

AE9/AP9/SPM V1.0 Runtime Algorithms

December 30, 2013

T. Paul O'Brien
Space Science Applications Laboratory
Physical Sciences Laboratories

Prepared for:

Air Force Research Laboratory
Kirtland AFB NM 87117-5776

Contract No. FA8802-09-C-0001

Authorized by: Engineering and Technology Group

Distribution Statement A: Approved for public release; distribution is unlimited.

PHYSICAL SCIENCES LABORATORIES

The Aerospace Corporation functions as an “architect-engineer” for national security programs, specializing in advanced military space systems. The Corporation's Physical Sciences Laboratories support the effective and timely development and operation of national security systems through scientific research and the application of advanced technology. Vital to the success of the Corporation is the technical staff's wide-ranging expertise and its ability to stay abreast of new technological developments and program support issues associated with rapidly evolving space systems. Contributing capabilities are provided by these individual organizations:

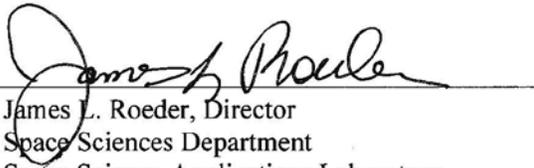
Electronics and Photonics Laboratory: Microelectronics, VLSI reliability, failure analysis, solid-state device physics, compound semiconductors, radiation effects, infrared and CCD detector devices, data storage and display technologies; lasers and electro-optics, solid-state laser design, micro-optics, optical communications, and fiber-optic sensors; atomic frequency standards, applied laser spectroscopy, laser chemistry, atmospheric propagation and beam control, LIDAR/LADAR remote sensing; solar cell and array testing and evaluation, battery electrochemistry, battery testing and evaluation.

Space Materials Laboratory: Evaluation and characterizations of new materials and processing techniques: metals, alloys, ceramics, polymers, thin films, and composites; development of advanced deposition processes; nondestructive evaluation, component failure analysis and reliability; structural mechanics, fracture mechanics, and stress corrosion; analysis and evaluation of materials at cryogenic and elevated temperatures; launch vehicle fluid mechanics, heat transfer and flight dynamics; aerothermodynamics; chemical and electric propulsion; environmental chemistry; combustion processes; space environment effects on materials, hardening and vulnerability assessment; contamination, thermal and structural control; lubrication and surface phenomena. Microelectromechanical systems (MEMS) for space applications; laser micromachining; laser-surface physical and chemical interactions; micropropulsion; micro- and nanosatellite mission analysis; intelligent microinstruments for monitoring space and launch system environments.

Space Science Applications Laboratory: Magnetospheric, auroral and cosmic-ray physics, wave-particle interactions, magnetospheric plasma waves; atmospheric and ionospheric physics, density and composition of the upper atmosphere, remote sensing using atmospheric radiation; solar physics, infrared astronomy, infrared signature analysis; infrared surveillance, imaging and remote sensing; multispectral and hyperspectral sensor development; data analysis and algorithm development; applications of multispectral and hyperspectral imagery to defense, civil space, commercial, and environmental missions; effects of solar activity, magnetic storms and nuclear explosions on the Earth's atmosphere, ionosphere and magnetosphere; effects of electromagnetic and particulate radiations on space systems; space instrumentation, design, fabrication and test; environmental chemistry, trace detection; atmospheric chemical reactions, atmospheric optics, light scattering, state-specific chemical reactions, and radiative signatures of missile plumes.

AE9/AP9/SPM V1.0 Runtime Algorithms

Approved by:

A handwritten signature in black ink, appearing to read "James L. Roeder", is written over a horizontal line.

James L. Roeder, Director
Space Sciences Department
Space Science Applications Laboratory
Physical Sciences Laboratories

© The Aerospace Corporation, 2013.

All trademarks, service marks, and trade names are the property of their respective owners.

Abstract

This report describes the algorithms employed by the AE9/AP9/SPM V1.0 runtime library to compute radiation and plasma environments. The algorithms can be used to generate mean, percentile, perturbed mean, and dynamic scenario environments. The algorithms populate the desired global environment on the model grid, evolve it forward in time as needed, and project it onto the times, locations, energy, and angular channels requested by the user.

Acknowledgments

The author acknowledges significant help from his colleagues on the AE9/AP9 team and many useful conversations with personnel at The Aerospace Corporation and the ONERA/DESP group.

Contents

1.	Introduction.....	1
2.	Runtime Tables	3
3.	Magnetic Coordinates Grid.....	4
4.	Populating a Requested Global State	6
4.1	Mean.....	7
4.2	Percentiles	7
4.3	Perturbed Mean	7
4.4	Monte Carlo Scenarios	8
5.	Projecting the Environment onto the Spacecraft.....	10
5.1	Fast Coordinates.....	10
5.2	Interpolation onto Target Magnetic Coordinates.....	10
5.3	Energy Integrals	11
5.4	Angle Integrals	11
5.5	Combined Interpolation and Integration	11
5.6	Temporal Interpolation (dynamic scenarios only).....	12
5.7	Plasma-Radiation Stitching in Energy.....	12
6.	Summary	13
	References.....	14

Tables

1.	Main Quantities in Each Model-Specific Runtime Tables Data File.....	3
2.	Grids for AE9/AP9/SPM V1.0 Models	4
3.	Monte Carlo Parameters for AE9 and AP9.....	9

1. Introduction

This introduction provides a high-level description of the AE9/AP9/SPM runtime algorithms (AE9 is energetic electrons, AP9 is energetic protons, and SPM is a set of models for different species of plasma, electrons, protons/hydrogen, oxygen, and helium). These algorithms are an evolution of those used in prior alpha and beta versions of AE9/AP9 [see, e.g., *O'Brien, 2005; O'Brien, 2007; O'Brien and Guild, 2010*] and have been summarized in *Ginet et al. [2013]*. Later sections will repeat this information in more detail with equations and mathematical notations. The runtime algorithms generate mean, percentile, and static perturbed mean radiation and plasma environments, as well as dynamic Monte Carlo scenarios of the radiation environment. In the case of the mean and percentile environments, the parameters of the statistical distribution at each grid point are given by unperturbed parameter maps. At each grid point, the parameter map provides a two-element vector that can be converted to the median and 95th percentile of the local statistical distribution. The median and 95th percentile can then be converted to the parameters of either a Weibull or log-normal statistical distribution. With those parameters, the mean or any percentile can be computed. The AE9 electron model uses a Weibull distribution, while the AP9 proton model and the plasma models use log-normal distributions. In the case of the perturbed mean static environments and the dynamic Monte Carlo environments, the unperturbed parameter map is perturbed using an anomaly matrix to obtain a different, perturbed parameter map for each scenario. The perturbations are uniquely identified by a random number seed, which is the scenario identifier (ID) number 1 through 999. The anomaly matrix represents the uncertainty in the parameters of the statistical distributions, including sensor response uncertainties, limited duration of sampling missions, counting statistics, and spatial extrapolation/interpolation.

For the dynamic Monte Carlo environments, an initial random state is generated and evolved forward in time for many iterations to ensure that proper temporal correlations are established before the mission simulation begins. Time evolution is achieved by a multi-lag, multivariate auto-regressive process, whose time evolution matrices (persistence and innovation conditioning matrices) are derived from observed spatiotemporal correlations, and the innovations themselves are again keyed to a random number seed corresponding to the scenario ID. The state vector itself is a reduced ~10-element vector of principal component amplitudes. The relationship between the principal component amplitudes and the flux level at any given grid point is provided by principal component matrices that are square-root matrices of a specialized spatial covariance matrix. In fact, the spatial covariance matrix is computed not from the fluxes, but from the fluxes converted to standard Gaussian variables. All of the multivariate linear time iteration operations retain the Gaussian nature of the data, and also preserve the zero mean and unit variance of a standard Gaussian. Thus, one must convert from the standard Gaussian to the local Weibull or log-normal statistical distribution to obtain flux at each grid point.

With flux given at each grid point, one must then compute the requested particle fluxes at the spacecraft location. For unidirectional differential fluxes, the computation is simply linear interpolation from the grid onto the local magnetic coordinates. However, for more commonly requested omnidirectional and/or integral fluxes, the computation may also involve integrating over the local magnetic

coordinates. Angular integrals to obtain omnidirectional flux are conducted in terms of local pitch angle, assuming symmetry around the local magnetic field direction.

The particle flux in each desired energy channel is computed at each point along the spacecraft trajectory, thus providing a simulation of what one would measure with an idealized particle sensor. For the static environments, and often for the dynamic environments, time variation is due mainly to motion of the spacecraft and to a lesser extent to motion of the model currents inside and outside the Earth. However, over time, the dynamic environments will explore a larger range of variation than what appears in the static environment due simply to spacecraft motion.

The simulated sensor data can be fed into an effects code to compute the outcome of radiation or plasma interactions with matter, parts, or systems. For effects due only to whole-mission linear accumulation (average, fluence, dose), the mean and perturbed mean static environments can be used, and can be run for only a representative time sample (either a few orbits, or a set of times randomly but uniformly distributed over the course of the mission). For effects that depend on the instantaneous particle flux (single-event effects, dose rate) or time history of particle flux (internal charging), the dynamic Monte Carlo scenarios must be used. Running full mission simulations can be so time consuming and can produce so much raw flux data that they can be a burden even to mid-range computing clusters and mass storage.

To obtain statistical confidence intervals, one must run multiple scenarios of either the static perturbed means or the dynamic Monte Carlo scenarios, compute the required effects from the fluxes, and then compute statistical distributions of those effects. Computing effects from percentiles of fluxes will likely lead to incorrect results because the calculation of percentiles is a nonlinear operation (sorting), even if the effects themselves are linearly dependent on the flux.

In the remainder of this document, we will describe the algorithms and equations in detail.

2. Runtime Tables

The AE9/AP9 model depends on several databases of runtime tables. Each model AE9, AP9, SPMH, SPME, SPMO, and SPMHE has a model-specific database with a name like AE9V10_runtime_tables.mat. These are binary Matlab save sets and conform to the HDF5 format standard (Matlab is a trademark of The Mathworks, Inc.). There are also support databases, e.g., the coefficients of the International Geophysical Reference Field (IGRF) model [Finlay *et al.*, 2010]. The contents of a model-specific database are identified in Table 1. Each quantity in the table will be used in the later sections of this document. Another report [see O'Brien, 2013] describes the statistical manipulations required to generate each quantity.

Table 1. Main Quantities in Each Model-Specific Runtime Tables Data File

Quantity	Variable Name	Symbol	Size	Purpose
Parameter map	theta	θ	$\sim 20,000^{\dagger} \times 2$	Represent the transformed 50 th and 95 th percentile flux on the coordinate grid
Anomaly matrix	Stheta	S_{θ}	$\sim 40,000^{\dagger} \times \sim 10$	Represents error covariance matrix for θ due to measurement errors
Principal Component Matrix [‡]	Q	Q	$\sim 20,000^{\dagger} \times \sim 10$	Represents the principal components of spatial variation
Time Step [‡]	dt	δt	Scalar	
Persistence Matrix [‡]	G	G	$\sim 10 \times 10 \times \sim 5$	Represents persistence of principal component amplitudes
Innovation Conditioning Matrix [‡]	C	C	$\sim 10 \times 10$	Allocates white noise driver to principal components
Conditioning Time [‡]	conditioning_time	N/A	Scalar	Specifies length of conditioning time needed to initialize state history
Grid	grid	N/A	(structured)	Stores information about the grid

[†]20,000 includes two grids and is appropriate for AE9 and AP9. SPM models are smaller

[‡]40,000 includes two entries for each grid point

[‡]Monte Carlo quantities are not defined for the plasma (SPM) models.

3. Magnetic Coordinates Grid

The *AE9/AP9 Tech Doc* provides a detailed discussion of the drift and bounce invariant coordinate systems used in the component models of AE9/AP9/SPM. Here, we will just provide a short review of the coordinate systems and then describe how they are used at runtime. The specific coordinate identities, ranges, and resolutions are given in Table 2.

The radiation belt models, AE9 and AP9, actually use two grids. The high-altitude grid uses energy (E), Kauffman’s K , and the third adiabatic invariant or flux invariant Φ . The low-altitude grid uses E , K , and the minimum altitude encountered on a drift orbit (h_{min}). It should be noted that K and h_{min} are drift invariants but not adiabatic invariants (i.e., they are constant only in a static magnetic field).

The plasma models (SPM) use E , equatorial pitch angle (α_{eq}), and McIlwain L (L_m) coordinates. Equatorial pitch angle is a bounce invariant, not a drift invariant, but was selected for historical reasons. It should be noted that the plasma particles respond to the global electric field strongly enough that their magnetic coordinates are not drift invariant anyway (a set of electromagnetic coordinates would be required).

The *grid* variable in the model-specific runtime tables provides all the information from Table 2, including identity of the coordinates, their units, ranges and resolutions, and any transforms needed (such as exponentiation or logarithms).

For AE9 and AP9, the runtime tables combine the parameter maps (θ) and the principal components matrix Q for the high- and low-altitude grids by “stacking” them, or concatenating them along the first dimension. For S_0 a second stacking is done with respect to θ_1 and θ_2 . The runtime code keeps track of this stacking with help from parameters stored in the *grid* variable.

Because (as we will see later) the model formulation can represent only non-zero fluxes, zero fluxes are achieved by removing points from the model grid. This is done in two stages: first, an atmospheric

Table 2. Grids for AE9/AP9/SPM V1.0 Models

Model	Species	Energy	High Altitude Grid		Low Altitude Grid	
			2 nd Inv.	3 rd Inv.	2 nd Inv.	3 rd Inv.
AE9	e-	0.04–10 MeV 21 channels	$0 \leq K^{1/2} \leq 4.5$ $\Delta(K^{1/2}) = 0.1$	$0.8 \leq \log_{10}\Phi \leq 0.3$ $\Delta(\log_{10}\Phi) = 0.025,$ $\Phi \sim G R_E^2$	$0 \leq K^{1/2} \leq 4.5$ $\Delta(K^{1/2}) = 0.1$	$-500 < h_{min} < 1000$ km $\Delta h_{min} = 50$ km
AP9	H+	0.1–400 MeV 22 channels	$K \sim G^{1/2} R_E$		$K \sim G^{1/2} R_E$	$0 < h_{min} < 1000$ km $\Delta h_{min} = 50$ km
SPME	e-	1–40 keV 16 channels	$5 \leq \alpha_{eq} \leq 85^\circ$ $\Delta\alpha_{eq} = 10^\circ$	$2 \leq L_m \leq 10$ $\Delta L_m = 0.5$	N/A	
SPMH	H+	1.15–164 keV				
SPMHE	He+/++	12 channels				
SPMO	O+/++					

loss cone is defined and used to remove points based on a relationship between the 2nd and 3rd coordinates. This loss cone filter is independent of energy and flux intensity. The second filter is determined during the statistical processing of the data [see *O'Brien, 2013*], and it removes points where the median flux is smaller than the maximum flux by a specified ratio. For AP9, this ratio is 10^{-10} ; for the other models, the ratio is 10^{-9} . Thus, the second filter is energy and intensity dependent. The lists of which grid points are involved in each filter are provided as part of the *grid* variable in the model-specific runtime tables.

Finally, the *grid* variable includes a set of “linear basis functions” that represent the lower, middle, and upper end points. These basis functions are tracked separately for Energy as opposed to the 2-D spatial region covered by the 2nd and 3rd invariants. The lower and upper bounds are defined to account for different kinds of grid boundaries when performing grid interpolations: no flux is allowed below $K = 0$, below the lowest energy grid point, or in the loss cone. At runtime, there is then no need to consult the definitions of the boundaries, instead relying on the lower and upper bounds for each grid point.

Because the *grid* variable is simply a saved version of the Matlab object used to define the grid for all the backend processing, it contains many vestigial entries that are not used at runtime. In a later version of the model, we may remove the items from the file to reduce potential for confusion.

4. Populating a Requested Global State

For each model, the parameter maps (θ) provide the median (m_{50}) and 95th percentile (m_{95}) flux at each grid point. At each grid point, the map provides two values θ_1 and θ_2 such that:

$$m_{50} = e^{\theta_1} \quad (1)$$

$$m_{95} = e^{\theta_1} + e^{\theta_2} \quad (2)$$

This transform ensures that any pair of real values for θ_1 and θ_2 will give $m_{95} > m_{50} > 0$. With the two percentiles in hand, one can derive the standard parameters of either the Weibull (used for AE9 electrons) or Log Normal distribution (used for everything else), and thereby compute any desired statistical property (usually the mean or a chosen percentile) of the flux at that grid point.

The cumulative distribution function for the Weibull is given by

$$F(x) = 1 - \exp [-(x/x_0)^\gamma], \quad (3)$$

and the parameters are related to the 50th and 95th percentiles by

$$\gamma = \left[\ln \frac{\ln 20}{\ln 2} \right] / \left[\ln \frac{\ln m_{95}}{\ln m_{50}} \right] \quad (4)$$

$$x_0 = \frac{m_{50}}{(\ln 2)^{1/\gamma}}. \quad (5)$$

The cumulative distribution function for the Log-Normal is given by

$$F(x) = \Phi \left(\frac{\ln x - \mu}{\sigma} \right), \quad (6)$$

where Φ is the cumulative distribution function for the unit normal or standard Gaussian. The parameters of the Log Normal are related to the 50th and 95th percentiles by:

$$\mu = \ln m_{50} \quad (7)$$

$$\sigma = \frac{\ln m_{95} - \mu}{\Phi^{-1}(0.95)}. \quad (8)$$

4.1 Mean

To populate the global state for a run through the mean environment, the θ parameters are used to compute m_{50} and m_{95} at each grid point according to the formulae above, and then the following formulae are applied to compute the mean at the grid point:

$$\langle x \rangle = x_0 \Gamma(1 + 1/\gamma) \text{ (Weibull)}, \quad (9)$$

$$\langle x \rangle = \exp[\mu + \sigma^2/2] \text{ (Log Normal)}, \quad (10)$$

where $\langle \rangle$ represents the population mean or average, and Γ is the complete Gamma function.

4.2 Percentiles

When a given percentile flux map is requested, it is rescaled onto the $[0,1]$ domain by the variable u . At each grid point, θ is converted to the parameters of the relevant cumulative distribution, and the flux is then given by the inverse of the cumulative distribution function F given above

$$x = F^{-1}(u). \quad (11)$$

4.3 Perturbed Mean

The perturbed mean uses the same formulae to compute the mean as does the unperturbed mean. However, it uses a perturbed parameter map. To distinguish the unperturbed and perturbed maps, we add the superscript (0) to the parameter map: $\theta^{(0)}$. The perturbation is computed from S_θ , which we will treat as a matrix, and we will treat $\theta^{(0)}$ and θ as vectors. The parameter map perturbation equation is then given in linear algebra notation as:

$$\vec{\theta} = \vec{\theta}^{(0)} + \underline{S}_\theta \vec{\epsilon} \quad (12)$$

Where $\vec{\epsilon}$ is a vector of uncorrelated uniform random variables distributed evenly between $-\sqrt{3}$ and $+\sqrt{3}$. This distribution has zero mean and unit variance, but, unlike the unit normal, it has finite bounds. It is, therefore, the case that the global error covariance of the parameter map is given by:

$$\text{cov}(\vec{\theta} - \vec{\theta}^{(0)}) = \langle (\vec{\theta} - \vec{\theta}^{(0)})(\vec{\theta} - \vec{\theta}^{(0)})^T \rangle = \underline{S}_\theta \langle \vec{\epsilon} \vec{\epsilon}^T \rangle \underline{S}_\theta^T = \underline{S}_\theta \underline{S}_\theta^T \quad (13)$$

This equation directly ties the runtime perturbations to the pre-computed errors in the parameter map θ via S_θ .

The perturbed parameter map can then be used along with equations from Subsection 4.1 to compute the perturbed mean flux at every grid point.

We note that the random number generator used to produce $\vec{\epsilon}$ is seeded with the scenario ID number, 1-999.

4.4 Monte Carlo Scenarios

To create a Monte Carlo scenario, one begins by perturbing the parameter map according to Eq. (12). Next, one must initialize the dynamic state of the radiation belts using a state history matrix. The state history is initialized to a series of vectors \vec{q}_t of Gaussian white noise (uncorrelated unit normal random variables). Then, for a model-dependent conditioning time [see *O'Brien, 2012*], the state history is updated using the autoregressive equations:

$$\vec{q}_t = \sum_{i=1}^{N_G} \underline{G}_i \vec{q}_{t-\tau_i} + \underline{C} \vec{\eta}_t, \quad (14)$$

where the \underline{G} and \underline{C} matrices and τ_i are part of the runtime tables, and $\vec{\eta}_t$ is a series of Gaussian white-noise vectors. Through a set of relationships given in *O'Brien [2012]* the \underline{G} and \underline{C} matrices tie the sequence of \vec{q}_t 's to the observed spatiotemporal correlations of the fluxes. After advancing Eq. (14) for the prescribed conditioning time, the state history is expected to represent the spatiotemporal lag correlations in the associated radiation belts. However, \vec{q}_t is only a small state vector, and it must be converted into flux on the model grid via a three-step process.

The first step in converting the state vector into flux is expanding it onto the model grid. The state vector \vec{q}_t is actually a vector of amplitudes of a set of principal components of spatial variation. The principal component matrix \underline{Q} (provided in the runtime tables) converts from the state vector to the normalized fluxes \vec{z}_t :

$$\vec{z}_t = \underline{Q} \vec{q}_t. \quad (15)$$

By construction, each q and each z has a long-term statistical distribution that is also a unit normal, and the q 's are uncorrelated with each other. Thus, the spatial covariance of the normalized fluxes is given by:

$$\underline{\Sigma} = \text{cov}(\vec{z}_t) = \langle \vec{z}_t \vec{z}_t^T \rangle = \underline{Q} \langle \vec{q}_t \vec{q}_t^T \rangle \underline{Q}^T = \underline{Q} \underline{Q}^T. \quad (16)$$

This equation directly ties the observed spatial covariance ($\underline{\Sigma}$) of the normalized fluxes to the dynamic states generated by the Monte Carlo scenarios.

The second and third steps un-normalize each of the z 's. The second step converts each z into its corresponding probability level u on the $[0,1]$ domain, according to:

$$u = \Phi(z). \quad (17)$$

The final step is to convert each u into a flux using the inverse of the cumulative distribution function at the grid point, i.e., using the same procedure in Subsection 4.2.

The global Monte Carlo state advances in time according to the time step in Table 3 and Eq. (14). The conversion from \vec{q}_t to flux is performed at each grid point at each time step.

Table 3. Monte Carlo Parameters for AE9 and AP9

Monte Carlo Parameter	AE9	AP9
Number of Principal Components (N_q)	8	9
Number of persistence matrices (N_G)	6	4
time step (δt , days)	1	7
Time lags ($\tau_i \delta t$, days)	1, 7, 14, 27, 183, 365	7, 28, 182, 364
Conditioning time (days)	1253.1	1503.5

We note that the random number generator used to initialize the state vector history \vec{q}_t and to produce $\vec{\eta}_t$ is seeded with the scenario ID number, 1–999. Also, the perturbed parameter map is the same one produced for the same scenario ID number for a perturbed mean state, i.e., in Subsection 4.3.

5. Projecting the Environment onto the Spacecraft

5.1 Fast Coordinates

For the drift invariants, Φ and h_{min} , the full coordinate calculation in the Olson-Pfizer quiet field [Olson and Pfizer, 1977] is too slow even on modern computers to be practical to do on demand in an interactive application for any significant number of telemetry points. We solve this problem by using a fast field-line tracer [Pfizer, 1991; 1995] and separate neural networks trained on previously computed databases of either Φ or h_{min} . The neural networks are trained to be high-precision replacements for the full drift trace. In addition, we introduced algorithms to identify and exclude Shabansky (bifurcated) drift orbits, as well as identifying points that are outside the domain of the model (which a drift shell integral would reveal, but which a neural network would mistake for valid inputs). The details of this technique are given in O'Brien and Huston, 2013.

We note that the use of neural networks to avoid the full drift invariant calculation was developed simultaneously by our partners at Los Alamos National Lab for a dynamic external field model [Koller et al., 2009; Koller and Zaharia, 2011]. Our networks are similar in concept. However, they address a simpler, quiet magnetic field model but have had to meet tighter requirements in terms of error performance and boundary definitions.

5.2 Interpolation onto Target Magnetic Coordinates

An essential step in determining the requested flux at the spacecraft location is determining the differential, unidirectional flux at a specific energy and direction of incidence at the spacecraft location. For V1.0, we do not account for finite gyroradius effects (such as the East-West effect). The model provides the differential, unidirectional flux in $\#/\text{cm}^2/\text{sr}/\text{s}/\text{MeV}$ on a grid in drift invariants. Interpolation is performed in a linear sense onto a grid of local coordinates, i.e., local pitch angles. Using our fast coordinate calculators, we can quickly obtain K , h_{min} , Φ , or L_m , or α_{eq} for any given location and direction. In the case of the AE9 and AP9 models, we begin with K , h_{min} , and determine whether the requested point falls within the low-altitude grid. If so, we proceed with the interpolation using the low-altitude grid. If not, we compute Φ and determine whether the point falls in the high-altitude grid.

Regardless of which coordinate grid we use, we compute the weights used for a linear interpolation as if we are multiplying the entire global state by a vector of weights (v). Of course, because only a small number of grid points is actually used to interpolate to a desired target location/direction, the weights are very sparse. The linear interpolation is performed in terms of the product of separate 1-D linear basis functions defined in the grid. Each linear basis function provided has the following form:

$$v(x) = \begin{cases} \frac{x-x_{low}}{x_{mid}-x_{low}} & x_{low} < x < x_{mid} \\ \frac{x_{high}-x}{x_{high}-x_{mid}} & x_{mid} < x < x_{high} \\ 0 & \text{otherwise} \end{cases} \quad (18)$$

The flux j at a given set of 3 coordinates $(x^{(1)}, x^{(2)}, x^{(3)})$ is then:

$$j(x^{(1)}, x^{(2)}, x^{(3)}) = \sum_{i,k,l} v_i^{(1)}(x^{(1)}) v_k^{(2)}(x^{(2)}) v_l^{(3)}(x^{(3)}) \hat{j}_{ikl}. \quad (19)$$

where \hat{j}_{ikl} is the model flux at the grid point i,k,l , computed in Section 4.

5.3 Energy Integrals

The user can request differential, integral, or “wide differential” energy channels. In the latter two cases, weights must be computed to provide energy integrals either from a lower bound to infinity (effectively the upper end of the model range) or between two specified energies, respectively. The integral energy weights are computed by replacing $v_i^{(1)}$ in Eq. (19) with an appropriate integral of Eq. (18). The “wide differential” weights are computed by subtracting the integral weights computed for the upper energy limit from the integral weights computed for the lower energy limit, and then dividing by the difference between the channel energy limits. That is, the wide differential channel is computed as the difference of two integral channels divided by the energy bandwidth.

5.4 Angle Integrals

Typically, the user requests an omnidirectional flux (J), which is taken to be an integral of directional fluxes over the span of local pitch angles. Local pitch angle integrals turn out to be integrals along a trajectory in the 2nd and 3rd coordinates because both depend on local pitch angle, α . This is approximated by an integral in α between fluxes interpolated onto a local grid in α :

$$J = 4\pi \int_0^{\pi} j(x^{(1)}, x^{(2)}(\alpha), x^{(3)}(\alpha)) \sin \alpha d\alpha \quad (20)$$

The integral is approximated linearly at 5°, 10°, 20°, ..., 90°:

$$J \approx \sum_m w_m j(x^{(1)}, x^{(2)}(\alpha_m), x^{(3)}(\alpha_m)). \quad (21)$$

5.5 Combined Interpolation and Integration

The weights w_m are combined with the linear basis functions v_i , v_j , and v_k , into a composite weight h_n that provides the combined weight for each flux on the grid, accounting for energy and directional integrals. This vector of weights h_n spans the coordinate grid and captures all the interpolation and integration required to produce the n^{th} energy channel. A separate h_n is computed for each energy channel using the same weights for the 2nd and 3rd coordinates.

The complete weights for the various energy channels are then combined together into a matrix H that spans the grid in rows and the requested energy channels in columns. Thus, if the global flux state is represented as a vector of fluxes \vec{j} spanning the grid, the fluxes \vec{J} in the requested energy channels are given by

$$\vec{J}_t = \underline{H}_t \vec{j}. \quad (22)$$

By judicious use of sparse matrices, H is computed first and then only the needed components of \vec{j} are computed, thus saving evaluations of the statistical functions in Section 4. We have included subscripts t in Eq. (22) to indicate that the weights must be recomputed every time step along the spacecraft trajectory (the model fluxes \vec{j} may also change, but we will address that in the next section).

5.6 Temporal Interpolation (dynamic scenarios only)

The final consideration for interpolating the global state onto the spacecraft trajectory and the desired energy and directional channels is addressing time interpolation in the dynamic Monte Carlo scenarios. Time interpolation is handled linearly in the computed flux channel. That is, the global state is computed only at fiducial times (integer time steps), and the requested flux channels are computed by linearly interpolating the derived local flux \vec{J}_t in time. This is equivalent to allowing the global state to progress linearly between fiducial times.

5.7 Plasma-Radiation Stitching in Energy

One complication arises from the separation of the electron and proton models into radiation models (AE9/AP9) and plasma models (SPME/SPMH): integral and wide differential channels that start in the plasma energy range but extend into the radiation energy range. In V1.0, we resolve this issue in post processing (see the application user's guide [Roth, 2013]). The approach is to request a wide differential channel from the plasma model and a complementary integral or wide differential channel from the radiation model.

If the lowest energy in the radiation model is denoted E_0 , then an integral channel at E_1 is given by:

$$J(E \geq E_1) = J_{\text{plasma}}(E_1 < E < E_0)[E_0 - E_1] + J_{\text{radiation}}(E \geq E_0). \quad (23)$$

As in Subsection 5.3, a wide differential channel is computed from the difference of two integral channels divided by the energy bandwidth.

6. Summary

This document presents a detailed description of the AE9/AP9/SPM runtime algorithms for the V1.0 release. These algorithms address the mathematical and geophysical representation of the radiation and plasma climatology models. The set of models can specify the design environments for total radiation dose, internal charging, proton single-event effects, and plasma dose effects for any Earth orbit. The runtime algorithms provide mechanisms for estimating the probability of occurrence for various hazardous conditions due both to dynamic variability and to model uncertainty.

References

- [AE9/AP9 Tech Doc] AE9/AP9/SPM V1.1 Radiation Environment Model: Technical Documentation, Air Force Research Lab Technical Report, *in preparation*.
- Finlay et al., "International Geomagnetic Reference Field: the eleventh generation." International Association of Geomagnetism and Aeronomy, Working Group V-MOD. *Geophys. J. Int.*, **183**(3), pp 1216-1230, December 2010. DOI: 10.1111/j.1365-246X.2010.04804.x.
- Ginet., G. P., T. P. O'Brien, S. L. Huston, W. R. Johnston, T. B. Guild, R. Friedel, C. D. Lindstrom, C. J. Roth, P. Whelan, R. A. Quinn, D. Madden, S. Morley, and Y. J. Su, "AE9, AP9 and SPM: New models for specifying the trapped energetic particle and space plasma environment," *Space Sci. Rev.*, 2013, doi:10.1007/s11214-013-9964-y.
- Koller, J. , G. D. Reeves and R. H. W. Friedel, "LANL* V1.0: a radiation belt drift shell model suitable for real-time and reanalysis applications," *Geosci. Model Dev.*, **2**, 113-122, 2009.
- Koller, J. and S. Zaharia, "LANL* V2.0: global modeling and validation," *Geosci. Model Dev.*, **4**, 669-675, doi:10.5194/gmd-4-669-2011, 2011.
- O'Brien, T. P., "A framework for next-generation radiation belt models," *Space Weather*, **3**, S07B02, doi:10.1029/2005SW000151, 2005.
- O'Brien, T. P., "Preliminary algorithms for computation of mission aggregate and worst-case fluxes from the empirical AE-9/AP-9 Model," TOR-2007(3905)-18, The Aerospace Corporation, El Segundo, CA, 2007.
- O'Brien, T. P., "Adding Multiple Time Lags to AE9/AP9 V1.0," TOR-2012(1237)-3, 2012a, The Aerospace Corporation, El Segundo, CA.
- O'Brien, T. P., "Generation of AE9/AP9/SPM V1.0 Runtime Tables," 2013, TOR-2014-00295, The Aerospace Corporation, El Segundo, CA 2013.
- O'Brien, T. P. and T. B. Guild, "Trapped Electron Model 2 (TEM-2)," TR-2010(3905)-2, Aerospace Corporation, El Segundo, CA, 2010.
- O'Brien, T. P. and S. L. Huston, "Fast Coordinates used in AE9/AP9 V1.0 for Invariant Coordinate Calculations," TOR 2013(1237)-1, The Aerospace Corporation, El Segundo, CA, 2013.
- Olson, W. P., and K. A. Pfitzer, "Magnetospheric Magnetic Field Modeling," Annual Scientific Report, Air Force Office of Scientific Research contract F44620-75-C-0033, McDonnell Douglas Astronautics Co., Huntington Beach, CA, 1977.
- Pfitzer, K. A., "Improved models of the inner and outer radiation belts," Scientific Report No. 1, Phillips Laboratory technical report PL-TR-91-2187, Hanscom AFB, MA, 1991.

Pfitzer, K. A. "Improved models of the inner and outer radiation belts," Final Report, Phillips Laboratory technical report PL-TR-96-2050, Hanscom AFB, MA, 1995.

Roth, C., "AE9/AP9/SPM V1.0 Radiation Environment Model: User's Guide," Nov. 2013, AFRL Technical Report, submitted.