE CONTENT	DESCRIPTION
1	6	- the first line tells the model how many 'energy channels' to simulate ('energy channel' is defined in the narrative below)
2	1 37000.	- subsequent lines define particle species and energies (in eV)
3	1 54000.	
4	1 80000.	
5	2 56500.	
6	2 71500.	
7	2 90000.	

particle energy at which the user desires for the model to calculate differential flux outputs. The combination of particles species and energy is called an 'energy channel' (6:56-57). A maximum of 37 energy channels are allowed. Electron entries precede proton entries, and proton entries precede oxygen ion entries. Within a chemical species, energies must be in ascending order.

Line 1 of the *enchan* file simply records the total number of energy channels the user desires for the model to simulate. Declaring this number allows the model to appropriately dimension its arrays.

4.2.3 Setup of the Magnetic Field Files

Another set of files which must be accessible to the executable code is the set containing the lookup table of magnetic field configurations. As mentioned in the previous chapter, there are nearly 1000 of these files, requiring over 350 Mb of storage space. The

easiest means of handling these files is to place them together in their own subdirectory, below the home directory of the MSM executable code.

The MSM code accesses the magnetic field files by OPEN statements in two different subroutines, *loadbm* and *mexist*. The FORMAT statement for these is the same in each subroutine, and includes the pathname to the B-field files. The user must make the pathname here specific to the file organization he establishes on his machine.

Additionally, the CHARACTER declaration at the beginning of each subroutine must have the correct length declared for a file called *filnam* in the subroutines. This will account for all the keystrokes in the user specific pathname mentioned above.

4.2.4 Executing Hot Restarts

It would be useful at this point to review Tables 11 and 14 in Chapter 3. Table 11 introduced a piece of post-processing software called *flxintrp.f*, and Table 14 introduced two pieces of utility software called *fixfiles.f* and *runmsm*. All of these need to be present in the MSM home directory.

Two methods are possible for hot restarts. The first is the manual method. After the 12 hour run from a cold start is completed and all the output files have been created, *flxintrp* must be executed. The *modflux* files created by this step must now be moved to another directory for storage so that they will not be overwritten the next time this step is taken. A suffix should be added to the names of these files in that directory in such a way that results from each run can be distinguished. If anything of significance is in the *errfile* for the run, that file can also be stored.

The *fixfiles* software must now be executed. When this is complete, all output files

need to be deleted, since they require 15 Mb of storage per run. The *tempflux* file created by executing *fixfiles* must now be renamed *xflux*, where 'x' is the run prefix from line 6 of the *msmin* file. The next 12 hour run is now ready for execution using the *msmwork* command.

The second method for accomplishing a hot restart is to execute the *runmsm* software from the beginning of a study. All of the steps enumerated in the manual method above are accomplished by *runmsm*. This allows for a "walk away" model run. The *runmsm* software will execute a series of ten 12 hour runs, as long as the input data files contain data to cover the period. A file called *nohup.out* is also created using this method. This file records the start and stop times of each run in the series, as well as recording the contents of the *errfile* for each run. If a person is familiar with the C programming language, *runmsm* can be modified to allow for a different number of consecutive runs.

4.2.5 Comparison Run

In order to help test the efficacy of the model testing platform at AFIT, B. Hausman at Rice provided a "test package." It consisted of *msmin* and *enchan* files and three hours of environmental input data from August 1990. It also included the binary output files generated on their platform at Rice, using the same Version 5.0 of the MSM code. When run at AFIT, the flux results agreed with the Rice results to the fifth decimal place. This test validated the overall performance of the code in the AFIT setting.

Now that the technical information needed to set up and execute a model run has been presented, it is possible to move on to the central theme of this chapter. As mentioned already, the order of presentation will be issues encountered and solutions formulated regarding the following: input data, MSM code, model output, satellite comparison data, and

post-processing software.

4.3 Input Data Issues

4.3.1 Operator Specified Input

The operator specified input files, *msmin* and *enchan*, have already been discussed at the conceptual level and in the context of setting up for model execution. However, several issues remain to be dealt with.

4.3.1.1 *msmin* File Issues

The MSM code, in its delivery form from Rice, is written such that the data contained in the *msmin* file must be typed in through the keyboard. In the *msmcon* routine, the READ statements for operator specified input are directed to logical unit 5 (keyboard default). Placing this data in a file allows for automation of the run process. Therefore, the logical unit was changed from 5 to 2 in *msmcon* to allow use of the *msmin* file.

4.3.1.2 *enchan* File Issues

Section 4.2.2 explained how to specify in the *enchan* file the energy channels desired for model differential flux output. When doing an accuracy study, specific energy channels must be chosen. They must correspond to particle species and energies available in a satellite data set. Otherwise, no comparisons are possible.

To understand how satellite data is made compatible with the model's differential flux output, the issue of differential channel correction of satellite data must be explained. This will be done in Section 4.6.2.3. At this juncture, it is sufficient to state that the energy channel values placed in the *enchan* file are selected to be the center values of the narrowed energy bands of the geosynchronous satellite data.

4.3.2 Environmental Input

4.3.2.1 Data Quality Issues

The output of any model can only be as good as the inputs provided. Acquiring raw environmental data sets and properly constructing model input files from them is a non-trivial task in the research effort. Great attention to detail is essential at this juncture. The remainder of this section will explore the issues which must be accounted for when setting up the environmental data input files. These issues were uncovered one by one in the process of validating the data set used in this study. For some issues, no written guidance was available. In these cases, present and former Rice scientists and students provided valuable counsel.

There are two reasons for including these details in this report. First, proper resolution of these issues is essential to being able to demonstrate the validity of the accuracy testing done. As mentioned in Chapter 2, difficulties existed in ETAC's original study, since the accuracy of their draft report was questionable due to an inadvertent mishandling of input data. The second reason for presenting these issues is that in any future MSM research, a new time frame suggested for analysis will require the construction of a new input data set, and all these issues will have to be revisited.

4.3.2.2 Formatting Issues

All the environmental data files are numeric data accessed by MSM through the subroutine *indata*. The READ statement in *indata* is an unformatted read. Thus, there are only three simple guides on formatting. The first point is that the files must contain seven columns in each row, each row corresponding to a given time tag. Secondly, the first three columns must correspond to 1) parameter value, 2) year, and 3) Julian decimal day. Finally,

the last four columns in each row are place holders, given values of -999. The number of spaces between columns, and whether the numbers are floating point or integer are not critical in unformatted reads.

4.3.2.3 *fkp* File Issues

The Göttingen Kp data, as archived by the National Geophysics Data Center (NGDC), comes as eight, three hour values per day. The first value applies from 0000-0300 Z, the second from 0300-0600 Z, etc. The NGDC files list the standard "+, o, -" descriptors in a decimal format. For example, a Kp index of "4-" is listed as 3.7, "4o" is 4.0, and "4+" is 4.3.

Certain issues must be taken into account with Kp data. These relate to the way the Kp values must be rendered in the input file. First, it must be highlighted that for Kp (and for most input parameters), the time increment between data points is much longer than the model's 15 minute time step. Thus, the model must interpolate between two data values to establish input values for the intervening time steps.

If Kp values were simply placed in the input file at each three hour mark, then the model would interpolate between the three hour values and produce a continuously sloping Kp input. Kp is intended to be constant for a three hour period in the model, since it is a three hour index (5). To model a constant value for a three hour period, the same value would have to be designated at both ends of the period. However, a difficulty would be encountered with the change-over points at the three hour marks. Here the model would see a step function with an infinite slope, and it could not handle this mathematically.

As a consequence, the Rice developers devised a thirty minute 'ramp' concept around the three hour marks (5). To understand this, suppose the NGDC Kp data file showed the

following values for Julian day 244 of 1989: 2.3 for 0000-0300 Z, 2.0 for 0300-0600 Z, and 0.7 for 0600-0900 Z. In the *fkp* file, the values would be rendered as shown in Figure 2. Figure 3 then shows the idea graphically.

```
2.3  1989.0  244.0104  -999  -999  -999  -999
2.3  1989.0  244.1146  -999  -999  -999  -999
2.0  1989.0  244.1354  -999  -999  -999  -999
2.0  1989.0  244.2396  -999  -999  -999  -999
0.7  1989.0  244.2604  -999  -999  -999  -999
0.7  1989.0  244.3646  -999  -999  -999  -999
```

Figure 2. Sample *fkp* File

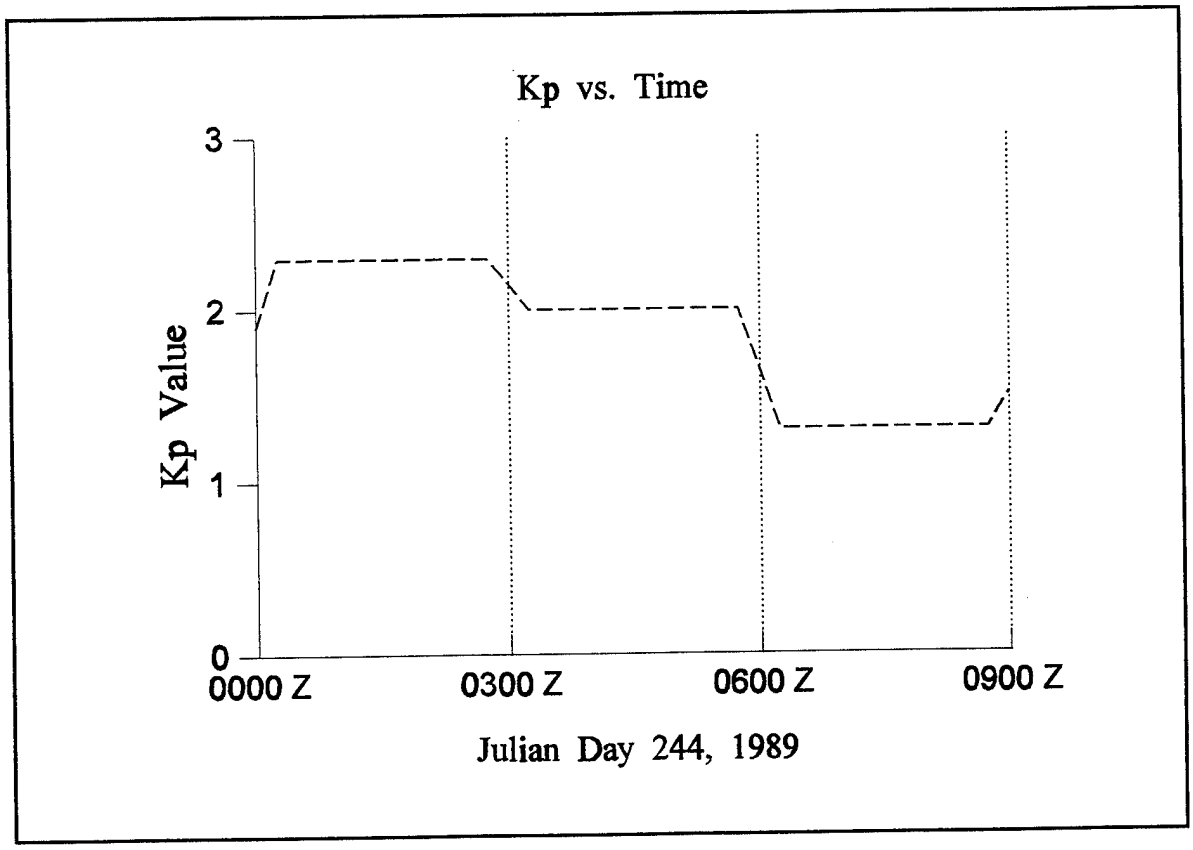


Figure 3. Sample *fkp* File Data Plot

The Kp values are assigned at 15 minutes after a three hour mark, and 15 minutes before the next three hour mark. In Figure 2, if we translate from decimal day format (multiply the fractional part by 1440 to get the number of minutes), we see that the 0000-0300 Z value of 2.3 is time tagged at the 0015 Z mark and again at the 0245 Z mark. Then, the 0300-0600 Z value of 2.0 is time tagged at the 0315 Z mark and again at the 0545 Z mark, etc. Thus, the model interprets the Kp value as flat for two and one half hours. Then, for one half hour across each three hour mark, there is a ramp, or slope, to the next Kp value.

The *fkp* file originally acquired from ETAC's study was discarded as faulty in this study due to various formatting issues of the types described above. ETAC subsequently provided an updated file for use in the AFIT study. It was important to independently verify that it was correct. Therefore, the data in the file were manually compared against original data acquired from the NGDC database for the period. Both the Kp values, and the way in which they were rendered in the data file were correct.

4.3.2.4 *dst* File Issues

The NGDC data files present Dst values as 24 numbers per day. The first number in the data file applies to the first hour of the day, 0000-0100 Z, the second number applies to 0100-0200 Z, etc. The Rice developers determined that the best way to render this in an input file was to give the data a half hour time lag, that is, assign the index value for a given hour to the half hour time tag of that hour (5). Thus, the 0000-0100 Z value gets assigned to 0030 Z, etc.

This brings up a potential stumbling block which presents itself when translating data from archived sources into input files. This problem is present any time the values in the

archived source are an index (over a given number of hours), or are averages of measurements over a certain time period, as opposed to being an environmental reading at a discrete point in time. Two issues must be carefully taken into account in these cases.

First, with respect to the archived data source, an hourly parameter such as Dst may sometimes be time tagged at hours 0 through 23 of the day, or perhaps in a different source at hours 1 through 24 of the day. It must be determined if a time tag of '1' indicates that the value applies from 0000-0100 Z, or from 0100-0200 Z, etc. Is it a 'before' or an 'after' time tag? For Kp, would a value time tagged in the archived data at 0300 Z apply from 0000-0300 Z, or from 0300-0600 Z? The documentation provided with the archived data should provide the answer, and must be carefully followed.

Secondly, when translating the values to an input file, consideration must be given to how the model reads the data. An example of this is how Kp was 'ramped'. In the case of Dst, an on the hour value was assigned to the next half hour time tag (assuming a 'before' time tag in the archived data). This means that for Dst, the model continuously interpolates between values at hour centers.

As in the case of the Kp data, it was important to independently establish that the *dst* file acquired for this study was sound. The data in the file were manually compared against original data acquired from the NGDC database. Both the Dst values, and the way in which they were rendered in the *dst* file were correct.

4.3.2.5 *eqedge* File Issues

This data is a derived data set, taken from raw DMSP SSJ/4 sensor readings, then processed at Phillips Lab (PL). It is usually approximately an hourly value, often having a

more frequent interval. No adjustment is needed to the time tags of the data values when they are rendered in the input files (5).

In this study, *eqedge* data was assumed to be valid and accurate as received from ETAC, since it originated at PL. No attempt was made to assimilate or question the algorithm applied to the raw data, or to pursue an understanding of the satellite sensor.

This file, *eqedge*, and the next two discussed, *pcp* and *xipatt*, are all derived from raw satellite data and can have a fairly high data density (number of values per time unit). When setting up inputs for a simulation period of one month or more, the number of entries in these files can easily exceed the model's upper limit of 1500 values for data arrays. Since this study used a data set covering all of September and October of 1989, these three files had to be split into halves covering one month. In a long contiguous run, when the execution reaches the crossover between months, some filename changes must be made to keep a continuous data supply in place.

4.3.2.6 *pcp* and *xipatt* File Issues

The data in these files are derived from DMSP SSIES sensor readings, processed at the University of Texas at Dallas (UTD). As in the case of *eqedge* data, *pcp* and *xipatt* data are usually approximately hourly values, sometimes more frequent. No adjustment is needed to the time tags of the data values when they are placed in input files (5).

The *pcp* and *xipatt* files originally acquired through ETAC were not used in this study. Instead, new data were acquired directly from UTD and formatted into input files. The reasons for this are explained in the next few paragraphs. The new data were assumed to be valid and accurate as received from UTD. No attempt was made to assimilate or question

the algorithms applied to the raw data, or to pursue an understanding of the satellite sensor.

It was desired that the newest version of all items be used in this study. Since the time of ETAC's study, the algorithm used at UTD to produce this data has been updated. Therefore, new *pcp* and *xipatt* data were acquired.

Comments in the MSM source code give instructions that if new *pcp* data is used in a model run, Line 8 of the *msmin* file must contain the value 1.00 (4:msmcon-1). It must contain the value 1.34 if *pcp* data derived by the old algorithm is used. (See also Table 15 of this report.)

Finally, some readers may be aware of an issue regarding time tagging of data from DMSP with respect to equatorial crossing times of the satellite (10). It is sufficient to report here that the updated algorithm used to create the new files for this study treated this timing issue in the same way as the old algorithm (7). Thus, the new data properly matches the built-in timing correction in MSM.

4.3.2.7 *swden & swvel* File Issues

The solar wind density and velocity data obtained through ETAC were used 'as is' for the model runs in this study. The ETAC draft report indicates that they obtained the data from the National Space Science Data Center (NSSDC) (2:6). As such, the data itself was assumed to be reliable.

The data are one hour averages given at one hour intervals. As to temporal format, when they were rendered in the input files by ETAC, no half hour time delay was used as it was in the case of the *dst* files. Instead, the data were time tagged at the beginning of the hour to which they applied as an average. This raised questions. Therefore, the files were

compared to test file set acquired from Rice, which they had formatted. With respect to this time formatting issue, the two sets matched identically. This indicates the time tagging of the solar wind files in this study is correct.

Another question surfaced about these files after the completion of all model runs. It was discovered that the values in the files do not precisely match a current NSSDC data set for the same satellite (IMP-8), same parameters (solar wind density and velocity), same time period (1989, Julian days 244-304). Density values differ by an average of roughly five percent, and velocity values by two percent.

The differences do not appear to be caused by a mishandling of the data files by ETAC. Mishandling would have caused a systematic discrepancy such as a time shift due to a missing data value, and this would have been obvious. The discrepancies between values in the two files are tiny, random differences at nearly every time step. Encouragingly, the two data sets show identical dynamical patterns over the long run.

The NSSDC data files may have simply been updated since the time ETAC acquired the data from them several years ago. It seems logical that the current data sets may be the product of new NSSDC algorithms used to determine solar wind density and velocity.

In the final analysis, the differences between the data sets are small enough that the effects on MSM outputs are assumed to be minor. However, no unambiguous answer has yet been obtained to the above issue. Therefore, consideration was given to excluding from this report any model output results which included as inputs the solar wind data. However, it seems to be a sensible assumption that the solar wind inputs are reliable, though slightly dated, and results derived using them as inputs provide useful information about the performance of

the model. Therefore, the decision was made to retain these results in this report. In the analyses performed in this study, many of the model runs were performed without solar wind data as inputs. Those which include it will be easily identified.

4.4 MSM Code Issues

4.4.1 Compiler Issues

Two compiler issues surfaced in this research effort. The first was previously reported in the ETAC draft report (2:16). In the delivered MSM code, there exists a subroutine named *grid* as well as a COMMON block with the same name. The SPARC 2 SUN Fortran compiler would not compile because of this name duplication. Since there are only two calls to the subroutine, but many occurrences of the COMMON block, the most expedient fix was to change the name of the subroutine from *grid* to *rgrid.*

The second compiler issue had to do with Fortran compiler version. The original Fortran version on the SPARC 2 workstation used in this study was Fortran 1.3. Inexplicable compiler errors occurred using this version. A simple switch to Fortran 1.4 resulted in a complete resolution of the situation, with no change in the MSM source code. It is believed in hindsight that the problem lies in the way in which the older Fortran compiler handles memory allocation (static vs. dynamic). It could be that a 'static' option declared in the compiling step might allow the use of Fortran 1.3 with MSM code on the SPARC 2. However, this was not proven, and it is suggested that Fortran 1.4 or later be used to compile the source code.

4.4.2 Run Time Comparison

MSM is very computationally intensive. Therefore, a comparison of performance speeds between different computer platforms may be useful to a future researcher. The

statistics in the following table are derived from executions of the model on five different UNIX machines, each with the identical one hour's worth of input data. Since it was not possible to get dedicated use of most of the machines, the percent of the dedicated CPU time in each case varies. This table is provided as a qualitative tool only. No claim of determination of absolute performance of these machines is implied. As a rule of thumb, the most significant figures for comparison are the CPU times. The lower the better. System time is related to things like input / output overhead. Elapsed time (wall time) depends strongly on how heavily multi-tasked the CPU was with other users (look at the percent of total CPU time given to the MSM tasking).

TABLE 17
MSM RUN TIME COMPARISON

MACHINE	CPU TIME (sec)	SYSTEM TIME (sec)	ELAPSED TIME (min)	PERCENT OF TOTAL CPU
SPARC 20	282.3	5.5	9:45	49
SGI R3000	308.9	51.3	7:13	83
KUBOTA	518.9	23.9	9:32	94
SPARC 10	613.8	38.8	11:23	95
SPARC 2	684.0	12.3	12:18	94

The SPARC 2 was generally available for dedicated usage. A typical run of 12 hours of data on this platform consistently required 140 minutes of wall time. A five day, hands off run using *runmsm* required roughly 24 hours. By comparison, when a SPARC 20 was free for essentially dedicated usage, a run of 12 hours of data took roughly 60 minutes, and a five day

run required approximately 10 hours.

4.5 Model Output Issues

The next three major sections of this chapter, 4.5, 4.6 and 4.7, are highly interconnected. The issues which arose regarding model output, satellite comparison data and post-processing software are difficult to separate. As a consequence, some topics are presented out of sequence with respect to section headings.

4.5.1 Assumptions Made Regarding Model Output

MSM includes no dipole tilt in its modelling. Differential flux output in the magnetic equatorial plane is assumed to be equivalent to that in the geographic equatorial plane. This built in assumption was accepted in this study.

Two necessary interpolations of output flux values were performed. The first, accomplished by the *flxintrp.f* software, is a spatial interpolation. The second, accomplished by the *compare.f* program, is a temporal interpolation. Both will be explained in detail in later sections. It was assumed that these interpolations are logical methods of bringing the model data to a place where it is comparable to the in situ satellite data. Only then can calculations of the squared error differences between the two data files be performed.

4.5.2 Post-Processing of Model Output

The model's direct output is differential flux values with the following characteristics: 1) binary format; 2) specified only at model grid points; 3) specified only at each 15 minute model time tag; 4) units of [particles/cm^2-sec-ster-eV]; 5) no quality control has been performed on the data. This is not an appropriate form for comparison with satellite readings.

Tables 11, 12 and 13 in Chapter 3 introduced post-processing software. To review,

after a model run, the user must accomplish post-processing of output data. In addition, satellite data must also be manipulated by software. The goal is to perform quality control on each data set, and modify them to the point where the units match, and the data values have temporal and spatial correlation.

This section will explain the steps taken with each piece of post-processing software used on the model output. Table 18 traces the changes made at each step in data manipulation. Particular attention should be paid to energy units.

TABLE 18
DETAILS OF POST-PROCESSING OF MODEL OUTPUT

STEP IN THE DATA MANIPULATION	UNITS	METHOD USED TO CHANGE DATA
MSM binary output files	particles/cm^2-sec-ster-eV	- these are the units of the model output
flxintrp.f - 1st operation	particles/cm^2-sec-ster-eV	- spatial interpolation
flxintrp.f - 2nd operation	particles/cm^2 -sec-ster-keV	- multiply by 1000 to change units
flxintrp.f - 3rd operation	log10 (particles/cm^2-sec-ster-keV)	- take log10
qcmsm.f	log10 (particles/cm^2-sec-ster-keV)	- quality control

The 1st operation of the *flxintrp.f* software is a spatial interpolation of the model's output. At the end of each 12 hour run, *flxintrp.f* reads in the flux values at all grid points at the 49 time steps in the *flux* file. It then reads in five other binary output files discussed in Table 11. Using this information, *flxintrp* is able to determine where the satellite orbit would fall within the model grid at each 15 minute time tag. Generally, these points do *not* coincide

with grid points at which the model has assigned a flux value. Therefore, *flxintrp* performs a linear, spatial interpolation of flux values between four neighboring model grid points. This generates values at the locations in space where the geosynchronous satellite would be in its orbit at each time tag.

The 2nd operation of *flxintrp.f* is nothing more than multiplying each interpolation value by 1000. This simply changes the units from (particles / cm^2 - sec - ster - eV) to (particles / cm^2 - sec - ster - keV).

The 3rd operation of *flxintrp.f* takes the log10 of the differential flux output values. Log10 values are easier to compare and graph.

The *flxintrp.f* software used at AFIT was an ETAC modified version of the corresponding software used at Rice called *calflx.f*. For this study, the format of *flxintrp.f* output was preferable to that of *calflx.f*. However, it was important to demonstrate that the modified software produced identical results compared to the original Rice version. Therefore, both programs were used to process identical MSM binary output files. Agreement was achieved to the fifth place past the decimal.

In Table 12, the *qcmsm.f* step was introduced Chapter 3. It was mentioned that during periods of high geomagnetic activity, the model boundary can shrink to less than geosynchronous orbit distances. When this occurs, the *flxintrp* software recognizes these occurrences and flags those time steps with 9.999 flux values in the *modflux* files. Before using the *compare* software to run accuracy tests against the in situ satellite data, these out of bounds values must be removed from the *modflux* files. This is the sole purpose of the *qcmsm.f* software.

4.6 Satellite Comparison Data Issues

4.6.1 Background Information

In Section 3.5, the subject of understanding satellite comparison data was introduced. To review, the data came from two DoD geosynchronous orbiting spacecraft, at longitudes of 15 and 70 degrees east. The data are five minute particle count averages for electrons and protons at approximately five minute intervals over broad energy bins. The data was originally acquired by ETAC through the Los Alamos National Laboratory and Air Force Global Weather Central. AFIT acquired it from the ETAC database.

At each time tag, the electron data file has particle counts for the energy channels 0.030 to 0.300, 0.044 to 0.300, 0.064 to 0.300 and 0.095 to 0.300 MeV. For protons the channels are 0.050 to 0.500, 0.063 to 0.500, 0.080 to 0.500 and 0.100 to 0.500 MeV. It is more convenient to speak of these in keV units, i.e. 30 to 300 keV, 44 to 300 keV, etc.

4.6.2 Satellite Data Processing

The satellite data must undergo significant modification of its original form to make it compatible with MSM flux outputs. The manipulations of the satellite data are more complex than those applied to the model output. It would be useful to review Table 12 in Chapter 3 where the basic concepts were introduced. This section will present the details of the procedures employed. Table 19 below outlines the changes in the satellite data at each step in its manipulation. After the table, elaboration is provided on each technique used. As in the last section, particular attention should be paid to energy units.

TABLE 19
DETAILS OF SATELLITE COMPARISON DATA PROCESSING

STEP IN THE DATA MANIPULATION	UNITS OF DIFFERENTIAL FLUX	METHOD USED TO CHANGE DATA
original satellite data	particles/cm^2-sec-ster-MeV **broad energy channels (bec)**	- these are the original units
epform.f	particles/cm^2-sec-ster-MeV (bec)	- change data format
epconv.f-1st operation	particles/cm^2-sec-ster-MeV (bec)	- quality control
epconv.f-2nd operation	particles/cm^2-sec-ster-MeV (bec)	- detector deadtime correction
epconv.f-3rd operation	particles/cm^2-sec-ster-keV **narrowed energy channels (nec)**	- differential channel correction
epconv.f-4th operation	log10(particles/cm^2-sec-ster-keV) (nec)	- take log10
epconv.f-5th operation	log10(particles/cm^2-sec-ster-keV) (nec)	- change data to Rice format
epqc.f-1st operation	log10(particles/cm^2-sec-ster-keV) (nec)	- quality control
epqc.f-2nd operation	log10(particles/cm^2-sec-ster-keV) **assigned to central value**	- changes data to *compare.f* format

4.6.2.1 Preliminary Processing

The *epform.f* software simply reads the data from the original Los Alamos file and writes it unaltered to a new file, changing only the format of the page. Normally this step would not be necessary. However, in this case the data in the original file was in an extremely unwieldy two page wide format.

The first operation of *epconv.f* consists of four quality control measures designed by ETAC to test the integrity of the in situ data (2:8). Errors often arise in transmission, or from

other causes, when dealing with raw satellite data. Each observation contains values for several energy channels. It is assumed that if a value in any of the channels is bad for a given five minute interval, all the values at that time step ought to be discarded. The energy channels, after all, essentially overlap.

First, for each five minute interval in a given species, the program checks to make sure that all broader energy channels have a higher flux value than any narrower energy channels. For example, the 30 to 300 keV channel ought to contain more particle counts than the overlapping 44 to 300 keV channel, etc. The program discards all observations which do not meet this test. Second, the program eliminates all values at time steps which contain any negative or zero flux values. Third, the program tests to make sure that adjacent energy channels do not differ by large amounts. The threshold used is the following: any broader energy channel must contain not greater than 100 times the next narrowest energy channel. Fourth, and last, the program tests for any extreme flux values. Observations are discarded at time steps which contain any flux readings greater than or equal to 10^{12} particles/cm^2-sec-ster-MeV. Overall, roughly 4% of the values are discarded by these measures.

4.6.2.2 Detector Deadtime Correction

The second *epconv.f* operation, the Detector Deadtime Correction, is used to add back the number of particles which have been missed by the sensors between instrument cycles. The ETAC draft report references a Quarterly Status Report No. 11, AF Geophysics Laboratory, as the source for this correction equation (2:8). It takes the following form:

$$f' = \frac{f}{1 - f * 3.1 \cdot 10^{-9}} \qquad (1)$$

where f ' is the deadtime corrected flux, and f is the measured flux in (particles / cm^2-second-MeV). It has a very minor effect, accounting for a 1% correction at most (2:7).

4.6.2.3 Differential Channel Correction

The third *epconv.f* operation, the Differential Channel Correction, is the most complex of the satellite data manipulations. Recall that MSM determines differential flux values for the energy channels declared in the *enchan* file. In the satellite data there are only broad energy channels. Since the channels are essentially overlapping measurements of a five minute interval, a narrowed energy channel can be created by subtracting adjacent channels. For example, the flux value for the 0.044-0.300 MeV channel is subtracted from the flux value for the 0.030-0.300 MeV channel to yield the value for the 0.030-0.044 MeV energy channel (equivalently, the 30 to 44 keV channel). Conceptually it is as simple as that. The full explanation is a little more involved.

The ETAC draft report references the Quarterly Status Report No. 11 as the source of the following formula for corrected differential flux (2:8):

$$\frac{df'}{dE} = \frac{[\, f'_b * \Delta E_b \; - \; f'_{b+1} * \Delta E_{b+1} \,] * C}{\Delta E_n \; (keV)} \qquad (2)$$

Here the f'_b and f'_{b+1} values are the deadtime corrected fluxes for adjacent broad energy

channels, in units of (particles / cm^2 - sec ster - MeV). The ΔE_b and ΔE_{b+1} are the broad energy channel bandwidths in terms of (MeV) units, i.e. [0.300 (MeV) - 0.030 (MeV)] = 0.270 (MeV), and [0.300 (MeV) - 0.044 (MeV)] = 0.256 (MeV), etc. (Please note, here the " - " is a minus sign, not the abbreviation for the word " to ".) Thus, each of the terms inside the brackets in the numerator has units of (particles / cm^2 - sec - ster) after the multiplications.

Before discussing the last term in the numerator (a constant), the denominator will be discussed. The term in the denominator, ΔE_n is the narrowed energy channel bandwidth, in units of (keV). For example, [44 (keV) - 30 (keV)] = 14 (keV) , and [64 (keV) - 44 (keV)] = 20 (keV), etc. (Again, the "-" symbols are minus signs.)

Tracking the energy units of numerator and denominator, it is seen that the Corrected Differential Flux, df ' / dE, has units of (particles / cm^2 - sec - ster - keV). These are the same units as the model output after post-processing of those data files.

The differential fluxes found by the method above are essentially average flux across the narrowed energy channel bandwidth. In the Rice methodology reported in the ETAC draft report, it is assumed that this average differential flux is suitably applied at the center of the bandwidth (2:8). For example, the average flux value for the 30-44 keV channel is assigned to the 37 keV particles. In this way, satellite data is finally changed to a form which is comparable to MSM calculated fluxes.

The last term in the numerator of Equation 2 is a channel sensitivity correction factor. H. Garrett at Rice determined that certain of the electron sensors underestimate the true flux values (8). The flux values for these channels must be multiplied by a correction factor. The constant C is 2.0 for the 30-44 keV channel and 1.4 for the 44-64 keV channel. It is 1.0 (i.e.

no correction) for all other channels, including all proton channels.

4.6.2.4 Final Processing

The fourth *epconv.f* operation is self explanatory. The log 10 of each differential flux value is calculated.

The fifth *epconv.f* operation simply writes the results to a file in the correct format for optional input as *epsat* and *epions* files into the MSM model for use in the *opterr* routine. As previously mentioned, this method was not used in this study.

The first *epqc.f* operations is another set of quality control measures. It is employed at this point to remove data spikes. Since all flux values at this point are in log10 form, each change in value of 1.0 represents an order of magnitude change. This amount, one order of magnitude, was selected as the threshold for filtering the time trace of flux values in any single narrow energy channel. This means that if the observations before or after a given time tag differ from that reading by 1.0 or more, all the values for that observation are discarded. The assumption here is that average flux values will not change more than one order of magnitude between two five minute intervals. This measure discards only about 1% of the data.

The second *epqc.f* operation is simply a change of format to make the final display of in situ satellite flux values easy to visually inspect. The *compare.f* software written for this study, which does the squared error difference calculations between model and satellite fluxes, expects the satellite values to be in this format.

Two final points need to be made about the original satellite comparison data files. First, each set of values is tagged with a longitude number, indicating the location of one of

three of the geosynchronous satellites. In the original file, two of the locations are in error. All observations tagged at 205 degrees actually belong to the 70 degree east satellite, and vice versa. This was confirmed by ETAC through personal communication with R. Belian at Los Alamos (3). Second, many more channels are present in the original files than are used in this study. Since the MSM only produces output deemed trustworthy in the 1-100 keV range, the higher energy channels were simply ignored (4:2-2).

4.7 Post-Processing Software Issues

All the various corollary programs used external to MSM have been discussed in sufficient detail in the context of other sections. The exception to this is the *compare.f* software.

This program, developed at AFIT, is used to accomplish a squared error accuracy analysis between the in situ particle flux files and the MSM output flux files. The *compare.f* program accomplish one more interpolation of model output before the actual comparison of data is performed.

Recall that the *fixfile.f* program executed a spatial interpolation on MSM output flux values. The *compare.f* program must now perform a temporal interpolation. This is necessary because in situ satellite readings come at approximately five minute intervals while model outputs come at fifteen minute intervals. In general, the time tags of the two data sets do not coincide. In order to maximize the number of data points compared, the time tags of the in situ data file are used as references, and then a linear interpolation is executed between MSM outputs, producing model flux values with time tags which match those of the in situ flux readings.