

**User's Guide for FVS-EMAP:
A simple tool for displaying FVS output in ArcView[®]**

**Andrew J. McMahan
Anthony Courter
Eric L. Smith**

**United States Department of Agriculture, Forest Service
Forest Health Technology Enterprise Team
Fort Collins, Colorado**

February 2002

FHTET-02-01

REFERENCE

McMahan, Andrew J.; Anthony Courter; Eric L. Smith. 2002.
User's Guide for FVS-EMAP: A simple tool for displaying FVS output in
ArcView®. FHTET 02-01. Fort Collins, CO: USDA Forest Service, State & Private
Forestry, Forest Health Protection, Forest Health Technology Enterprise Team. 30 p.

ABSTRACT

FVS-EMAP (**F**orest **V**egetation **S**imulator – **E**vent **M**onitor **A**rcView **P**roject) is a simple tool for moving simulation results from the USDA Forest Service's Forest Vegetation Simulator (FVS) into the ArcView® Geographic Information System. FVS projections of a group of stands can provide many output variables projected into several future time periods. Using FVS-EMAP, maps can be created using stand output variables for future time periods and for one or more simulation scenarios. Input to FVS-EMAP is generated by the FVS Event Monitor using COMPUTE keywords. All variables calculated via the COMPUTE keyword can be mapped using this process. A shapefile representing the stand boundaries is required as a "template" onto which FVS-EMAP will map EM output variables. This user's guide provides comprehensive instructions for using FVS-EMAP.

THE AUTHORS:

Andrew J. McMahan is a Systems Analyst, INTECS International, Inc.,
Fort Collins, CO.

Anthony Courter is a Geographer, INTECS International, Inc.,
Fort Collins, CO.

Dr. Eric L. Smith is Quantitative Analysis Program Manager, Forest Health
Technology Enterprise Team, USDA Forest Service, Fort Collins, CO.

CONTACT INFORMATION:

Eric L. Smith
USDA Forest Service, FHTET, Suite 331
2150 Centre Avenue, Building A
Fort Collins, CO 80526-1891

Phone: (970) 295-5841
E-Mail: elsmith@fs.fed.us

Contents

Introduction	1
Overview	2
Using the Event Monitor to Calculate Desired Output Variables	4
Introduction	4
Creating an Event Monitor Addfile	4
Including an Addfile to an FVS Simulation	7
Requesting a .cp2 Output File	7
Summary	8
The FVS-EMAP ArcView® Project: EMAP.apr	9
Overview	9
Function 1: Joining Event Monitor Output to a Shapefile in ArcView®	10
Legends	12
Support Files	13
View and Theme Labels	14
Function 2: Animating the Display	14
Function 3: Creating an AREALOCS Supplemental Record File from an ArcView®	14
Shapefile	14
Directory Structure	17
Saving and Sharing Your Work	18
Citations	20
Appendix: Example Event Monitor Addfiles	21
Summary_stats.kcp	22
mpb_lpp_amman.kcp	23
Spruce_Beetle_Hazard.kcp	27

Introduction

The Forest Vegetation Simulator (FVS; Wykoff et al. 1982, Van Dyck 2001a) is the USDA Forest Service's nationally supported system for projecting forest stand conditions into the future, including growth, mortality, impacts of insects, disease, and fire, and forest management activities. For many types of analyses, users are interested in projecting changes for a group of stands that are all located within the same geographic area. Geographic information system (GIS) software can display these changes in a map format or relate these changes to other geographical data. FVS-EMAP (**F**orest **V**egetation **S**imulator – **E**vent **M**onitor **A**rcView[®] **P**roject)—a custom ArcView[®] project, developed and made available by the Forest Health Technology Enterprise Team (FHTET)—is a simple tool that allows FVS variables to be imported, analyzed, and displayed in ArcView[®] GIS software (Environmental Systems Research Institute, Inc.).

Using the FVS-EMAP requires three basic steps. The first step is incorporating the FVS Event Monitor (Crookston 1990) keyword COMPUTE—along with accompanying mathematical expressions—into an FVS simulation. This will generate the output to be mapped. Collections of Event Monitor keywords and expressions may be written to and saved as a separate file (an “addfile”) to be “included” in FVS simulations. We provide a simple Event Monitor addfile (*Summary_stats.kcp*) that provides basic FVS output variables (see the Appendix), but any user-created Event Monitor application that follows the instructions included here will work. The second step is invoking the Compute2 post processor, which re-writes the COMPUTE variable output to a form usable by the FVS-EMAP. The third step is running the custom ArcView[®] project, *EMAP.apr*. This project is essentially a “blank” ArcView[®] project that provides the user a custom, “push button,” menu-driven interface to load and display the FVS Event Monitor-generated data.

In order to use the FVS-EMAP, you need:

1. Standard FVS software and stand data files;
2. An appropriate FVS Event Monitor keyword file;
3. An ARC/INFO[®] coverage or ArcView[®] shapefile which provides the information needed to map the stand boundaries of the stands being simulated;
4. ArcView[®] GIS software, version 3.2 or higher (currently available for all USDA Forest Service personal computers); and,
5. The ArcView[®] project, *EMAP.apr* (available from FHTET. See below);

FVS software and documents can be downloaded from:

<http://www.fs.fed.us/fmsc/fvs/>

The **FVS-EMAP** project software (*EMAP.apr*) and documentation updates can be downloaded from:

<http://www.fs.fed.us/foresthealth/technology/products.shtml>

or by contacting FHTET in Fort Collins, CO.

Also available at the same FHTET website are a number of **Event Monitor addfiles** and their documentation. A few are also available to be copied directly from this document's appendix. Users may also create their own Event Monitor addfiles.

This guide assumes that users have a basic knowledge of FVS and ArcView[®] GIS. The ArcView[®] project provided for this application (*EMAP.apr*) was designed in the PC version of ArcView[®] GIS 3.2.

Overview

FVS-EMAP represents a simple approach for bringing simulation results from FVS into ArcView[®]. FVS simulation projections of sets of stands can provide several output variables (for example: stand basal area, stand age, trees per acre) projected into several future time periods, resulting in a large number of possible map displays reflecting combinations of variables and time periods. Different management and disturbance scenarios for the same set of stands will create additional sets of possible displays.

Figure 1 displays a flowchart of the basic elements of the FVS-EMAP process. The standard FVS process, beginning in the upper left, uses a location file (*.loc), a stand list file (*.slf), and treelists as FVS input, resulting in FVS outputs, as requested. In the FVS-EMAP, the output desired for use in ArcView[®] comes from the process shown in the upper right. Event Monitor keywords and expressions (often written and saved as a separate file—an addfile—with a filename extension “.kcp”) are processed by the Event Monitor portion of the FVS system (at the beginning of user-requested FVS cycles).

Output from the Event Monitor COMPUTE statements is written to a file via the use of the “Compute2” post-processor available in Suppose. This output file is assigned a filename extension of “.cp2”. Additional details concerning the use of the Event Monitor and creation of .cp2 files are contained in the following sections of this document as well as in Crookston (1990), Dixon (2002), and Van Dyck (2001b).

The ArcView[®] portion of the FVS-EMAP is shown at the bottom of Figure 1. Once a .cp2 file has been created, the user opens *EMAP.apr* from within ArcView[®], and then loads the .cp2 file and the relevant shapefile into the project. Then the user can analyze, manipulate, and display the FVS data as well as any other data available for that location. Additional details concerning the use of the ArcView[®] portion of this process are contained in subsequent sections of this document.

Both the Event Monitor and the Compute2 post-processor are designed to handle simulations built and run either with or without the Parallel Processing Extension¹ (PPE) to FVS (Crookston and Stage 1991). Thus, whether you are using FVS to perform non-

¹ The Parallel Processing Extension to FVS is used to govern simulations of multiple stands *in “parallel”*; that is, every stand is grown through each FVS cycle, before a subsequent cycle is simulated. This permits stand-to-stand interactions to be simulated. Multi-stand simulations that do *not* use the PPE simulate growth one-stand-at-a-time throughout *all* cycles before simulating the next stand; that is, stands are simulated *serially*.

PPE, single stand runs; serial, multi-stand runs; or PPE-controlled multi-stand runs, the Event Monitor will function properly, and output reported to a *.cp2* file will be readable by the FVS-EMAP project.

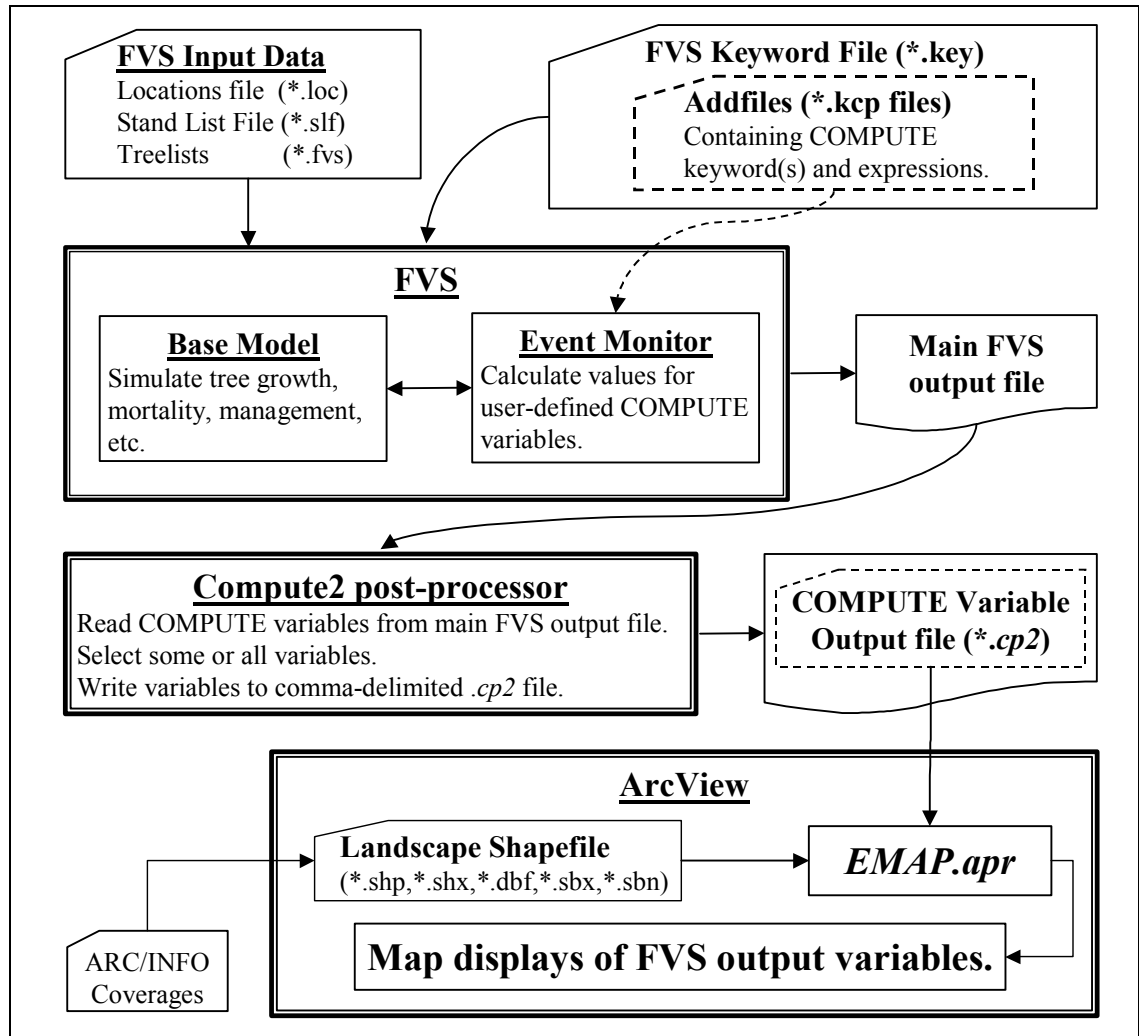


Figure 1. Flowchart diagramming the flow of information from FVS and the Event Monitor into the FVS-EMAP ArcView® project *EMAP.apr*.

Using the Event Monitor to Calculate Desired Output Variables

Introduction

Using the Event Monitor to make FVS results available to ArcView® does not require extensive knowledge of this powerful FVS application. In this section, we describe some of the essential features as they apply to the FVS-EMAP. Complete documentation of the Event Monitor is contained in the *User's Guide to the Event Monitor: Part of the Prognosis Model Version 6* (Crookston 1990), and in *Essential FVS: A User's Guide to the Forest Vegetation Simulator* (Dixon, 2002).

The Event Monitor is a sub-program within FVS that allows the user to schedule “events” (e.g., simulate management activities, such as a harvest) based on the value of “monitored” (evaluated) variable(s). (As a simple example, a thinning operation could be scheduled to occur whenever the stand basal area exceeds some threshold value.) What the Event Monitor provides to the FVS-EMAP is the ability to **compute the value** of an almost infinite variety of possible **user-defined variables** via the use of the Event Monitor's COMPUTE keyword. Values of the COMPUTED variables are reported to the main FVS output file. Thus, for the FVS-EMAP, the Event Monitor is being used to produce output data² representing the value of desired variables at one or all points of simulated time³. These COMPUTE-variable output data—after being processed by the “Compute2” post-processor—are used as input into the FVS-EMAP ArcView® project. The following section describes constructing COMPUTE statements.

Creating an Event Monitor Addfile

A large number of site, stand, and tree variables are available in the Event Monitor. [For a complete listing of available variables, users should refer to the *Essential FVS* (Dixon 2002)]. The COMPUTE keyword allows users to mathematically manipulate any of these recognized variables, thereby calculating new user-defined variables. **It is these new user-defined variables—calculated via the COMPUTE keyword—that are written to an output file (.cp2), which is read-able by this ArcView® project (EMAP.apr).**

Accordingly, a user may set-up an FVS simulation so that specific variables of interest are written to a .cp2 output file. (These “variables of interest” are the ones that a user wants to spatially portray via the *EMAP.apr*.) This is performed by constructing a text file containing the FVS/Event Monitor keyword COMPUTE followed by any number of mathematical expressions containing recognized variable names on the right-hand side of each expression and a user-defined variable name on the left-hand side of

² Note: although we describe using the Event Monitor only to provide COMPUTE variable output and not to schedule management activities, this does not preclude a user from using the Event Monitor to do both at the same time.

³ In FVS, the Event Monitor is invoked only in FVS cycle-beginning years. Thus, values for variables in the Event Monitor output file are reported only for FVS cycle-boundary years. Furthermore, COMPUTE variable output may be requested for all cycle-beginning years, or for one specific cycle-beginning year.

each expression. The expressions must be followed by the END keyword. This file should be saved as an ASCII text file, with a filename extension “.kcp.” It may then be added to any or all stands in an FVS simulation as an addfile—i.e., as an external file⁴ to be read-in and applied to all stands or to a specific (user-defined) subset of stands in the simulation (see the next section).

For example, one of the recognized variables in the Event Monitor is **BBA**, which is the before-thin (total) basal area in the stand (in square feet per acre). If the user wanted this variable written to the .cp2 file, an addfile (.kcp file) that would accomplish this would read:

```
COMPUTE          0
B_Area = BBA
END
```

The zero in the COMPUTE keyword's field 1 tells FVS to compute this variable for every projected growth cycle⁵, and “**B_Area**” is the new user-defined variable name⁶ whose value will be reported in the .cp2 output file.

If the user wanted this same variable, but converted to square meters per hectare, then the addfile would read:

```
COMPUTE          0
MetricBA = BBA * 4.356
END
```

Besides having the Event Monitor simply return values for pre-defined Event Monitor variables to the .cp2 file, a user may construct elaborate calculations involving a number of conditions (evaluated with an “IF...THEN...COMPUTE” logic) and/or a number of variables. For example, Amman et al. (1977) designed a hazard rating system for mountain pine beetle (*Dendroctonus ponderosae* Hopkins) in lodgepole line (*Pinus contorta* var. *latifolia* Engelmann). The Amman system uses three stand attributes: elevation, age, and average diameter at breast height (DBH) of host trees greater than 5 inches DBH. Each stand is assigned a rating of 1, 2, or 3 for each of these factors; the three factors are then multiplied together, resulting in a final risk rating score ranging from 1 to 27. By using the Event Monitor to calculate these scores, users can generate a .cp2 file containing the values of these scores, thus permitting the subsequent mapping (via the FVS-EMAP project) of stand hazard ratings over simulated time. An Event Monitor addfile, which calculates these Amman system ratings, is provided in the Appendix.

⁴ Alternatively, the COMPUTE keyword and expressions may be directly inserted into the keyword file, rather than being constructed as a separate addfile. The utilization of addfiles, however, represents an efficient way to perform, save, and share Event Monitor applications.

⁵ The COMPUTE keyword has only one parameter field. The value entered in this field is the cycle (or year) in which the following expressions are to be evaluated. Enter “0” if you want the expressions evaluated in all cycles. Default = 1.

⁶ User-defined variable names must be no more than eight characters long.

Additionally, the Event Monitor contains many *functions* that can calculate values of variables for a specific subset of trees. One such function is the SPMCDDBH function (Table 1). This very powerful function can return up to 11 different variables for any (or all) tree species, for any (or all) tree-value class(es), for any tree status class (live, dead, or harvested), within any user-defined height and/or DBH ranges. Use of this function is demonstrated in the examples in the appendix. The eight arguments of the function are outlined in Table 1, below. Table 2 describes the 11 possible variables users may request (via the function's first argument).

Table 1. Descriptions of the eight arguments of the Event Monitor function SPMCDDBH.

Argument	Description	Default value
1	Numeric code representing the output variable desired (number 1-11. See Table 4.)	None
2	Tree species code (enter zero to sum over all species). Tree species codes are FVS-variant specific.	None
3	Tree value class (IMC codes 1, 2, or 3). Enter zero to sum over all value classes.	None
4	Lower diameter limit (DBH in inches)	0
5	Upper diameter limit (DBH in inches)	999
6	Lower height limit (feet)	0
7	Upper height limit (feet)	999
8	Tree status (0=live trees; 1=recent mortality; 2=harvested)	0

Table 2. Description of the output variables available via SPMCDDBH, and referenced via a numeric code in the function's first argument.

Argument 1 Numeric Code	Variable
1	Trees per acre
2	Basal area per acre
3	Cubic foot volume per acre
4	Board foot volume per acre
5	Quadratic mean diameter
6	Average height
7	Percent cover
8	Average dwarf mistletoe rating
9	Merchantable cubic foot volume per acre
10	Average diameter growth (per cycle)
11	Stand density index

For example, to extract the board foot volume per acre for Douglas-fir⁷ greater than 6 inches DBH over all tree value classes for trees between 30 and 60 feet tall that are currently alive in the stand, the function would read:

```
COMPUTE          0
Dfvolgt6 = SPMCDBH(4,DF,0,6,999,30,60,0)
END
```

Users should be aware that the Event Monitor “schedules” the COMPUTE statements such that the non-conditional (“straight”) COMPUTEs are done first, then the conditional COMPUTEs are done next (if the ‘IF’ condition evaluates to “TRUE”). The evaluation of the conditional COMPUTEs is done iteratively so that, after the program has calculated one set of conditional COMPUTEs (those that were “TRUE”), it will re-evaluated the remaining conditional statements to see if any others (that were previously “FALSE”) are now “TRUE” as a result of the most recent computation. This will be re-iterated until no more COMPUTEs are scheduled for this cycle.

Including an Addfile to an FVS Simulation

The easiest way to add a *.kcp* file to the simulation (keyword) file is to include it from within Suppose by selecting ‘Edit Keyword File’ (from the main Suppose ‘Selections’ window) and then click the ‘Insert from file’ button. Browse to the directory on your computer where the file is located (it may reside anywhere), select the file, then select ‘Open’. The file will be added to all of the stands belonging to the *current group*⁸ (if the ‘Current Group’ radio button is selected in the ‘Selections’ window), or to the *current stand* (if the ‘Current Stand’ radio button is active in the Selections window). Then close the ‘Edit Simulation File’ window.

For example, if you want to include the addfile to *all* stands in a multi-stand simulation, be sure that a group that contains all stands (e.g., group ‘All’) is the current group, and that the ‘Current Group’ radio button is selected (as opposed to the ‘Current Stand’ radio button) in the main Suppose Selections window, *before* you choose the ‘Insert from file’ button.

Requesting a .cp2 Output File

Including an Event Monitor addfile containing the COMPUTE keyword to an FVS simulation file will result in *all* COMPUTE variables being written to the *main* FVS output file (for the cycles indicated in the COMPUTE keyword’s field 1). In order to generate a *.cp2* file (the file needed by the FVS-EMAP) you must select the post-processor that will create the file.

⁷ Species codes may be either alphabetic or variant-specific numeric.

⁸ The “current group” (or “current stand”) is the group (or stand) that appears in the window above the ‘Change Group Membership’ button. Open the window’s pulldown menu to change the current group or stand.

To invoke the writing of the *.cp2* file:

- Select (from the main Suppose 'Selections' window) '**Select Post Processors**';
- Scroll down to, and select: '**Compute2 – Table of Concatenated Compute Variables**' (comma delimited);
- Select '**Include**'.

By default, the Compute2 post-processor writes *all* COMPUTE variables to the *.cp2* file. However, when the post-processor is launched, it presents to the user a new interface, wholly separate from Suppose and FVS, that allows the user to edit this *.cp2* file. The user may choose which of the COMPUTE variables to include in the *.cp2* file.⁹

Summary

The COMPUTE keyword may be used to assign values to user-defined variable names. Assignment of values to user-defined variable names can be done in a number of ways:

1. The user-defined variable name can be assigned a value of a pre-defined Event Monitor variable, *or* a mathematical modification of one or more Event Monitor variables (*or any other previously defined variable*), for example:

```
COMPUTE          0
Varibl_1 = BBA
Varibl_2 = Varibl_1 * 4.356
Varibl_310 = BTOPHT/SITE * 100
END
```

2. The user-defined variable name can be assigned *conditionally*: that is, depending upon the value of some other variable; for example:

```
IF
BBA GT 160
THEN
COMPUTE          0
SCORE = 3
END
ENDIF
```

3. The user-defined variable name may be assigned a value calculated using built-in Event Monitor functions, for example:

```
COMPUTE          0
OLD_DBH = SPMCDDBH(5,7,0) - SPMCDDBH(10,7,0)
END
```

Variables calculated via COMPUTE statements are written to a *.cp2* file by requesting the Compute2 post-processor in Suppose. This *.cp2* file is readable by the FVS-EMAP ArcView[®] project (*EMAP.apr*), described in the next section.

⁹ If desired, the user may use this interface to create *any number* of separate *.cp2* files—each derived from the same FVS output file, and each containing its own suite of COMPUTE variables.

¹⁰ In this example, BTOPHT and SITE are pre-defined Event Monitor variables.

The FVS-EMAP ArcView® Project: EMAP.apr

Overview

The FVS-EMAP ArcView® project is essentially a “blank” ArcView® project. It differs from a “traditional” blank project in that it contains three additional scripts—written in ArcView’s Avenue® programming language—which have been compiled and embedded into the project. Each of these scripts performs a specific action or “function”.

These functions are accessed via buttons on the button bar (Table 1). One function is available when the ‘Project’ window is active; the other two are available when a View is active. The function that loads a .cp2 file (Function 1) is also available via the “Project” pulldown menu. Table 4 outlines what each function does.

Table 3. Descriptions of the three custom functions of the FVS-EMAP ArcView® project (EMAP.apr) and their accessibility.




Function	What it does	Available when...	Button
1	Loads .cp2 file output variables	a Project window is active	
2	Animates the display	a View window is active	
3	Creates an AREALOCs supplemental record file	a View window is active	

Table 4. Descriptions of the three custom functions of the ArcView® project EMAP.apr.

Function	What the function does:
1	Prompts the user to locate a .cp2 file—an FVS output file containing values of all Event Monitor COMPUTE variables (reported for FVS cycle-beginning years requested by the user)—then prompts the user to locate a shapefile, and then joins the .cp2 file output to the shapefile.
2	“Animates” the active View by displaying each Theme in the View one-at-a-time in two-second intervals, from top to bottom (which will be in chronological order if the Themes are positioned by Function 1).
3	Creates a supplemental record file for PPE keyword AREALOCs. The function: 1) extracts polygon areas, and x- and y-coordinates of polygon centers, and 2) launches an external program (<i>arealocs.exe</i>), which will rewrite the extracted information into a text file appropriately formatted for FVS.

Each function operates independently of the others. Function 1 requires merely that a .cp2 file and a shapefile reside on the users computer. Function 2 requires that some Themes exist in the active View. Function 3 requires that at least one Theme has been loaded into a View, and that the executable program *arealocs.exe* resides in the same

directory (the **project directory**¹¹) as does this project: *EMAP.apr*. The following sections describe the three functions in detail.

Function 1: Joining Event Monitor Output to a Shapefile in ArcView®

Event Monitor COMPUTE variable output, written to a comma-delimited text file (a *.cp2* file), may be read into an ArcView® project and joined to a shapefile via Function 1 of this customized ArcView® project. By performing a few simple point-and-click steps (in ArcView®, after running an FVS simulation), a user can quickly create time-series map displays (Views) in ArcView®, displaying FVS COMPUTE variables. Step-by-step instructions for using the Function follow. Instructions below assume that the user has already made an FVS run that included the COMPUTE keyword, and has invoked the post-processor that creates the *.cp2* file—see the section: *Requesting a .cp2 Output File*.

1. Launch ArcView® and open the project *EMAP.apr*.
2. Select the “join” button from the button bar (see Table 1) or select ‘Load *.cp2 file’ from the Project menu. (The “join” button that loads the *.cp2 file is available only when the Project window is active.)
3. You will be prompted to locate the *.cp2* file that contains the FVS output variables—generated via the COMPUTE expression(s)—you want to map.
4. After selecting the *.cp2* file, you will be prompted to locate a shapefile. The shapefile’s database table (a *.dbf* file)¹² must contain a field that contains stand identification codes **identical to** the stand identification codes used in the FVS simulation (keyword STDIDENT). (See discussion following these ten steps.)
5. You will then be prompted to select one variable you want mapped¹³. The list of displayed variables will contain all variables reported in the *.cp2* file.
6. After you have selected a variable to map, the project will re-write the data from the *.cp2* file into a number of new *.dbf* files, which the project will put into the subdirectory ‘SupportFiles’. These new *.dbf* are the files that the project will actually use when it performs the “joins”. There will be one new *.dbf* file for each year reported in the *.cp2* file. (See the following section on *Support Files*.)
7. Next, the project will prompt you to select the field from the shapefile’s *.dbf* containing the stand identification codes. (See footnote 14.)

¹¹ The “project directory” is the directory on your computer in which this project (*EMAP.apr*) resides.

¹² The shapefile’s attribute (*.dbf) table may be viewed by selecting ‘Theme → Table’ from the project Menu (after the shapefile has been added to a View). Alternatively, it may be opened outside of ArcView® in a spreadsheet software (e.g. Microsoft Excel).

¹³ Although you may only select one variable to portray in the map View, you may create many Views within a project, with each View representing its own output variable (from any *.cp2* file).

8. Next, the program will search for an existing legend (in a 'Legends' subdirectory). If the project finds one or more legends corresponding (in name) to the variable name, the program will prompt the user to select one. If a matching legend is not found, the program will create a legend. See the next section for further details.
9. The project will then create a map View portraying the selected variable on the landscape. The View will contain one Theme (a map "layer") for each year in the *.cp2* file, plus one Theme (named "landscape") portraying the outlines (boundaries) of the polygons in the shapefile.
10. Be sure to save your new project with a new name (via 'File→Save as...').

The "join" performed by the project is done "virtually"; that is, the shapefile's original **.dbf* table is not modified (a permanent join will not be performed). A new "virtual" table is created (for each Theme/layer) by joining the shapefile's *.dbf* with data from the newly-created *.dbf* files that the project creates from your *.cp2* file (see step 6).

As mentioned in step 4, the stand identification codes in the shapefile must be identical to the stand identification codes in the FVS simulation. However, it does not matter if the *type* of those codes differs (e.g. *character* type versus *numeric* type).

For example, stand identification codes in the project-created *.dbf* files (step 6) are *always* written as type **character** (even if they are comprised only of digits). In the shapefile's *.dbf* file, however, the stand identification code field (step 7), while likely formatted as type character, might be formatted as type **numeric**. Regardless of the type of the stand identification codes in the shapefile, the project will successfully join¹⁴ the data from the *.cp2* file with the shapefile for all records where the stand identification codes *exactly* match. No blank spaces are allowed in the codes.

The order of the stand identification codes—in either the *.cp2* file or the shapefile's *.dbf* table—does not matter.

Also, it is okay if there are polygons (stands) in the shapefile that do not exist in the FVS simulation output. Those polygons will simply not be "populated" with output variable data. Further, it is okay if there are stands simulated in FVS that are not in the shapefile. If this is the case, the project will notify the user that the *.cp2* file data for such stands could not be joined to the shapefile. The project will offer to write a report listing those stands.

¹⁴ If the selected stand ID code field from the shapefile *.dbf* (see step 7) is of a *numeric* type, the project will write a new field into the shapefile *.dbf*. This new field will be identical to the original chosen (numeric) field, except that it will be of type *character*. It will be given a name that is the original fieldname, preceded by a 'C' (for 'character'). This new field will be the one upon which the join will be based. If the join fails (e.g. no stand identification codes matched between the shapefile *.dbf* and the FVS/*.cp2* *.dbf*'s) then the program will remove this new field, thus returning the shapefile to its original state. Program prompts will inform the user anytime the shapefile *.dbf* is going to be edited. The program also will allow the user to abort the procedure.

Legends

The legend creation process in FVS-EMAP works as follows.

After the variable to be mapped is selected and the join performed, the project will look for a pre-formed legend in a 'Legends' subdirectory, which must exist immediately subordinate to the project directory. (See the section *Directory Structure*.) The project will search for *all* legends whose filename begins with the same string of characters as the name of the variable to be mapped. If the project finds *one* match, it will ask the user if s/he wants to use that legend. If the user selects 'No', the project will create a new legend (see next paragraph). If *more than one* legend exists that has a name beginning with the variable-name string, then the user will be prompted to choose from a list of matching legends. If the user does not want to use a pre-formed legend, s/he may 'Cancel' at this point, and let the program create a new legend.

If a pre-formed legend is not used, the project will create a new legend. When the project enters this mode, it prompts the user for which type of legend to create: a 'Unique Values' legend, or a 'Graduated' legend.

'Unique Values' legends should be used for discrete (or categorical) variables—variables that take on a finite (and relatively small) number of specific values, such as a hazard rating score or size class. Conversely, a 'Graduated' legend should be used for mapping continuous variables—variables that may take on a large (or possibly infinite) number of values, such as basal area.

If a 'Unique Values' legend is chosen, then each unique value in each Theme will be portrayed by a separate legend entry. The project will prompt the user to select a color scheme for the legend. If the chosen legend color scheme contains fewer colors than there are values in the data being mapped, then colors in the legend will be repeated (i.e. one color will represent more than one value). A default color scheme is provided should the user not select one.

If a 'Graduated' legend is chosen, the project will group the data into categories. By default, the project will group the data into five categories, though the project gives the user the option to change the number of categories (to anywhere between three and seven). The values bounding each category will be determined in one of two ways, depending upon how the user chooses to have the project create the categories.

The 'Graduated' legend categories will be designed based on either: (1) *natural breaks*¹⁵ in the data, in which case the values bounding each category are based on inherent "groupings" in each Theme's data, or: (2) *equal intervals*¹⁵ between categories, with categories being determined by evaluating the dataset's entire range of values—across *all* Themes. The user controls, by responding to a program query, which type of legend will be created. If the natural breaks type of legend is chosen, then the values

¹⁵ Refer to ArcView's Help library for more information about ArcView's "natural breaks" and "equal interval" legend types.

bounding each legend category likely will vary from Theme-to-Theme (as determined by each Theme's natural "grouping" of its data). If, on the other hand, an equal interval legend is chosen, then values bounding each category will be *identical* from Theme to Theme. The natural breaks legends are useful for identifying and analyzing possible patterns in data that may exist *within one Theme*. The equal interval legends (probably more often preferred) facilitate interpretation when comparing one Theme to another.

Users opting for a 'Graduated' legend will also be given a choice of color ramps to use for the legend. A default color ramp is provided should the user not select one.

Important note: New legends created by the user—or by the project—(i.e. any legend that is not a 'pre-formed' legend) will *not* automatically be saved into the Legends subdirectory. Users wanting to save a legend may do so via the 'Save' button in ArcView's Legend Editor[®]. If the user provides a filename for this new legend that begins with the same character string as the variable name that it represents, and places it in the 'Legends' subdirectory, then the next time the project maps this variable (from *any* .cp2 file containing this variable name) the legend will be available to be automatically applied. (See discussion above.)

Support Files

As mentioned in Step 6 (above), the project creates a series of new .dbf files each time a variable is read-in from a .cp2 file. ***There will be one new .dbf file created for each year in the .cp2 file***, each one representing one year's worth of data from the chosen .cp2 file. Each .dbf file will contain a column containing the stand identification codes, a column for the year represented in the file, and one column for each output variable in the .cp2 file. There will be one record (one row) in each .dbf file for each stand in the .cp2 file.

The filename given to the new .dbf files will be a concatenation of (1) the output variable chosen to be mapped, (2) the simulation year represented by the file, (3) the original simulation name, and (4) a unique date/time stamp¹⁶. An example follows.

Suppose a user runs an FVS simulation on a group of stands from the year 2000 until 2050, with a cycle length of 10years. Assume the user includes in the simulation a .kcp file that COMPUTEs a mountain pine beetle hazard rating—represented by the variable name of 'MPB_Haz'—and that the user invokes the Compute2 post processor, thereby generating a .cp2 file. The user names his simulation 'Scenario1'.

For Step 3 (above), the user would choose 'Scenario1.cp2'. For Step 5, the user chooses 'MPB_Haz'. For step 6, then, the project will write five new files to the SupportFiles directory:

¹⁶ The unique time stamp is the date in seconds. Although the time stamp may be the same for one set of .dbf files, it will always be different between sets, hence in this sense, the time stamp is unique. The time stamp is added to the filename so that a user may load into FVS-EMAP results from a subsequent simulation having the same name as an earlier simulation without worrying about overwriting an earlier set of .dbf files.

MPB_Haz2000_Scenario1on1023456789.dbf
MPB_Haz2010_Scenario1on1023456789.dbf
MPB_Haz2020_Scenario1on1023456789.dbf
MPB_Haz2030_Scenario1on1023456789.dbf
MPB_Haz2040_Scenario1on1023456789.dbf

For this example, these will be the new *.dbf* files by which the project will perform the join. They will be written to the SupportFiles subdirectory. The first part of each *.dbf*'s filename is the variable chosen to be mapped; the second part of the filename is the year represented in the *.dbf*; the third part is the simulation name; the last part (the long number) is the date in seconds. (Since these files are written very quickly, the set of *.dbf* files thus created may have the same time stamp.)

View and Theme Labels

The project will provide a title for the newly-created map View. The View's title will include the simulation name and the variable name. Using the example from the previous section, the View's title will read:

‘Scenario1 by the variable MBP_haz’.

Each Theme in this View will be given a name that is a concatenation (with a space in between) of the variable name and the year. Again, using the earlier example, the Theme representing the ‘MPB_Haz’ variable for the year 2010 would be labeled:

‘MPB_Haz 2010’

Function 2: Animating the Display

After creating a multi-themed View representing a series of time periods, you may select the ‘Animate Display’ button (see Table 3). The program will prompt you to verify that you want to continue. After you respond “Yes”, the program will display each Theme, one at a time (in two-second intervals) from top to bottom (which will be in chronological order if the Themes were placed by Function 1). This function may be used with *any* ArcView® map View, not just with Views created by Function 1.

Function 3: Creating an AREALOCS Supplemental Record File from an ArcView® Shapefile

The AREALOCS supplemental record file is a file needed by the Parallel Processor Extension (PPE) (Crookston and Stage 1991) keyword AREALOCS and contains data regarding the location and area of stands. It is not needed by users of non-PPE variants. Unlike the first two functions described in this document, which are essentially “post-processors” (that is, they manipulate model output), this third function is a “pre-processor”—it manipulates data to make it ready to go *into* an FVS simulation.

Specifically, the four fields of data in the supplemental record file are: StandID, x-coordinate of stand center, y-coordinate of stand center, and stand area. These data are not necessarily needed by the PPE, but may be needed depending upon how you intend to

use the PPE. If, for example, you want to run the Westwide Pine Beetle (WWPB) Model extension to FVS, keyword AREALOCS—along with its appropriately formatted list of supplemental records—is *required*. This function creates this supplemental record file.

The function built into this ArcView[®] project that creates the AREALOCS supplemental record file is accessed via the button with an “XY” on it (see Table 3). This button is available when a View window is active. At least one shapefile should be loaded into the View. (The loaded shapefile(s) should be the one(s) from which you desire supplemental record file(s) to be created.) The function requires that the Fortran program *arealocs.exe*¹⁷ resides in the project directory.

Specifically, this function:

1. extracts stand centroid information (x- and y-coordinates of stand [polygon] centers), and stand area from an ArcView[®] shapefile;
2. optionally writes this information into the shapefile's attribute table (feature table—i.e., the *.dbf* file associated with the shapefile);
3. writes and saves a separate *.txt* file containing only the four fields of data (comma delimited) needed for the AREALOCS supplemental records (i.e., standID, x-coordinate, y-coordinate, and area); then
4. launches a Fortran executable program that reads the newly-created *.txt* file and then writes the data (appropriately formatted) to a file (which can then be used in a PPE/FVS simulation as the AREALOCS supplemental record file).

To use this function, follow the following steps:

1. Verify that the program executable file—*arealocs.exe*—resides in the same directory as does the ArcView[®] project: *EMAP.apr*.
2. Launch the ArcView[®] project *EMAP.apr*.
3. Add a Theme to a View. If the added Theme is not already in a shapefile format, convert it to a shapefile (from the menubar, choose: ‘Theme→Convert to Shapefile’). This shapefile should represent the stands/polygons for which you desire centroid and area information for your AREALOCS supplemental record file.
4. You may load more than one Theme/shapefile. However, the program, when launched, will only operate on one Theme at a time.
5. Select the ‘XY’ button depicted in Table 3.
6. The program will prompt you to select a Theme (from a list of Themes loaded into the project). Select a Theme.

¹⁷ *arealocs.exe* is a separate stand-alone program—launched from within this ArcView project—that will read-in data provided by ArcView, and will then re-write that data so that they are FVS-ready.

7. The program will then ask you if you want the polygon/stand area and centroid information added to the Theme's attribute table. Answering "yes" or "no" will *not* affect the AREALOCS supplemental record file that will be created. If you answer "yes," then three new fields will be added to the shapefile's attribute table (i.e., the *.dbf* file associated with the shapefile)—one field/column each for the stand area and x- and y-coordinates of stand centers.
8. The program will then write a comma-delimited text file (named ***export_out.txt***) containing the four fields of required data: StandID, x-coordinate, y-coordinate, and area (in the same base units as in the originating shapefile).
9. The program will then launch the Fortran executable ***arealocs.exe***, which will create your supplemental record file. This program will launch a DOS window into which you will respond to messages.
10. First, you will need to respond to a query about the distance units of the shapefile data. This information is needed so that the program can write the supplemental record file in the appropriate units: distance in meters and area in acres. If you do not know the units, respond to the query with the "o" (for "other") units. You can correct for differing units later, within your FVS simulation via PPE keyword AREALOCS. If you know the shapefile's distance units, the program will convert distances to meters and areas to acres.
11. Then, the program will prompt you to provide a name for your output file (the file that will contain the supplemental records to be used with keyword AREALOCS in your FVS simulation). You may provide any name you like (no spaces), with any extension you like, or you may accept the default name: ***arealocs.alc***.
12. The program will quickly finish and will provide you with a brief message. Press <Enter> on the keyboard to complete the program execution. Two new files now exist in your project directory: *export_out.txt* (which you no longer need) and your *arealocs.alc* file (with whatever name you gave it in step 11).
13. If you subsequently re-run this program on another shapefile, you will need to do one of three things to ensure that the *arealocs.alc* file just written does not get overwritten. Either: 1) move the newly created file to another directory, or 2) rename the newly created file, or 3) give a different name to the next file to be created. In any case, re-running the program on a different shapefile from the same location will write over the *export_out.txt* file, but that should not matter—that file is no longer needed once the *arealocs.alc* file is written.

The program described will extract the desired information for *every* polygon in the shapefile. Having excess stand information in this file will likely *not* cause FVS/PPE simulation failures. For example, if you are using this AREALOCS supplemental record file for a WWPB Model simulation, then extraneous records in the AREALOCS supplemental record file will not present a problem. (Specifically, if there are records in the supplemental record file for stands that are not included in the FVS simulation, they will be ignored.) What *will* present a problem, however, is if there is *not* a record in the

AREALOCs supplemental record file for every stand to be simulated. Every stand to be simulated in a WWPB Model simulation needs to exist in the AREALOCs supplemental record file or else the model will fail to execute. Thus, users needing an AREALOCs supplemental record file for an FVS simulation should ensure that either: 1) the shapefile from which they are extracting information contains all of the stands to be simulated in FVS, or 2) if not, then this function/program needs to be run on another shapefile and the resulting supplemental record file (or at least the needed part(s) thereof) must be appended to the first.

Directory Structure

Downloading FVS-EMAP from the FHTET website (see the *Introduction* section) retrieves to the user's computer a self-extracting Zip file (extension ".exe"). Launching this file (from wherever you downloaded it to on your computer) will result in the creation of three new directories on your computer:

1. 'FVS_EMAP', which, by default, will be created immediately under your root directory (e.g. C:/FVS_EMAP);
2. a 'Legends' subdirectory (immediately subordinate to FVS_EMAP); and
3. a 'SupportFiles' subdirectory (also immediately subordinate to FVS_EMAP).

Into the 'FVS_EMAP' directory will be extracted the ArcView® project itself (*EMAP.apr*) as well as the *arealocs.exe* executable program. This directory thus becomes the 'project' directory.

The 'Legends' and 'SupportFiles' subdirectories are required for Function 1 to work. (They are not required by the other two Functions.) These two directories will initially be empty (upon download), but will be used by the project later—see below.

Although the *EMAP.apr* project will function properly no matter where it resides, ***we recommend that user's maintain the directory structure as created upon extraction (i.e. that the project EMAP.apr resides in a directory named FVS_EMAP which resides immediately under "C:/".*** Maintaining this directory structure: (1) ensures that files downloaded later (e.g. new legends) can efficiently be unzipped to their proper location; and (2) will facilitate project sharing between computers (see next section).

The 'Legends' subdirectory is the location from which the project accesses pre-formed legend files (*.avl files). The project "looks" for this directory when creating the map Views, regardless of whether or not any pre-formed legends exist or will be used. If the project cannot find the Legends subdirectory (e.g. if you deleted it), the project will issue a warning indicating that the Legends directory was not found¹⁸.

¹⁸ The project will, however, still join the FVS output data with the shapefile, and will still create a map View, however the map (and its legend) will be a 'single-symbol' map, consisting of one color.

Pre-formed legend files (*.avl) may come from a variety of sources including: (1) legends obtained by the user from external sources¹⁹; (2) legends created (and saved) by the user from this or other ArcView® project applications; or (3) legends previously created by *EMAP.apr* and saved by the user.

The 'SupportFiles' subdirectory will contain the *.dbf* files created by and needed by the project. (See the *Support Files* section.) It is these *.dbf* files that are joined to the shapefile, thus allowing the map Views to be created. One *.dbf* file will be used for each map Theme (or map "layer"). If this 'SupportFiles' directory gets moved, renamed, or deleted, then the project will fail when it attempts to join COMPUTE variable output to a shapefile (i.e. after Function #1's third dialog window).

Saving and Sharing Your Work

The *EMAP.apr* project is a "read only" project. It will not let you alter itself. After you have modified the project by adding data, you will need to use the "Save as..." command, thereby creating a new project with a new name (e.g., *Analysis_1.apr*).

ArcView® stores absolute pathnames for the shapefiles and *.dbf* files it needs to "re-assemble" your project each time you open it. Thus, the project is "looking for" your shapefile(s) and *.dbf* files in very specific locations on your computer. If you move any of these files or rename directories or if you share the files and the project with others, then the project will, by default, prompt you (or the new user) to locate *each file* it needs to rebuild *each Theme* in the project. This can be tedious if there are many Themes in the project. Users maintaining a common directory structure, however, (as discussed above) will find project sharing much easier, because the project will be able to find automatically the files it needs.

If users are sharing project-related files between two computers having dissimilar directory structures, users may edit their saved project file in a text editor²⁰. Users may perform a search for the "Path" strings, and then perform a "Find and Replace" operation, replacing all of the old (now obsolete) absolute path(s) with the new (currently valid) path(s).

Alternatively, if all project files are moved into *one* directory (a placement scheme you may likely employ if you are packaging up the project to send to someone else), you may replace all of the old *absolute* paths with "...", followed by the filename. This, in essence, makes the path a *relative* path instead of an absolute path. Therefore, anyone

¹⁹ For example, pre-formed legends are available along with many of the Event Monitor addfiles (*.kcp* files) available at the FHTET website. These legends—created by FHTET staff—represent specific COMPUTE variables from the addfiles. For example, packaged with the Mountain Pine Beetle Risk Rating Event Monitor(I) comes a legend file (*MPB_HAZ.avl*) portraying the variable representing the calculated risk rating: *MPB_HAZ*.

²⁰ The following discussion is not a necessary requirement for project sharing, it merely helps the user avoid dealing with all of the dialog boxes prompting you "Where is..." each file it needs.

opening this edited project should not encounter difficulties associated with the project being unable to locate files it needs.

For example, suppose when you initially build your project, your shapefile resides in:

c:/GIS/shapefiles/

If you want to send your project to someone else, you could copy all of the pertinent files to one place. For example, into one place (e.g. c:/temp) you could copy the following files:

```

Analysis_1.apr
landscape.shp
landscape.shx
landscape.sbx
landscape.sbn
landscape.dbf
MPB_Haz2000_Scenario1on1023456789.dbf
MPB_Haz2010_Scenario1on1023456789.dbf
MPB_Haz2020_Scenario1on1023456789.dbf
MPB_Haz2030_Scenario1on1023456789.dbf
MPB_Haz2040_Scenario1on1023456789.dbf

```

(These, collectively, comprise the shapefile.)

Then, in a text editor, open this copy of *Project_Run1.apr*, and using a “Find and Replace function, replace every instance of:

“Path c:/FVS_EMAP/SupportFiles/*filename*”

and “Path c:/GIS/shapefiles/*filename*”

(where the word “Path” begins every line in which a pathname is identified, and where “*filename*” would be one of the shapefiles or *.dbf* files in the list above)

with “Path *../filename*” (by replacing the actual paths with “*../*”).

Then, save this edited project (from within the text editor). Now, the next time the project is opened, ArcView® will look for all of the pertinent files from within the directory in which it (the project itself) resides, *no matter where the project resides*. Hence, the project is now portable to any computer running ArcView® GIS version 3.2 or higher.

Important Note: Once this “amended” project is opened and *saved* again (from within ArcView®), ArcView® will “re-map” new absolute paths to these files. Therefore, if the data files and the project file are eventually going to be re-located again, this procedure is not extremely useful. This procedure *is* very useful, however, to get your project(s) and data from one computer to another in a way that will allow the project to quickly and efficiently access the needed data, thus facilitating project sharing.

Citations

Note: most of the FVS-related documents cited below are available online at:

<http://www.fs.fed.us/fmfc/fvs/documents/gtrs.php>

The FVS Home Page is at:

<http://www.fs.fed.us/fmfc/fvs/>

- Amman, Gene D., Mark D. McGregor, Donn B. Cahill, and William H. Klein. 1977. Guidelines for reducing losses of lodgepole pine to the mountain pine beetle in unmanaged stands in the Rocky Mountains. USDA Forest Service Gen. Tech. Rep. INT-36. Intermountain Forest and Range Experiment Station, Ogden, UT. 22 pp.
- Crookston, Nicholas L. 1990. User's guide to the Event Monitor: Part of Prognosis Model Version 6. USDA Forest Service Gen. Tech. Rep. INT-275 Intermountain Research Station, Ogden, UT. 27 pp. [Electronic Version (modified Sept. 2001) available online: http://www.fs.fed.us/fmfc/fvs/documents/gtrs_event_monitor.php (last accessed 11/01)]
- Crookston, N.L. and A.R. Stage. 1991. User's guide to the Parallel Processing Extension of the Prognosis Model. USDA Forest Service Gen. Tech. Rep. INT-281. Intermountain Research Station, Ogden, Utah. 88 pp.
- Dixon, Gary E., comp. 2002. Essential FVS: A user's guide to the Forest Vegetation Simulator. Internal Report, Fort Collins, CO: U.S. Dept. Agriculture, Forest Service, Forest Management Service Center. 183 pp.
- Schmid, J.M. and R.H. Frye. 1976. Stand ratings for spruce beetles. USDA Forest Service Res. Note RM-309. Rocky Mountain Forest and Range Experiment Station, Fort Collins, CO. 4 pp.
- Van Dyck, Michael G. 2001a. Keyword reference for the Forest Vegetation Simulator. USDA Forest Service, Forest Management Service Center, Fort Collins, CO. 106 pp.
- Van Dyck, Michael G. 2001b. User's guide to the post processors for the Forest Vegetation Simulator. USDA Forest Service, Forest Management Service Center, Fort Collins, CO. 27pp.
- Wykoff, William R., Nicholas L Crookston, and Albert Stage. 1982. User's guide to the Stand Prognosis Model. USDA Forest Service, Gen. Tech. Rep. INT-133. Intermountain Forest and Range Experiment Station, Ogden, UT. 112 pp.

Appendix: Example Event Monitor Addfiles

Following are three example Event Monitor addfiles. All three are available at:

<http://www.fs.fed.us/foresthealth/technology/products.shtml>

The first (*Summary_stats.kcp*) produces basic FVS summary statistics.

The second (*mpb_lpp_ammann.kcp*) calculates stand-level bark beetle hazard ratings (for mountain pine beetle in lodgepole pine) based on Amman et al. (1977).

The third (*Spruce_Beetle_Hazard.kcp*) calculates stand-level spruce beetle hazard ratings based on Schmid and Frye (1976).

All examples have a zero in every COMPUTE's field 1; thus, all will generate output for every simulation cycle. If you want output written for only one FVS cycle (instead of for all of them), replace the zero with the year (or cycle) for which you desire output.

If you are accessing an electronic version of this document, then any of the example *.kcp* files presented below (or parts thereof) may be copied and pasted into a text file, and used as (or in) an FVS addfile. Since FVS reads data from specific columns in the keyword files, you must use a text editor and a font that will preserve column spacing. The monospaced Courier font, used in the examples below, preserves column spacing.

In order to get the output variables written to a *.cp2* output file, you must invoke the post-processor that will prepare the file. In Suppose:

- From the main 'Selections' window, choose 'Select Post Processors'.
- Select 'Compute2 - Table of Concatenated Compute Variables (comma delimited)'.
- Select 'Include'.
- Then 'Close' the 'Select Post Processors' window.

Summary_stats.kcp

! This Event Monitor Addfile returns standard FVS output variables.
!
! Note: Variable names ending "_B" are before-thin values. Variable names
! ending "_A" are after-thin values. Both are cycle-beginning values.
! See the Essential FVS (Dixon 2002) for definitions of variable names
! on the right hand side of each expression.

```
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!  
!!  
!! Prepared 11/01 AJ McMahan, INTECS International, Inc., for: !!  
!! Slightly modified 12/02, AJM !!  
!! !!  
!! Forest Health Technology Enterprise Team, USDA Forest Service, !!  
!! 2150 Centre Ave Bldg A Suite 331, Ft. Collins, CO 80526-1891 !!  
!! !!  
!! Contact: Eric Smith 970 295-5841 (email: elsmith@fs.fed.us) !!  
!! !!  
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
```

```
COMPUTE 0  
CYC_END = CENDYEAR  
STND_AGE = AGE
```

```
TPA_B = BTPA  
BA_B = BBA  
SDI_B = BSDI  
CCF_B = BCCF  
HT_B = BTOPHT  
QMD_B = BADBH
```

```
CUFTVOL = BTCUFT  
mrch_3FT = BMCUFT  
mrchBDFT = BBDFT
```

```
TPA_rem = RTPA  
BA_rem = BBA-ABA  
3FT_rem = RTCUFT  
merc_rem = RMCUFT  
BDFT_rem = RBDFT
```

```
TPA_A = ATPA  
BA_A = ABA  
SDI_A = ASDI  
CCF_A = ACCF  
HT_A = ATOPHT  
QMD_A = AADBH
```

```
cycLNGTH = (CENDYEAR-YEAR)+1  
ACCRETN = ACC  
MORT_VOL = MORT  
MAI_m3FT = MAI  
WEIGHT = SAMPWT
```

END

mpb_lpp_amman.kcp

```

! THIS EVENT MONITOR CALCULATES A RISK RATING for MPB in LPP BASED ON THE
! SYSTEM OF AMMAN, et al. (1977)
! [Amman, McGregor, Cahill & Klein. 1977.  Guidelines for reducing losses
! of LPP to the MPB in unmanaged stands in the Rocky Mountains.
! USDA FS GTR INT-36, Intermountain F&R Exp Sta., Ogden, UT 22pp.]
!
! RISK IS BASED ON THREE FACTORS:
!
!     STAND ELEVATION,
!     STAND AGE,
!     AND HOST QMD (OF HOST TREES >5 INCHES DBH)
!
! EACH FACTOR IS ASSIGNED A VALUE (1,2, OR 3) BASED ON STAND CONDITIONS.
! RISK RATING IS BASED UPON THE PRODUCT OF THE THREE FACTORS:
!
!     ELEVATION      AGE      DBH      FACTOR VALUE
!     >10400         <60     <7       1
!     9400-10400     60-80   7-8      2
!     <9400          >80     >8       3
!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
! (Note: elevation breakpoints (E_THOLDS) vary with latitude.           !
! Those noted in table above are for ~ 39 deg. N latitude.             !
! See the "INITIALIZE" section below to set your E_THOLDS.)           !
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!
! RISK RATING: FACTOR1 * FACTOR2 * FACTOR3
!
! (Possible values: 1, 2, 3, 4, 6, 8, 9, 12, 18, 27)
!
! 1-9 = LOW; 12-18 = MODERATE; 27 = HIGH
!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!
!! Prepared 07/01 AJ McMahan, INTECS International, Inc., for:         !!
!!
!! Forest Health Technology Enterprise Team, USDA Forest Service,     !!
!! 2150 Centre Ave Bldg A Suite 331, Ft. Collins, CO 80526-1891      !!
!!
!! Contact: Eric Smith 970 295-5841 (email: elsmith@fs.fed.us)      !!
!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!! INITIALIZE !!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!

COMPUTE          0
AMMAN1 = -1
AMMAN2 = -1
AMMAN3 = -1
QMD_GT_5 = SPMCDBH(5,LP,0,5,999)
END

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!
!!                               DEFINE E_THOLDS
!!
!! ADD asterisk(s) (*) or exclamation points (!) in column 1 to
!! "comment out" the line.  (If "*" or "!", the line will be printed

```

```
!! [as a comment] in output; if "!!", line will not be printed in output.)
!!
!! REMOVE asterisks/exclamation points to "activate" the line.
!!
!! The following blocks define E_THOLDS for various latitudes.
!! "Activate" the applicable pair for your latitude,
!! or enter your own E_THOLDS.
!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
```

```
COMPUTE 0
```

```
!! For ~33 degrees N Latitude
```

```
!!E_THOLD1 = 13070
!!E_THOLD2 = 12070
```

```
!! For ~35 degrees N Latitude
```

```
!!E_THOLD1 = 12180
!!E_THOLD2 = 11180
```

```
!! For ~37 degrees N Latitude
```

```
!!E_THOLD1 = 11290
!!E_THOLD2 = 10290
```

```
!! For ~39 degrees N Latitude
```

```
E_THOLD1 = 10400
E_THOLD2 = 9400
```

```
!! For ~41 degrees N Latitude
```

```
!!E_THOLD1 = 9510
!!E_THOLD2 = 8510
```

```
!! For ~43 degrees N Latitude
```

```
!!E_THOLD1 = 8620
!!E_THOLD2 = 7620
```

```
!! For ~45 degrees N Latitude
```

```
!!E_THOLD1 = 7740
!!E_THOLD2 = 6740
```

```
!! For ~47 degrees N Latitude
```

```
!!E_THOLD1 = 6850
!!E_THOLD2 = 5850
```

```
!! For ~49 degrees N Latitude
```

```
!!E_THOLD1 = 5960
!!E_THOLD2 = 4960
```

```
!! OTHER--User Defined
```

```
!!E_THOLD1 =
!!E_THOLD2 =
```

```
END
```

```
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!                               END OF DEFINING E_THOLDs                    !!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
```

```
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!! FACTOR 1 (ELEVATION) !!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
```

```
IF
ELEV GT E_THOLD1
THEN
COMPUTE                0
AMMAN1 = 1
END
ENDIF
```

```
IF
ELEV GE E_THOLD2 AND ELEV LE E_THOLD1
THEN
COMPUTE                0
AMMAN1 = 2
END
ENDIF
```

```
IF
ELEV LT E_THOLD2
THEN
COMPUTE                0
AMMAN1 = 3
END
ENDIF
```

```
!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!! FACTOR 2 (AGE) !!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!
```

```
IF
AGE LT 60
THEN
COMPUTE                0
AMMAN2 = 1
END
ENDIF
```

```
IF
AGE GE 60 AND AGE LE 80
THEN
COMPUTE                0
AMMAN2 = 2
END
ENDIF
```

```
IF
AGE GT 80
THEN
```

```
COMPUTE          0
AMMAN2 = 3
END
ENDIF
```

```
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!! FACTOR THREE (QMD OF HOST > 5") !!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
```

```
IF
QMD_GT_5 LT 7.0
THEN
COMPUTE          0
AMMAN3 = 1
END
ENDIF
```

```
IF
QMD_GT_5 GE 7.0 AND QMD_GT_5 LE 8.0
THEN
COMPUTE          0
AMMAN3 = 2
END
ENDIF
```

```
IF
QMD_GT_5 GT 8.0
THEN
COMPUTE          0
AMMAN3 = 3
END
ENDIF
```

```
!!!!!!!!!!!!!!!!!!!!!!!!!!
!! RISK RATING !!
!!!!!!!!!!!!!!!!!!!!!!!!!!
```

```
IF
AMMAN1 GT 0 AND AMMAN2 GT 0 AND AMMAN3 GT 0
THEN
COMPUTE          0
AMM_HAZ = AMMAN1 * AMMAN2 * AMMAN3
END
ENDIF
```

```
*****
```

Spruce_Beetle_Hazard.kcp

```

! Event Monitor Program to calculate Spruce Beetle Risk Rating
! Based upon Schmid and Frye 1976
! (Stand Ratings for Spruce Beetles.  RMF&RES, USDA FS, Research Note RM309)
!
! Physiographic location is one of the risk rating system's criteria.  This
! location information is not available in the Event Monitor.  Thus, users
! will need to identify these stands and then place them into their own
! "groups" in FVS/Suppose and attach the appropriate Event Monitor addfile
! to the appropriate groups.
!
! This Event Monitor file is for use with non-creek bottom stands.
! A separate Event Monitor file should be used for creek bottom stands.
!
! The system:
!
! RISK FACTOR
!   VALUE   PHYSIOGRAPHIC LOCATION   DBH(of trees>10")   STAND BA   %SPRUCE
!     3      Creek bottoms             >16"              >150      >65%
!     2      SI(100)>= 80                >12-16"          >100-150  >50-65%
!     1      SI(100) < 80                10-12"           <=100    <=50%
!
! Total risk is then determined by summing the risk factor values for the 4
! criteria.
!
! Note: SI in table above is on 100-year basis.  Some FVS variants (e.g., UT)
! use 50-year SIs.
! THIS EVENT MONITOR USES 50-YR SIs!  Threshold of SI=80 (100 yr) translates
! to SI=52 at 50 yrs (From Alexander site index curves for Englemann spruce,
! USDA FS Res. Pap. RM-32, 1967.)
! *****
!
! This Event Monitor written 11/01 by A J McMahan (INTECS International, Inc.)
! for the Forest Health Technology Enterprise Team, USDA Forest Service,
! 2150 Centre Ave., Bldg. A, Ft. Collins, CO 80526-1891.
!
! Contact:  Eric Smith      970 295-5841 (email:  elsmith@fs.fed.us)
! *****
!
! Variable Names:
!
! SB_HAZRD = Spruce Beetle Hazard Rating
!   Note: The Spruce Beetle Hazard Rating equals the sum of the 4 component
!   rating factors (SBF1, SBF2, SBF3, SBF4), ** UNLESS ** : (1) there is no
!   spruce in the stand with a dbh >= 10" (in which case the rating will be
!   "1"); or (2) less than 5% of stand BA is spruce, in which case the
!   overall rating is zero.
!
! QMD_SP10 = QMD OF SPRUCE GREATER THAN (or =) 10"
! TOT_BA   = TOTAL STAND BA
! PER_SPRC = PERCENT OF STND_BA THAT IS SPRUCE (of all trees >2.6"dbh)
! SITE_NDX = STAND SITE INDEX FOR THE SITE SPECIES
!   (Note that the default SITE species for FVS variants may not be
!   Englemann spruce, hence if ES is not entered as the site species in the
!   FVS keyword file, then the site index variable available to the Event
!   Monitor may not be the SI for ES!
! *****

```

```
!!!!!!!!!!!!!!!!!!!!!!  
!!! INITIALIZE !!!  
!!!!!!!!!!!!!!!!!!!!!!
```

```
COMPUTE          0  
SBF1=-1  
SBF2=-1  
SBF3=-1  
SBF4=-1  
SB_HAZRD=-1
```

```
QMD_SP10 = SPMCDBH(5,8,0,10,999)  
SITE_NDX = SITE  
TOT_BA   = SPMCDBH(2,0,0)  
PER_SPRC = SPMCDBH(2,8,0,2.6,999) / SPMCDBH(2,0,0,2.6,999)
```

END

```
!!!!!!!!!!!!!!!!!!!!!!
```

```
!!!!!!!!!!!!!!!!!!!!!!  
!!! FIRST FACTOR !!!  
!!!!!!!!!!!!!!!!!!!!!!
```

```
!  
! Assume this is NOT a creek bottom stand !  
! (If it is a creek bottom stand, then a different *.kcp file should be  
! attached, not this one!)  
!  
! Note: Site indices herein are on 50-yr basis.  
! Thus, this Event Monitor may be used with FVS variants EM, TT, and UT,  
! which use 50-yr SIs (at least for Englemann spruce!) These variants do  
! not necessarily use 50-yr SIs for all tree species!  
!  
! Note that these SIs differ from values used by Schmid and Frye, who  
! use 100-yr SIs.
```

```
IF  
SITE GE 52  
THEN  
COMPUTE          0  
SBF1=2  
END  
ENDIF
```

```
IF  
SITE LT 52  
THEN  
COMPUTE          0  
SBF1=1  
END  
ENDIF
```

```
!!!!!!!!!!!!!!!!!!!!!!  
!!! SECOND FACTOR !!!  
!!!!!!!!!!!!!!!!!!!!!!
```

```
IF  
QMD_SP10 GT 16  
THEN  
COMPUTE          0  
SBF2 = 3  
END  
ENDIF
```



```

IF
QMD_SP10 GT 12 AND QMD_SP10 LE 16
THEN
COMPUTE          0
SBF2 = 2
END
ENDIF

```

```

IF
QMD_SP10 LE 12 AND QMD_SP10 GE 10
THEN
COMPUTE          0
SBF2 = 1
END
ENDIF

```

```

IF
QMD_SP10 LT 10
THEN
COMPUTE          0
SBF2 = 0
END
ENDIF

```

```

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!  THIRD FACTOR  !!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

```

```

IF
TOT_BA GT 150
THEN
COMPUTE          0
SBF3 = 3
END
ENDIF

```

```

IF
TOT_BA GT 100 AND TOT_BA LE 150
THEN
COMPUTE          0
SBF3 = 2
END
ENDIF

```

```

IF
TOT_BA LE 100
THEN
COMPUTE          0
SBF3 = 1
END
ENDIF

```

```

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!  FOURTH FACTOR  !!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

```

```

IF
PER_SPRC GT 0.65
THEN
COMPUTE          0
SBF4 = 3
END
ENDIF

```

```
IF
PER_SPRC GT 0.50 AND PER_SPRC LE 0.65
THEN
COMPUTE          0
SBF4 = 2
END
ENDIF
```

```
IF
PER_SPRC LE 0.50 AND PER_SPRC GE 0.05
THEN
COMPUTE          0
SBF4 = 1
END
ENDIF
```

```
IF
PER_SPRC LT 0.05
THEN
COMPUTE
SBF4 = 0
END
ENDIF
```

```
!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!! RISK RATING  !!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!
```

```
IF
SBF1 NE -1 AND SBF2 NE -1 AND SBF3 NE -1 AND SBF4 NE -1 AND &
  SB_HAZRD EQ -1
THEN
COMPUTE          0
SB_HAZRD = SBF1 + SBF2 + SBF3 + SBF4
END
ENDIF
```

```
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!! CONDITION: IF SPRUCE GREATER THAN 10 INCHES  !!!
!!!           THEN RISK IS VERY LOW             !!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
```

```
IF
SB_HAZRD GT -1 AND SBF2 EQ 0 AND (NOT (SBF4 EQ 0))
THEN
COMPUTE
SB_HAZRD = 1
END
ENDIF
```

```
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!! CONDITION: IF SPRUCE BA IS VERY LOW  !!!
!!!           THEN RISK VALUE IS ZERO    !!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
```

```
IF
SB_HAZRD GT -1 AND SBF4 EQ 0
THEN
COMPUTE
SB_HAZRD = 0
END
ENDIF
```

