AEROSPACE REPORT NO.
ATR-2016-03268

AE9/AP9 Proton Displacement Damage Kernels (version 2)

October 12, 2016

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

Prepared for:

Sr. Vice President
Engineering and Technology Group

Authorized by: Engineering and Technology Group

Abstract

The AE9/AP9 displacement damage kernel utility is a Python application to generate kernel data files that AE9/AP9 can use to estimate proton displacement damage in Si and GaAs semiconductors. Each kernel contains all the information needed to convert from proton fluence to equivalent 1-MeV neutron fluence at various depths of Al shielding for a specific part. The aim is to provide a fast means of computing displacement damage analogous to the dose-depth calculations that are customarily provided by space environment radiation applications. The utility approximates the effects of shielding using the continuous slowing down approximation (CSDA). CSDA neglects the effects of most nuclear interactions and deflection of the particle trajectory, which can be important under some circumstances. Only spherically symmetric, aluminum shielding is supported. Displacement damage in the target material is computed by scaling the non-ionizing energy loss (NIEL) of the proton spectrum inside the shielding to the fluence of 1-MeV neutrons that would produce the same displacement damage dose. A prior displacement damage kernel was developed using a more complete physical transport through shielding, TOR-2013-00529, but was only suitable for Si semiconductors and used a fixed energy and shielding depth grid. The kernel itself is an Extensible Markup Language (XML) file produced by the utility. This report describes the algorithms used by the utility to generate the XML file, validation of the calculations, and limitations of the utility. Future versions of AE9/AP9 will be able to use such kernels to compute displacement damage for individual mission scenarios, and will thereby be able to use the existing AE9/AP9 statistical machinery to compute confidence levels for displacement damage.

Acknowledgments

The author acknowledges useful discussions with J. Srour, S. R. Messenger, B. P. Kwan, and M. D. Looper, as well as other colleagues at The Aerospace Corporation.

Contents

Figure

1. Introduction

Like many radiation effects, proton displacement damage is a linear functions of the incident proton spectrum. In *O'Brien and Kwan* [2013], we developed the concept of an AE9/AP9 kernel, which is an Extensible Markup Language (XML) file that contains the transform matrix that describes the linear transform from a fluence versus energy spectrum to a displacement damage versus depth spectrum. The XML file conforms to a standard described in *O'Brien and Whelan* [2015]. The XML file also contains metadata that describes the energy and depth grids, units, and other potentially useful information. However, the main purpose of the kernel file is to provide the transform matrix.

In this report, we update the *O'Brien and Kwan* [2013] kernel with a utility that allows the user to select Si or GaAs target material, and the user can select the energy and depth grids, as for other kernels [see, e.g., *O'Brien,* 2015]. The price of this flexibility is that the shielding effects are reduced to the continuous slowing down approximation (CSDA), which neglects all nuclear interactions and scattering. The new kernel is able to support both Si and GaAs target material through the use of non-ionizing energy loss (NIEL) tables provided with the Solar Cell Radiation Environment Analysis Models (SCREAM) software (v5.16) [*Messenger et al.*, 2012].

In the following sections, we will describe the equations used for the CSDA shielding effects and for the computation of the displacement damage behind shielding. We will then perform some validation against similar (but not identical) calculations in the ESA's Space Environment Information System (SPENVIS) tool (www.spenvis.oma.be).

All of the relevant calculations and the ability to write the AE9/AP9 standard XML file format are provided in a Python utility with a command line interface (see the Appendix) and a programmatic interface (documented in the source code itself following Python conventions).

2. The Degraded Spectrum

The calculation of the effects of shielding are identical to those described in *O'Brien* [2015]. Therefore, this is an abbreviated treatment.

We make the following assumptions to calculate the effects of shielding:

- The geometry is spherically symmetric
- The shielding material is aluminum
- Protons traveling through the shield travel in straight lines, losing energy according to the continuous slowing down approximation (CSDA)

The AP9 model will provide the incident fluence *j(E)*, and the displacement damage calculation will be performed upon the degraded differential fluence $j'(E)$. In general, the degraded differential proton fluence $j'(E;T)$ after passing through shielding *T* is a linear transform of the input differential fluence *j(E)*:

$$j'(E;T) = \int_0^\infty j(\tilde{E})G(E,\tilde{E})d\tilde{E}. \tag{1}$$

Given the CSDA assumption, the Greens function *G* is a Dirac delta function. In this approximation, a particle has a range before coming to a stop that is a monotonic function of energy, and "uses up" a quantity *T* of that range when passing through the shield. So, we can write:

$$\rho(E) = \rho(\tilde{E}) - T, \tag{2}$$

where $\tilde{E}$ is the incident energy, *E* is the degraded energy, and the function $\rho(E)$ provides the range for a particle with energy *E*. Defining $\mathcal{E}(r)$ as the energy of a proton with range *r*, we can write the energy change as:

$$E = \mathcal{E}(\rho(\tilde{E}) - T), \tag{3}$$

noting that $\mathcal{E}(r)$ is the functional inverse of $\rho(E)$.

The integral transform (1) becomes:

$$j'(E;T) = j(\mathcal{E}(\rho(E)+T))\left[\left.\frac{d\mathcal{E}}{dr}\right|_{\rho(E)}\left.\frac{d\rho}{dE}\right|_{\mathcal{E}(\rho(E)+T)}\right]^{-1}, \tag{4}$$

where we have inverted Eq. (3) to obtain $\tilde{E}=\mathcal{E}(\rho(E)+T)$, and the derivatives account for the implied Dirac delta function in $G(E,\tilde{E})$. By noting that $\frac{d\mathcal{E}}{dr}$ and $\frac{d\rho}{dE}$ are related to the linear energy transfer, *dE/dx*, also known as stopping power, we have:

$$j'(E;T)=j(\mathcal{E}(\rho(E)+T))\left.\frac{dE}{dx}\right|_{\mathcal{E}(\rho(E)+T)}\left(\left.\frac{dE}{dx}\right|_{E}\right)^{-1}. \tag{5}$$

Here range, denoted by *r,* ρ, or *x*, is given in areal mass density in g/cm^2, and energy *E* is in MeV. The areal mass density for any material is the volume mass density (g/cm^3) times the depth of shielding (cm); thus, we can treat other materials in terms of their "Al equivalent" shielding through the areal mass density. Because we are working with an assumption of spherical symmetry, the fluence is assumed to be omnidirectional (i.e., integrated over all angles of incidence), with units of protons/cm^2/MeV.

As in the SEE kernel utility [*O'Brien*, 2015], we obtain our proton range and stopping power data from the International Commission on Radiation Units and Measurements [ICRU, 1993] via the National Institute for Standards and Technology (NIST): http://physics.nist.gov/PhysRefData/Star/Text/PSTAR.html. We fit the range-energy relationship to a polynomial in the natural logarithms:

$$\begin{aligned}\ln\rho(E)=&-10.077944894\left(\frac{\ln E}{10}\right)^6+16.652344291\left(\frac{\ln E}{10}\right)^5\\&-3.5612924583\left(\frac{\ln E}{10}\right)^4-14.172779811\left(\frac{\ln E}{10}\right)^3+9.6164697150\left(\frac{\ln E}{10}\right)^2\\&+16.691495048\frac{\ln E}{10}-7.1138007166.\end{aligned} \tag{6}$$

Likewise, we fit the stopping power provided by NIST to a polynomial (which is numerically more stable than differentiating $\ln\rho(E)$):

$$\begin{aligned}\ln\frac{dE}{dx}=&-35.1341\left(\frac{\ln E}{10}\right)^8+9.93853\left(\frac{\ln E}{10}\right)^7+61.3322\left(\frac{\ln E}{10}\right)^6\\&-24.4878\left(\frac{\ln E}{10}\right)^5-25.1057\left(\frac{\ln E}{10}\right)^4+18.9366\left(\frac{\ln E}{10}\right)^3\\&-3.08627\left(\frac{\ln E}{10}\right)^2-8.50949\frac{\ln E}{10}+6.55932.\end{aligned} \tag{7}$$

The domain of applicability for both of these fits is 10^{-3} to 10^4 MeV.

Because we will perform later interpolations in terms of range rather than energy, we introduce the concept of differential fluence at a given *range* as:

$$\hat{j}(r) = j(\mathcal{E}(r)). \tag{8}$$

Thus, we can write the degraded spectrum as:

$$\hat{j}'(r;T) = \hat{j}(r+T)\left.\frac{dE}{dx}\right|_{\mathcal{E}(r+T)}\left(\left.\frac{dE}{dx}\right|_{\mathcal{E}(r)}\right)^{-1}. \tag{9}$$

3. Displacement Damage via NIEL Scaling

For proton and neutron displacement damage, the primary mechanism is nuclear recoil, which is characterized by non-ionizing energy loss (NIEL). The displacement damage dose from a spectrum of protons is given by:

$$D(T) = \int_0^\infty j'(E;T) S_p(E) dE, \tag{10}$$

where we have used the degraded spectrum $j'(E;T)$ after accounting for shielding thickness T, and $S_p(E)$ is the proton NIEL in MeV/(g/cm^2). The displacement damage dose $D(T)$ has units of MeV/g. We convert to equivalent 1-MeV neutron fluence via NIEL scaling [*Messenger et al.*, 2001]:

$$j_n(T) = \frac{D(T)}{S_n(E=1\text{ MeV})}, \tag{11}$$

where $S_n(E = 1\text{ MeV})$ is the NEIL for a 1-MeV neutron in Si or GaAs. The NIEL for a 1-MeV neutron in silicon is 0.00202 MeV/(g/cm^2) as in *O'Brien and Kwan* [2013]; in gallium arsenide, it is 5.83x10^{-4} MeV/(g/cm^2) [*Messenger et al.*, 2010]. Proton and NEIL factors are provided as auxiliary data tables in the SCREAM v5.16 distribution package. For convenience, the proton NIEL factors for Si and GaAs are fitted with log-log polynomials:

$$\ln S_p^{(\mathrm{Si})}(E) = -3.772340\times10^{-8}(\ln E)^{10} - 2.548842\times10^{-7}(\ln E)^9 + 4.819647\times10^{-6}(\ln E)^8 + 3.222443\times10^{-5}(\ln E)^7 - 2.098495\times10^{-4}(\ln E)^6 - 1.386437\times10^{-3}(\ln E)^5 + 3.200604\times10^{-3}(\ln E)^4 + 2.818910\times10^{-2}(\ln E)^3 + 2.581182\times10^{-3}(\ln E)^2 - 0.9613505\ln E - 2.774328 \tag{12}$$

$$\ln S_p^{(\mathrm{GaAs})}(E) = -3.899905\times10^{-8}(\ln E)^{10} - 1.089147\times10^{-7}(\ln E)^9 + 5.595386\times10^{-6}(\ln E)^8 + 1.315995\times10^{-5}(\ln E)^7 - 2.925291\times10^{-4}(\ln E)^6 - 6.026951\times10^{-4}(\ln E)^5 + 6.340144\times10^{-3}(\ln E)^4 + 2.050628\times10^{-2}(\ln E)^3 - 3.452284\times10^{-2}(\ln E)^2 - 0.9202241\ln E - 2.926445 \tag{13}$$

Both of these polynomials are valid from 0.2 keV to 10^3 MeV.

4. Calculating the Kernel

The displacement damage kernel itself evaluates Eq. (11) numerically for a variety of shielding depths given the target material and incident fluence. It has many similarities to the single-event effects kernel calculation in *O'Brien* [2015], from which the following borrows heavily. As explained in the Appendix, there are many user-configurable options. The user must specify the target material as Si or GaAs. The user may also specify both the incident energy grid (with *N* points) and the output shielding grid (with *M* points). The user may specify linearly spaced grids, logarithmically spaced grids, a logarithmic grid that also includes zero, or even a simple list of values to use as a grid. The shielding grid may also be provided in length units of cm, mm, or mils, rather than areal density, and the utility will convert to areal density using an Al mass density of 2.7 g/cm^3.

At a high level, the kernel can be thought of as the matrix $\underline{\underline{A}}$ in a matrix-vector equation:

$$\vec{j}_n = \underline{\underline{A}}\vec{j}. \tag{14}$$

In this equation, the equivalent 1-MeV neutron fluence $\vec{j}_n$ is specified on a grid of depths T_i:

$$j_{ni} = j_n(T_i). \tag{15}$$

The incident proton fluence is given on a grid of energies E_k:

$$j_k = j(E_k) = \hat{j}(\rho(E_k)). \tag{16}$$

The kernel matrix $\underline{\underline{A}}$ is then built to achieve:

$$j_n(T_i) = \frac{\int_0^\infty j'(E;T)S_p(E)dE}{S_n(E=1\text{ MeV})} \approx \sum_k A_{ik}j_k. \tag{17}$$

To compute A_{ik}, we must include the estimate of $j'(E;T)$ from $j(E)$ as another weighted sum (a linear interpolation):

$$\frac{\int_0^\infty j'(E;T)S_p(E)dE}{S_n(E=1\text{ MeV})} \approx \sum_{k'} \frac{S_p(E_{k'})}{S_n(E=1\text{ MeV})} j'_{ik'}\Delta E_{k'} \approx \sum_k A_{ik}j_k \tag{18}$$

$$j'_{ik'} = j'(E_{k'};T_i) = \hat{j}'(\rho(E_{k'});T_i) \approx \sum_k j_k \left.\frac{dE}{dx}\right|_{E_k} \left(\left.\frac{dE}{dx}\right|_{E_{k'}}\right)^{-1} v_{k'k;i} = D_{k'k;i}j_k \tag{19}$$

$$D_{k'k;i} = \left(\left.\frac{dE}{dx}\right|_{E_{k'}}\right)^{-1} \sum_k \left.\frac{dE}{dx}\right|_{E_k} v_{k'k;i}, \tag{20}$$

where $\Delta E_{k'}$ is a trapezoidal integration weight, and $v_{k'k;i}$ is an interpolating function:

$$\Delta E_{k'} = \begin{cases} \frac{E_2 - E_1}{2} & k' = 1 \\ \frac{E_N - E_{N-1}}{2} & k' = N \\ \frac{E_{k'+1} - E_{k'-1}}{2} & \text{otherwise} \end{cases} \tag{21}$$

$$v_{k'k;i} = v(E_{k'}, E_k; T_i) = \begin{cases} \frac{r_{k'} + T_i - r_{k-1}}{r_k - r_{k-1}} & r_{k-1} < r_{k'} + T_i \le r_k \\ \frac{r_{k+1} - r_{k'} - T_i}{r_{k+1} - r_k} & r_k \le r_{k'} + T_i < r_{k+1} \\ 0 & \begin{array}{c} r_{k'} + T_i \le r_1 \\ r_{k'} + T_i \ge r_N \\ \text{otherwise} \end{array} \end{cases} \tag{22}$$

$$r_k = \rho(E_k) \tag{23}$$

$$r_{k'} = \rho(E_{k'}) \tag{24}$$

Note that interpolation is done in the range domain: interpolating flux from incident range r_k onto degraded range $r_{k'}$, where r_k and $r_{k'}$ are points on the same grid. The logic is that a particle with degraded range $r_{k'}$ must have had incident range $r_{k'}+T$. Because $r_{k'}+T$ will not usually correspond exactly to the incident ranges r_k, it is necessary to interpolate. (It is also possible to simply define a special $r'_{k'}$ degraded range grid as the incident grid offset by T; however, this approach is numerically less stable, especially near energies that just barely make it through the shield.) We note that the interpolation treats the incident flux above the last energy grid point as zero, which is usually reasonable for a steeply falling spectrum.

Putting it all together, we have:

$$j_n(T_i) \approx \sum_{k'} \frac{S_p(E_{k'})}{S_n(E=1\text{ MeV})} \Delta E_{k'} \left(\left.\frac{dE}{dx}\right|_{E_{k'}}\right)^{-1} \sum_k \left.\frac{dE}{dx}\right|_{E_k} v_{k'k;i} j_k \approx \sum_k A_{ik} j_k. \tag{25}$$

So the matrix $\underline{\underline{A}}$ is:

$$A_{ik} = \sum_{k'} \frac{S_p(E_{k'})}{S_n(E=1\text{ MeV})} \Delta E_{k'} \left(\left.\frac{dE}{dx}\right|_{E_{k'}}\right)^{-1} \sum_k \left.\frac{dE}{dx}\right|_{E_k} v_{k'k;i}. \tag{26}$$

5. Validation

To validate the new displacement damage kernel utility, we compare to the old kernel that was based on MULASSIS [*O'Brien and Kwan*, 2013] and to SPENVIS. Both of these reference calculations are for spherical aluminum shielding around a silicon target. Like the MULASSIS kernel, SPENVIS uses a pre-computed full physics Monte Carlo simulation to determine the effects of shielding. The comparison to the old kernel is straightforward. However, comparison to SPENVIS requires some additional work because SPENVIS produces displacement damage in terms of equivalent fluence of 10-MeV protons, rather than 1-MeV neutrons. Using NIEL scaling, we can convert the 10-MeV proton fluence to equivalent 1-MeV neutron fluence by multiplying the SPENVIS proton fluence by the ratio of the NIEL for 10-MeV protons to the NIEL for 1-MeV neutrons. The SCREAM data files state that 10-MeV protons have a NIEL of 9.43E-3 MeV/(g/cm^2) in Si, and, as noted before, 1-MeV neutrons have a NIEL of 2.02E-3 MeV(/g/cm^2) in Si. So, we multiply the SPENVIS 10 MeV proton fluence by 4.67 to obtain the equivalent 1-MeV neutron fluence.

For our test, the input spectrum is the solar proton spectrum generated by SPENVIS for one year at geostationary orbit during solar maximum in the ESP-PSYCHIC model at the 95% confidence level. We use SPENVIS directly to compute the equivalent 10-MeV proton fluence, which we convert to equivalent 1-MeV neutron fluence using the NIEL scaling just described. The MULASSIS kernel and the new CSDA kernel are both evaluated using Eq. (14), where the matrix $\underline{\underline{A}}$ is provided by each kernel. The kernel evaluation pads the input spectrum with zeros as needed beyond the last energy supplied (500 MeV). Figure 1 shows the resulting damage versus depth curves for each method. The new CSDA kernel is close to, and often between, the two reference curves. The figure title provides root-mean-squared (RMS) and maximum absolute errors in the natural log fluxes, which can be interpreted as percent errors. We see that the new CSDA kernel agrees with SPENVIS to within 26% at all depths, with an RMS error of 16%. The new kernel agrees with the MULASSIS kernel to within 48%, with an RMS error of 32%. Recognizing that the MULASSIS kernel involved quite a bit of fitting to construct the kernel, it is fair to consider SPENVIS the better reference. In any case, it is clear that the new kernel's performance is well in family with the two reference approaches.

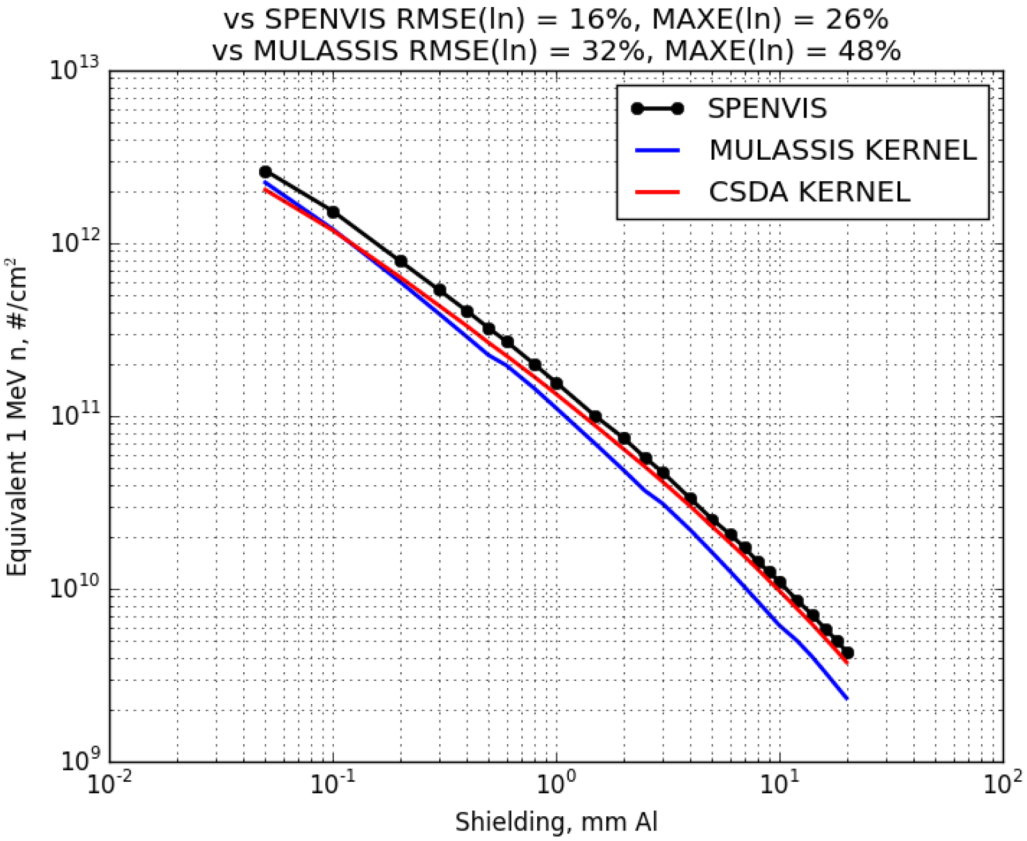

Figure 1. Comparison of output from the new "CSDA KERNEL" to the old "MULASSIS KERNEL" and SPENVIS.

6. Summary

This report described a utility for generating AE9/AP9 kernels that convert an input proton fluence spectrum into displacement damage versus depth of shielding. The new utility uses the continuous slowing down approximation (CSDA) to account for the effects of spherical aluminum shielding. The utility uses the NIEL-scaling method to compute equivalent 1-MeV neutron fluence in silicon and gallium arsenide (GaAs) semiconductors.

The new kernels generated by the utility can be used to specify the proton-related displacement damage in solid-state devices within the AE9/AP9 numerical infrastructure. This will allow users to exploit AE9/AP9's statistical machinery to compute confidence levels, rather than having to approximate the confidence levels using a third-party application.

Because the kernel assumes spherical shielding and makes the CSDA approximation, it is not as precise as a geometry-specific, full-physics calculation would be. However, it provides a valuable first-order reference for satellite design to evaluate different orbit options in terms of the typical amount of shielding required to achieve a given limit on proton-induced displacement damage.

Because the AE9/AP9 kernel utility produces its output in units of areal mass density (g/cm^2) of aluminum, it is possible to approximate the effects of other shielding materials. The equivalent aluminum shielding for a given thickness of another material can be computed by converting to areal mass density, and the kernel can then be used by assuming interactions are equivalent to those in the same areal mass density of aluminum. This approximation is quite common and should be generally acceptable for orbit trade studies. The kernel utility does not do the material conversion; rather, the conversion will be done at run time by the AE9/AP9 application using a library of shielding materials.

7. References

ICRU (1993). International Commission on Radiation Units and Measurements. ICRU Report 49, *Stopping Powers and Ranges for Protons and Alpha Particles*.

Messenger, S. R., Summers, G. P., Burke, E. A., Walters, R. J. and Xapsos, M. A. (2001), "Modeling solar cell degradation in space: A comparison of the NRL displacement damage dose and the JPL equivalent fluence approaches," *Prog. Photovolt: Res. Appl.* **9**: 103–121. doi:10.1002/pip.357

Messenger, S. R., E. M. Jackson, J. H. Warner, and R. J. Walters (2010), "SCREAM: A new code for solar cell degradation using the displacement damage dose approach," in *35th IEEE Photovoltaic Specialists Conference*, p. 1106.

Messenger, S. R., E. A. Jackson, J. H. Warner, R. J. Walters, (2012), "Advancements to SCREAM: Multiple spectrum input and ShielDDDose options," in *Proc. IEEE Photovoltaic Specialists Conf.,* **3281**, Austin, TX 2012.

O'Brien, T. P., and B. P. Kwan (2013), *Using pre-computed kernels to accelerate effects calculations for AE9/AP9: A displacement damage example*, TOR-2013-00529, The Aerospace Corporation, El Segundo, CA.

O'Brien, T. P. and Whelan, P. (2015), *Specification for radiation effects kernels for use with AE9/AP9*, ATR-2015-02436, The Aerospace Corporation, El Segundo, CA.

O'Brien, T. P. (2015), *AE9/AP9 Proton Single Event Effect Kernel Utility*, TOR-2015-02707, The Aerospace Corporation, El Segundo, CA.

Appendix—Displacement Damage Kernel Utility Command-line Interface

The AE9/AP9 displacement damage utility creates an XML kernel file for use with AE9/AP9. The target material and a kernel tag are required. Options are available to specify the shielding grid and the primary proton energy grid. The utility can be invoked on the command line as:

`python ae9ap9_dd_kernel.py` *`arguments`*

where *`arguments`* represents one or more user-supplied options.

Special arguments:
`--help` print this help message
`--test` run a predefined test

Normally required command line arguments (user supplied-values in *italics*):

Specifying the cross section:
`--Target=Si` silicon target
`--Target=GaAs` gallium arsenide target

Specifying the kernel Tag:
`--Tag=`*`tag`*
The kernel tag cannot have spaces or special characters
Exactly one tag option is required. The tag is used by AE9/AP9 to generate filenames when saving kernel outputs.

Optional command line arguments:

Specifying the output file name (optional):
`--xmlfile=`*`filename`*
The filename can be a local file or it can include a full path as well
If the filename is not specified, then one is created by appending ".xml" to the *tag*

Specifying the shielding grid (optional):
`--Tgrid=`*`first:last/`*`N` *first* to *last* in *N* evenly spaced steps
`--Tgrid=`*`first:last/`*`N/log` *first* to *last* in *N* logarithmically spaced steps
`--Tgrid=`*`first:last/`*`N/0log` zero followed by *first* to *last* in *N*-1 logarithmic steps
`--Tgrid=`*`first:step:last`* *first* to *last* with specified step size (*last* may not be included)
`--Tgrid=`*`first,second,third,...,last`* complete list of grid points
At most one `--Tgrid` option is allowed
The default is `--Tgrid=0.01:7/30/0log`

Specifying the shielding unit (optional):
`--Tunit=milsAl` mils Al
`--Tunit=cmAl` cm Al
`--Tunit=mmAl` mm Al
`--Tunit=g/cm2` g/cm^2 areal density
At most one `--Tunit` option is allowed
The default is `--Tunit=g/cm2`

Specifying the energy grid in MeV (optional):
`--Egrid=`*`first:last/`*`N` first to last in N evenly spaced steps
`--Egrid=`*`first:last/`*`N/log` first to last in N logarithmically spaced steps
`--Egrid=`*`first:step:last`* first to last with specified step size (last may not be included)
`--Egrid=`*`first,second,third,...,last`* complete list of grid points
Arguments follow same format as `--Tgrid`, except there is no `0log` option
At most one `--Egrid` option is allowed
The default is `--Egrid=1:1E3/50/log`