AEROSPACE REPORT NO.
TOR-2012(1237)-3

Adding Multiple Time Lags to AE9/AP9 V1.0

August 10, 2012

Paul O'Brien
Space Science Applications Laboratory
Physical Sciences Laboratories

Prepared for:

Space and Missile Systems Center
Air Force Space Command
483 N. Aviation Blvd.
El Segundo, CA 90245-2808

Contract No. FA8802-09-C-0001

Authorized by: Engineering and Technology Group

AEROSPACE
Assuring Space Mission Success

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.

AEROSPACE REPORT NO.
TOR-2012(1237)-3

Adding Multiple Time Lags to AE9/AP9 V1.0

August 10, 2012

Paul O’Brien
Space Science Applications Laboratory
Physical Sciences Laboratories

Prepared for:

Space and Missile Systems Center
Air Force Space Command
483 N. Aviation Blvd.
El Segundo, CA 90245-2808

Contract No. FA8802-09-C-0001

Authorized by: Engineering and Technology Group

AEROSPACE REPORT NO.
TOR-2012(1237)-3

Adding Multiple Time Lags to AE9/AP9 V1.0

Approved by:

[illegible]

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

SC-2175(5666, 9, JS)

Abstract

The new AE9/AP9 models of the natural radiation belts at Earth capture belt dynamics via Monte Carlo scenarios. Because the radiation belts exhibit dynamics on timescales from hours to years, multiple time lags are needed in the dynamic Monte Carlo scenario generators. In the predecessor models (alpha and beta versions), time evolution carried information from only one prior state, that is, only one time lag. This document describes the calculations and challenges associated with adding multiple time lags to the autoregressive part of the V1.0 AE9/AP9 Monte Carlo scenario generator.

This page intentionally blank.

Contents

1 Introduction

O'Brien and Guild [2010] (hereafter TEM2) provides a first-order autoregressive (AR-1) equation for the time evolution of the principal components ($\vec{q}$) of the spatial variation of the radiation belts:

$$\vec{q}_{t+\delta t} = \underline{\underline{G}}\vec{q}_t + \underline{\underline{C}}\vec{\eta}_{t+\delta t}, \tag{1}$$

In this document, we extend that equation to an arbitrary set of N_G time lags, and we offset the time notation by δt to make the meaning of the time lags more clear:

$$\vec{q}_t = \sum_{i=1}^{N_G} \underline{\underline{G}}_i \vec{q}_{t-\tau_i} + \underline{\underline{C}}\vec{\eta}_t, \tag{2}$$

This is an AR-N process [for discussion, see, e.g., *Neumaier and Schneider*, 2001].

The AR-1 is given by $N_G = 1$ and $\tau_1 = 0$. We assume that the time lags $\vec{\tau}$ will not necessarily be uniformly spaced. For example, $\vec{\tau}$ might be [6 hours, 1 day, 9 days, 27 days, 6 months, 1 year]. We note that $\tau_1 \equiv \delta t$ becomes the basic unit of time stepping, and that we will enforce that all other $\tau_i = T_i \delta t$, where T_i is an integer, and $T_1 \equiv 1$. In our example $\vec{\tau}$, $\vec{T}$ is given by [1, 4, 36, 108, 730, 1460]. We will also define an unlisted value $T_0 \equiv 0$.

The addition of arbitrary time lags presents three problems which will be addressed below: how to compute the $\underline{\underline{G}}_i$ and $\underline{\underline{C}}$ matrices, how to initialize $\vec{q}$ to begin a scenario, and how to store the history of $\vec{q}_t$ to support evaluation of (2). This document concludes with some estimates of the size of the matrices that must be computed and manipulated.

2 Computing AR-N Matrices

In TEM2, the AR-1 equations are derived from spatial ($\underline{\underline{\Sigma}}$) and one-day-lag ($\underline{\underline{R}}$) covariance matrices, transformed into corresponding matrices for the principal components via $\underline{\underline{Q}}$ and $\underline{\underline{Q}}^\dagger$:

$$\begin{aligned}
\underline{\underline{Q}}\,\underline{\underline{Q}}^T &\approx \underline{\underline{\Sigma}}, &(3)\\
\underline{\underline{Q}}^\dagger \underline{\underline{Q}} &= \underline{\underline{I}}, &(4)\\
\underline{\underline{G}} &= \left[\underline{\underline{Q}}^\dagger\, \underline{\underline{R}}^T \left(\underline{\underline{Q}}^\dagger\right)^T\right]^{\delta t/T}, &(5)\\
\underline{\underline{C}} &= \left(\underline{\underline{I}} - \underline{\underline{G}}\,\underline{\underline{G}}^T\right)^{1/2}. &(6)
\end{aligned}$$

(Note: the dagger (†) denotes a matrix pseudo-inverse, which is typically computed via singular value decomposition).

For the AR-N model, we will need a set of lag covariance matrices. Also, we will not be able to use 1-day time lag covariance matrix and adjust to δt, but rather we will have to use lag covariance matrices that exactly match certain time lags derived from $\vec{\tau}$. We define a generic Gaussian covariance matrix for the fluxes (i.e., a linear covariance matrix for the $\vec{z}$'s, which represent particle fluxes transformed to Gaussian variables) and its counterpart for the principal components ($\vec{q}$'s) as:

$$\begin{aligned}
\hat{\underline{\underline{R}}}_M &= \left\langle \vec{z}_t \vec{z}^T_{t-M\delta t}\right\rangle = \underline{\underline{Q}} \left\langle \vec{q}_t \vec{q}^T_{t-M\delta t}\right\rangle \underline{\underline{Q}}^T, &(7)\\
\underline{\underline{R}}_M &= \left\langle \vec{q}_t \vec{q}^T_{t-M\delta t}\right\rangle = \underline{\underline{Q}}^\dagger \hat{\underline{\underline{R}}}_M \left(\underline{\underline{Q}}^\dagger\right)^T. &(8)
\end{aligned}$$

We note that $\hat{\underline{\underline{R}}}_0 = \underline{\underline{\Sigma}}$ and $\underline{\underline{R}}_0 = I$ in TEM2, and that $R_{-M} = R_M^T$. The needed $\hat{\underline{\underline{R}}}_M$ matrices will have to be determined from data or from a reanalysis.

We create a series of matrix equations by taking the expected value of $\vec{q}_t \vec{q}^T_{t-T_j\delta t}$:

$$\left\langle \vec{q}_t \vec{q}^T_{t-T_j\delta t}\right\rangle = \underline{\underline{R}}_{T_j} = \sum_{i=1}^{N_G} \underline{\underline{G}}_i \underline{\underline{R}}_{T_j-T_i} + \left\{ \begin{array}{cl} \underline{\underline{C}}\,\underline{\underline{C}}^T & T_j = 0 \\ 0 & \text{otherwise} \end{array}\right., \quad (9)$$

where the $\underline{\underline{C}}\,\underline{\underline{C}}^T$ term arises from $\left\langle \vec{\eta}_t \vec{q}_t^T\right\rangle$ because $\vec{\eta}_t$ is uncorrelated with all prior $\vec{q}_t$.

To put the equations in the usual form where we left-multiply the unknowns, we will use:

$$\underline{\underline{R}}^T_{T_j} = \sum_{i=1}^{N_G} \underline{\underline{R}}_{T_i-T_j} \underline{\underline{G}}^T_i + \left\{ \begin{array}{cl} \underline{\underline{C}}\,\underline{\underline{C}}^T & T_j = 0 \\ 0 & \text{otherwise} \end{array}\right. \quad (10)$$

We will solve for all the $\underline{\underline{G}}_i$ simultaneously, and then solve for $\underline{\underline{C}}$.

$$\begin{bmatrix} \underline{\underline{R}}^T_{T_1} \\ \underline{\underline{R}}^T_{T_2} \\ \underline{\underline{R}}^T_{T_3} \\ \vdots \\ \underline{\underline{R}}^T_{T_{N_G}} \end{bmatrix} = \begin{bmatrix} \underline{\underline{I}} & \underline{\underline{R}}_{T_2-T_1} & \underline{\underline{R}}_{T_3-T_1} & \cdots & \underline{\underline{R}}_{T_{N_G}-T_1} & \\ \underline{\underline{R}}^T_{T_2-T_1} & \underline{\underline{I}} & \underline{\underline{R}}_{T_3-T_2} & \cdots & \underline{\underline{R}}_{T_{N_G}-T_2} & \\ \underline{\underline{R}}^T_{T_3-T_1} & \underline{\underline{R}}^T_{T_2-T_1} & \underline{\underline{I}} & \cdots & \underline{\underline{R}}_{T_{N_G}-T_3} & \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ \underline{\underline{R}}^T_{T_{N_G}-T_1} & \underline{\underline{R}}^T_{T_{N_G}-T_2} & \underline{\underline{R}}^T_{T_{N_G}-T_3} & \cdots & \underline{\underline{I}} & \end{bmatrix} \begin{bmatrix} \underline{\underline{G}}^T_1 \\ \underline{\underline{G}}^T_2 \\ \underline{\underline{G}}^T_3 \\ \vdots \\ \underline{\underline{G}}^T_{N_G} \end{bmatrix} \tag{11}$$

The number of unique $\hat{\underline{\underline{R}}}_M$ matrices required is less than or equal to $1 + (N_G + 1)N_G/2$, of which all but one ($\hat{\underline{\underline{R}}}_0 = \underline{\underline{\Sigma}}$) are converted to covariances in the principal components ($\underline{\underline{R}}_M$). Equation (11) is square and the matrix to be inverted is symmetric. After solving for the $\underline{\underline{G}}_i$, we will solve for $\underline{\underline{C}}$ using the T_0 case of (9):

$$\underline{\underline{CC^T}} = \underline{\underline{I}} - \sum_{i=1}^{N_G} \underline{\underline{R}}_{T_i} \underline{\underline{G}}_i^T \tag{12}$$

A few additional manipulations can show that this expression necessarily yields a symmetric $\underline{\underline{CC^T}}$. To obtain $\underline{\underline{C}}$, we take a Cholesky or eigenvalue square root of $\underline{\underline{CC^T}}$.

3 Accuracy and Stability Criteria

We must ensure that $\underline{\underline{CC}}^T$ is symmetric and does not have any eigenvalues with magnitude larger than 1. Also, because (2) is only an approximation of the time series dynamics, it is possible that the solution $\underline{\underline{G}}_i$ is unstable. The stability condition is that none of the eigenvalues of the total time evolution matrix have magnitude 1 or greater.

The first (accuracy) condition is:

$$\left| \left(\underline{\underline{CC}}^T\right)_{ij} - \left(\underline{\underline{CC}}^T\right)_{ji} \right| < \epsilon, \tag{13}$$

where $\epsilon << 1$ is some small error tolerance, e.g., 0.01.

The second (stability) condition is:

$$\begin{aligned} |\Lambda_{jj}| &< 1, & (14)\\ \underline{\underline{CC}}^T &= \underline{\underline{V\Lambda V}}^T, & (15)\end{aligned}$$

where $\underline{\underline{\Lambda}}$ is diagonal, and $\underline{\underline{VV}}^T = \underline{\underline{I}}$.

The final (stability) condition is:

$$\begin{aligned} \left|\tilde{\Lambda_{jj}}\right| &< 1, & (16)\\ \underline{\underline{\tilde{G}\tilde{G}}}^T &= \underline{\underline{\tilde{V}\tilde{\Lambda}\tilde{V}}}^T, & (17)\\ \underline{\underline{\tilde{G}}} &= \begin{pmatrix} \underline{\underline{\tilde{G}}}_1 & \underline{\underline{\tilde{G}}}_2 & \cdots & \underline{\underline{\tilde{G}}}_{N_G-1} & \underline{\underline{\tilde{G}}}_{N_G} \\ \underline{\underline{I}} & \underline{\underline{0}} & \cdots & \underline{\underline{0}} & \underline{\underline{0}} \\ \underline{\underline{0}} & \underline{\underline{I}} & \cdots & \underline{\underline{0}} & \underline{\underline{0}} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \underline{\underline{0}} & \underline{\underline{0}} & \cdots & \underline{\underline{I}} & \underline{\underline{0}} \end{pmatrix}, & (18)\\ \underline{\underline{\tilde{G}}}_t &= \begin{cases} \underline{\underline{G}}_i, & t = T_i \\ \underline{\underline{0}}, & t \notin \left\{\vec{T}\right\} \end{cases} & (19)\end{aligned}$$

where $\underline{\underline{\tilde{\Lambda}}}$ is diagonal, and $\underline{\underline{\tilde{V}\tilde{V}}}^T = \underline{\underline{I}}$. (Implementation note: $\underline{\underline{\tilde{G}}}$ is sparse, and can be handled with sparse matrix algorithms. In particular, Matlab's "eigs" function can be used to compute only the largest magnitude eigenvalue of a sparse matrix).

In practice, empirically determined lag covariance matrices sometimes lead to time evolution solutions that do not meet the stability criteria. In order to ensure that the stability criteria are met, a tapering factor is used. A given lag correlation coefficient is multiplied by a value γ^{T_i}, where $\gamma \leq 1$ such that covariances at longer lags are suppressed relative to those at shorter lags. This effectively suppresses spurious empirical correlations. In the final version of the V1.0 runtime tables, the stability conditions were met for $\gamma = 1$, thus requiring no tapering.

4 Computing the lagged covariances on a reduced grid

Computing covariance matrices is one of the most computationally intensive parts of the back-end processing necessary to generate the AE9/AP9 runtime tables. One way to reduce the computing power needed is to reduce the size of the grid on which the covariance matrices are computed.

We already implement a version of this for the spatial covariance matrix $\underline{\underline{\Sigma}}$, from which we compute $\underline{\underline{Q}}$. In that process, we decimate the grid, compute a reduced $\underline{\underline{\breve{\Sigma}}}$ and a reduced $\underline{\underline{\breve{Q}}}$, and then interpolate the columns of $\underline{\underline{\breve{Q}}}$ onto the full grid to obtain $\underline{\underline{Q}}$ (with some normalization corrections along the way).

Similarly, we can compute $\underline{\underline{\hat{R}}}_{T_j}$ on a reduced grid: $\underline{\underline{\breve{R}}}_{T_j}$, and, using only the corresponding rows of $\underline{\underline{Q}}$, i.e., $\underline{\underline{\breve{Q}}}$, we can obtain $\underline{\underline{R}}_{T_j}$:

$$\underline{\underline{R}}_{T_j} = \underline{\underline{\breve{Q}}}^{\dagger} \underline{\underline{\breve{R}}}_{T_j} \left(\underline{\underline{\breve{Q}}}^{\dagger} \right)^{T} . \tag{20}$$

As it turns out, we only need slightly more rows in $\underline{\underline{\breve{R}}}$ than N_q. For example, for the AE9 beta runtime tables (AE9VB/rev d), with $N_q = 9$, 9 or more randomly selected rows of $\underline{\underline{Q}}$ typically lead to a well-defined $\underline{\underline{\breve{Q}}}^{\dagger}$. Therefore, the lag covariances can be computed on a dramatically decimated grid, and, to some extent, the points that make up that decimated grid can be selected based on sample size. Also, there is no need to interpolate, as we only need the covariance matrices for the principal components for $T_j \neq 0$.

5 Initializing a Scenario

The "brute force" approach to initializing a scenario is to seed a series of $\vec{q}_{-\delta T}$ to $\vec{q}_{-T_{N_G}\delta t}$ with Gaussian white noise and then step forward (2) many ($>> T_{N_G}$) times. As a rough guide, the initialization will have to evolve for several times the longest lived eigenmode of $\underline{\underline{\tilde{G}}}$. This approach could conceivably leave out long-lag perturbations. Initializing for $N/\left(1 - \max\left|\tilde{\Lambda}_{jj}\right|\right)$ time steps should shrink long-lag perturbations to have magnitude e^{-N}; so, $N \sim 5$, would suppress such effects to be smaller than 0.01. Initial estimates suggest that for typical values of N_q and T_{N_G}, this initialization approach would take no more than 1-2 seconds. We can verify the validity of this scheme by initializing multiple instances of the $\vec{q}_t$ history to distinct samples of Gaussian white noise, but then evolving each instance with the same series $\vec{\eta}_t$. By determining when the different instances converge, we can determine how long to iterate (2) to initialize the data.

A more rigorous approach would be to initialize the meta-state vector $\hat{\vec{q}}^T = [\vec{q}^T_{-\delta t}, \vec{q}^T_{-T_2\delta t}, \cdots, \vec{q}^T_{T_{N_G}\delta t}]$ with a properly-conditioned multivariate Gaussian so that $\hat{\vec{q}}$ preserves the covariance $\left\langle \hat{\vec{q}}\hat{\vec{q}}^T \right\rangle$. This would involve building and factoring a square matrix $N_q \times T_{N_G}$ on a side. Such a matrix would include $\underline{\underline{R}}_M$ for all M from 0 to T_{N_G}, which is far beyond what is needed to compute $\underline{\underline{G}}_i$ and $\underline{\underline{C}}$ because $T_{N_G} >> 1 + (N_G + 1)N_G/2$. It may be possible to solve this problem by only building and factoring the sub-matrices that make up the large covariance matrix, but so far my attempts to divine such an approach have failed.

6 Maintaining History of $\vec{q}$

To evaluate (2), we need to store a history of $\vec{q}_t$ for at least $T_{N_G} + 1$ times. The natural memory structure for this is a loop queue: data does not shift each time step, rather, a pointer that defines $\vec{q}_t$ moves through the loop. After a time value falls off the end of the list, its memory address is reused for the next $\vec{q}_t$.

7 Estimated Sizes of Matrices

Typical values are $N_q = 10$, $\delta t = 6$ hours, $N_G = 6$, $\tau_{N_G} = 1$ year, $T_{N_G} = 1460$. Therefore, a typical $\underline{\underline{R}}_{T_i}$, $\underline{\underline{G}}$, or $\underline{\underline{C}}$ matrix is 10×10, 22 $\hat{\underline{\underline{R}}}_{T_i}$ matrices are needed, and there are 600 unknowns in (11). The large matrices are all sparse, and can be solved efficiently by solvers that account for sparseness.

The loop queue for $\vec{q}_t$ must hold 14,600 entries, or ~ 115 kilobytes at double precision (64-bit). The large matrix in (11) is only 60×60.

The slowest part of building the tables for the AR-N model is computing the 22 $\hat{\underline{\underline{R}}}_{T_j}$. Recent code changes to the back-end processing have enable us to limit the size of $\hat{\underline{\underline{R}}}_{T_j}$ computed from the data. Currently, $\hat{\underline{\underline{R}}}_{T_j}$ is computed on a limited grid that has at most about 14,000 grid points. To keep development time under control, the limit could be further reduced without significant loss of precision (14,000 points is still far larger than the $N_q \sim 10$ principal components that are retained).

References

O'Brien, T.P., and T.B. Guild, Trapped Electron Model 2 (TEM-2), TR-2010(3905)-2, The Aerospace Corporation, El Segundo, CA, 2010.

Neumaier, A., and T. Schneider, Estimation of parameters and eigenmodes of multivariate autoregressive models, *ACM Trans. Math. Soft., 27*(1), 27-57, 2001.