FIA BioSum: A User's Guide (DRAFT)

By

Larry Potts
Lesley Bross
Sara Loreno
Tina Mozelewski
Jeremy Fried
Sebastian Busby

21 March 2024

NOTE: This Users Guide is in DRAFT status. Changes will continue to be made until it is published as a PNW GTR.

This version is deficient or at least far from a final product in many ways, including inconsistency in formatting, notes and comments among co-authors as to refinements and clarifications that are needed, etc. Apologies for that.

Acknowledgements

We are indebted to many remarkable people who have contributed to the development of BioSum over the past 18 years. The original, pre-software BioSum team in the early 2000s consisted of Jamie Barbour, Roger Fight, Glenn Christensen, Guy Pinjuv, Dale Weyermann and Jeremy Fried. Larry Potts joined as lead programmer in 2004 as we set about building an analysis tool, and has continued in this role ever since. Analysts and colleagues Peter Daugherty, Keith Gilless, Regis Cassidy, Bruce Hartsough, Marlin Johnson, Don Vandendriesce, Nick Crookston, Elizabeth Reinhardt, John Szymoniak, Dennis Fergueson and Bentesh Sharma contributed ideas, prescription suggestions and improvements on analyses conducted in BioSum's first decade. Circa 2010 and beyond, Terrie Jain, Bill Stewart, Olaf Kuegler, Demetrios Gatziolis, Joshua Petitmermet, John Sessions, Rob Keefe, Conor Bell and Jonathan Sandquist contributed critical ideas that helped develop BioSum and analysts Sara Loreno, Carlin Starrs and Tina Mozelewski and programmers Lesley Bross and Tyler Sorg joined the software development team and moved the effort forward on many fronts. Soorya Bateman and Xioaping Zhao provided valuable assistance with production of an earlier version of the user's manual, and quality assurance testing.

We gratefully acknowledge funding support provided for the initial development of the framework by Bryce Stokes of the Washington Office and Roger Condit of the Western Forest Leadership Coalition, Dave Cleaves with the National Fire Plan, and the PNW Forest Inventory and Analysis Program. Development of this BioSum software would not have been possible without the National Fire Plan funding provided by Jamie Barbour of the PNW's Focused Science Delivery Program, Sue Willits of the PNW's Forest Inventory and Analysis Program, the California Energy Commission, Oregon Department of Forestry, the Joint Fire Sciences Program, the Pacific Northwest Research Station and Andrew Gray of the PNW Resource Monitoring and Assessment Program.

Foreword

Development of Bioregional Inventory Originated Simulation Under Management (BioSum), an analysis framework supported by field data collected by the USFS Forest Inventory and Analysis Program, began in 2001 with a mission of “forest biomass summarization” as the escalating cost of suppressing large wildland fires inspired a shift in thinking about managing forested landscapes to reduce wildfire hazard. Given that many forests characterized by high fire hazard have an abundance of small-diameter trees traditionally considered unsuitable for conversion into merchantable wood products, concern surfaced as to how to finance fuel treatments in such forests. Much of the removable material consists of “dirty” (i.e., including non-wood components such as bark and foliage) chips generated by chipping entire trees of noncommercial species, and tops and limbs of merchantable trees. A widely touted solution was to further develop the bioenergy industry and encourage appropriate siting of biomass processing facilities to create markets for this otherwise unutilized resource. This led to numerous questions about production economics – how much merchantable wood and dirty chips would be produced by landscape-scale fuel treatment? How much could be feasibly recovered and hauled to processing facilities? Where should bioenergy facilities be built? Increasingly, there were also silvicultural questions – how much area is in need of treatment? What fraction of this could be treated without subsidy? On how much land and for how long would treatments be effective? The list of questions grew quickly, and providing timely responses based on statistically representative data and scientifically valid analytic approaches was challenging. Most questions were sufficiently complex that they went far beyond what any ad-hoc analysis of even the most comprehensive forest inventory data could deliver.

We developed BioSum 0.1 as a framework for analyzing and summarizing woody biomass possibilities, relying on the systematic sample of forests as represented in data collected by the US Forest Service’s Forest Inventory and Analysis program. It included off-the-shelf stand simulation, harvest cost, and fire hazard models; a custom-built, geographically explicit transportation cost model; and multiple technical and decision focused assumptions, parameters and heuristics. The system had to be “hand-cranked” to generate useful information, and iterative analysis and experimentation were woefully constrained by the lack of automation in that analysis “system”. Initial results for a study area in southwest Oregon became available at the time the Healthy Forests Restoration Act was being debated by the U.S. Congress in 2002. They were received with great interest. Considerable interest was expressed in building BioSum models for other areas of the country. An analysis of a larger study area in western Oregon and northern California soon followed, as well as a bi-state effort covering all of Arizona and New Mexico. Results were used, for example, in strategic planning by the U.S. Forest Service Region 3, the California Department of Forestry and Fire Protection, and a group seeking investment in a bioenergy plant in Lakeview, Oregon.

To enable this kind of analysis to be easily, even routinely, carried out anywhere that the requisite data is available, the PNW Research Station’s Focused Science Delivery program provided a seed grant through the National Fire Plan (which was matched and greatly extended by PNW's Forest Inventory and Analysis program) to integrate the tools in the BioSum framework to support analysis of questions about fuel treatment feasibility, costs, and yields, and identifying promising locations for adding wood processing capacity. As development proceeded, new uses were contemplated and the system evolved to allow users to rely on any stand characteristic or set of characteristics as the basis for triggering management activity and/or to define effective management. Moreover, the model evolved from a static, one time evaluation of effectiveness comparing a pre-treatment to a post treatment value to modeling a dynamic, multi-decade sequence of silvicultural activities, costs, revenues and product flows, thanks to tighter integration with the FVS stand projection capability. This advance allowed, for example, evaluation and comparison of treatment longevity among treatments across all kinds of forests. Additional funding to support these developments was provided by the California Energy Commission, the Energy Biosciences Institute, Oregon Department of Forestry, the Joint Fire Sciences Program, and the Pacific Northwest Research Station.

Conceived as an "analyst-friendly" modeling tool (not a "goof-proof" gaming exercise for casual users), this implementation of BioSum is intended for analysts accustomed to using, or willing to master, the Forest Vegetation Simulator (FVS) stand projection model. Familiarity with Microsoft Access is essential for manipulating and querying complex databases, and ability to follow instructions describing a GIS workflow is necessary to compute vehicle travel time estimates for the haul cost component. Users should also understand what kinds of silviculture will both accomplish their objectives (e.g., enhanced resistance to fire, insects and disease; sustainable stand structures; desired habitat elements) and be realistic to implement; they must also be prepared to craft decision rules concerning locally appropriate operational harvest systems for implementing silvicultural prescriptions and what kind of results qualify as evidence of effective management.

For the most part, BioSum, now at version 5.86, is "analyst-friendly" in that interaction with the software occurs via error-trapped entry screens, push-button and check-box option selection, and drop-down menus. We hope that these features will promote analytic efficiency and ultimately reduce errors that would otherwise detract from the validity of results.

BioSum's developers have considered the possibility that users may ultimately wish to replace some components of the system with better or more locally appropriate versions when these become available. The largely modular program architecture and underlying database platform should facilitate such experimentation by the adventurous analyst seeking a slightly different kind of analysis. BioSum's developers have often exploited these features to extend analysis beyond what the software had been designed to do, and to build a complete replacement for the originally implemented harvest cost model. Some users will likely be interested in developing back-end reporting tools for summarizing and displaying simulation outputs. Due to limitations of time and funding, as well as the challenge of choosing which outputs to focus on developing, this tool, as of version 5, is not yet designed to produce presentation-ready output.

Because all outputs are stored as MS Access database tables, analysts can rely on general purpose office-automation software (e.g., Excel) or statistical analysis tools to format, graph and print simulation results. We expect that issues and foci of inquiry will vary among analysts, so some customization in the summarizing and interpreting of results is inevitable. We are happy to share with other users any back-end tools developed for BioSum at the BioSum web site (http://www.biosum.info), as they become available.

Finally, a word of caution – user-specified assumptions and parameters drive this modeling framework at least as much as the inventory data and transportation cost surface. To ensure that model results will be valid, care must be taken when designing and specifying treatments, defining effectiveness, and setting requirements for including full and partial FIA plots in the population of stands to be treated. Choose your assumptions wisely, and enjoy the powerful insights you are sure to develop using BioSum and FIA data!

Jeremy Fried, Portland, OR 31 December 2018

Terms and Conventions

In this guide, we try to be consistent in the use of terms; however, some synonyms and abbreviations are necessary for brevity

Table of equivalents

Rx Package	Silvicultural sequence involving one or more prescriptions over time
FIA	Forest Inventory and Analysis
FVS	Forest Vegetation Simulator
Rx	Prescription (implemented at a single point in time)
Cycle (BioSum)	One of four time points (cycles) in which silvicultural prescriptions may be applied and their effects, costs, and revenues are calculated in a BioSum analysis.
Cycle (FVS)	One of many (typically 12 or 13) user-defined time points (simulation years) in which silvicultural prescriptions may be applied and simulation model outputs are produced in FVS.

Further, computer interface terminology evolves over time, so the terms used here may or may not be those most currently in vogue.

Table of Contents

Chapter 1: Introduction

The Forest Inventory and Analysis (FIA) based BioSum (Bioregional Inventory Originated Simulation Under Management) has evolved as a policy analysis framework and workflow management software into something akin to a Leatherman™—adept at addressing a surprisingly broad spectrum of questions centering on forest management at broad spatial scale. BioSum can address a question as "simple" question as characterizing the distribution of per acre costs and net revenues of applying a particular silvicultural treatment over all the forests for which it might be suited as of today. It can also address more challenging questions that depend on illustrating alternative forest futures based on projecting the outcomes of alternative trajectories defined by the confluence of management, succession and natural disturbance for large, multi-million acre, multi-owner landscapes. BioSum can address complex questions concerning the effectiveness and economic feasibility of managing to improve forest health and resilience, identify propitious sites for facilities that manufacture wood products or generate bioenergy, and quantify the climate benefits produced by enhanced carbon sequestration and substitution potential that result from forest management, in fire-prone landscapes and elsewhere.

All of these questions can be addressed by 1) using FIA plot data to represent initial stand conditions so as to achieve a statistically representative analysis of all forests, or any specified subset, 2) using the Forest Vegetation Simulator (FVS) to implement silvicultural prescriptions, calculate stand metrics that relate to management effectiveness and carbon dynamics, project growth and mortality over time, and simulate disturbances, climate change or any other process for which FVS is parameterized to develop a richly detailed, broad spectrum of potential futures, 3) using modules in the BioSum software to account for costs and revenues of management and evaluate and rank effectiveness of alternative management choices.

BioSum was designed to operate as a "what-if" tool capable of providing objective, statistically valid answers to questions concerning landscape-scale forest treatment. Some examples of the kinds of questions BioSum can address include:

1. Where should a new biomass-to-energy conversion facility be situated?
2. Where is a good place to develop or expand capacity for processing merchantable timber?
3. On how much forest area would treatment X, Y or Z be effective if done today? How would the answer change if we cared about effectiveness over the next 30 years?
4. On how many acres could treatment X pay for itself?
5. How many more acres could be treated if diameter caps were not imposed? What about if $100 per acre was available to subsidize utilization of harvest residues?
6. How would my landscape change, in terms of area by stand structure class or other ecologically-driven aggregate, if fuel treatments were applied wherever they could be effective?

7. How much merchantable and submerchantable wood could a million-dollar fuel treatment program generate?
8. What proportion of the acres where fire hazard reduction would be beneficial are within national forests?
9. How would a putative price change in merchantable wood delivered to a mill or harvest residues delivered to a bioenergy or biochar facility affect the area treated and the wood yield? How do these outcomes vary as effectiveness criteria are changed?
10. Which forest restoration treatments have the greatest longevity?

It is also important to mention what BioSum is *not* designed to do. When used for facility siting, for example, the imprecision inherent in the simplifying assumption that all of the acres represented by a plot occur at the plot location, makes it inappropriate for tactical application in pinpointing the truly “best” site. Other considerations, such as power transmission infrastructure (and air quality constraints in the case of a proposed bioenergy facility), as well as access to work force, may be at least as important as minimizing costs of hauling feedstock. Owing to the typically low density of forest inventory plots, BioSum isn’t well-suited to analyzing small areas. Anything less than several dozen potentially treatable plots would be cause for concern; robust analyses will build on hundreds of plots covering a broad range of forest stand structures. Moreover, BioSum lacks any spatial representation that would address how treatments relate to reduced burned area, for example, since such questions depend on the specific placement of treatments, something outside the scope of this tool.

Our research group uses BioSum to analyze how landscape-scale forest management could play out over the next few decades, under alternative scenarios focused on forest restoration, hazard reduction, and maximization of climate benefits via GHG emissions avoided. How you use the model is up to you. We’re confident that there are many innovative and valid applications that we’ve never considered—we look forward to hearing about the new ways you find BioSum to be useful.

Required Data and Analyst Skills

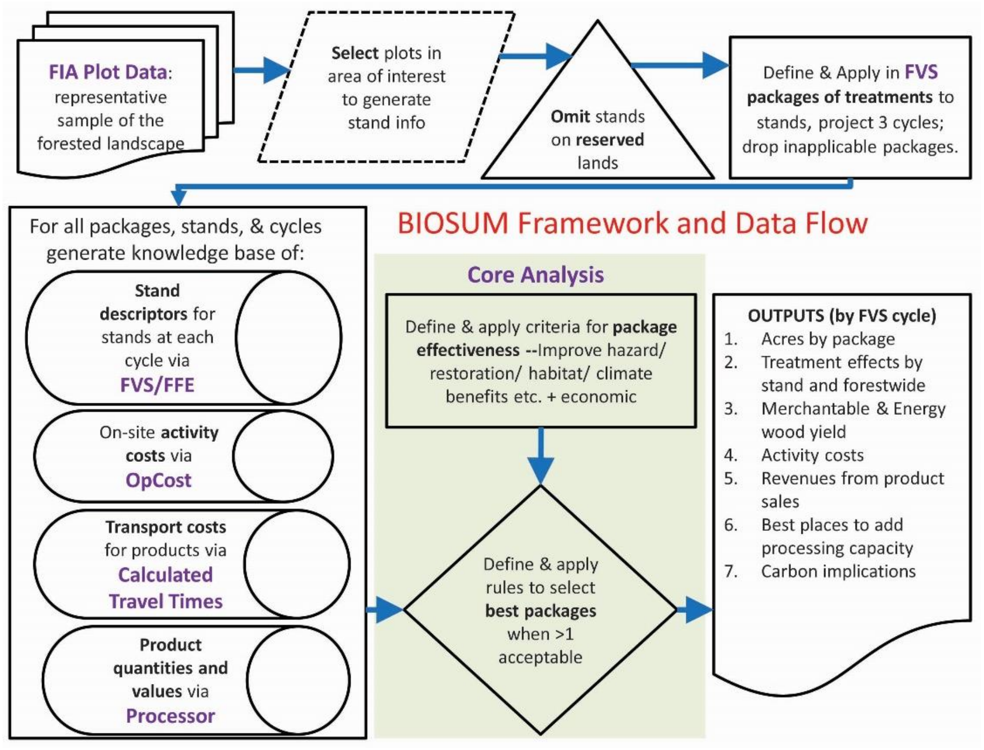
Though the example questions outlined above run the gamut, most depend on the same underlying information: inventory tree lists, plot locations, processing site locations, the road network, and a healthy serving of assumptions and parameters brought by the user, including treatment specifications, effectiveness criteria, delivered prices for wood, and some cost parameters. The specific requirements for these assumptions and parameters are documented throughout this User Guide.

Successful BioSum analyses typically require a team covering multiple knowledge domains, including silviculture, fire and fuels management (if management is focused on enhancing resistance to fire), policy and economics, and perhaps most critically, the FVS modeling environment. An experienced FVS analyst can devise and implement appropriate simulations that account for all prescription elements in each silvicultural sequence, including surface fuel treatment; fuel managers and silviculturists also provide, for example, “yard loss” parameters to account for differences in logging slash generated by whole tree harvest systems versus log-length systems that leave tops and limbs where trees are felled. For FVS variants that don't contain an endogenous "establishment model", local expertise may be required to develop regeneration parameters to be used in the simulation, for example, with the aid of a REPUTE analysis (Vandendrieche 2010).

What is BioSum?

BioSum consists of several components, including a Microsoft Windows™ computer program (BioSum Manager) that manages the workflow of a BioSum analysis (Figure 1), the FIA data on which it feeds, the protocols and guidelines outlined in this Users' Guide, and the problem definition, including assumptions and sideboards, brought by the user. BioSum Manager is a project management "shell", containing several task-oriented modules to manage analysis workflow, that requires that the 32 bit version of Microsoft Office be installed; all other required software is included in the BioSum distribution setup file. BioSum guides analysis workflow through four of BioSum's five modules, while protocols guide the user through travel time calculations, as follows. Starting with the FIA plot data for the user's area of interest, the Database module (1) loads that data as a representative sample of the forests for which management options are to be explored. The FVS module (2) is for defining management alternatives, formulated as a set of silvicultural sequences in FVS, implemented over time, that may differ among initial conditions and may include varying triggering criteria, covering the range of possibilities for which understanding is sought. A no action and/or business as usual sequence is typically an important alternative against which others can be evaluated. BioSum Manager generates FVS readable databases for exchanging information between BioSum and FVS. Travel times (3) are calculated between all plots and all processing facilities for use in computing cost per acre for hauling merchantable wood and utilized harvest residues a.k.a. "chips" or "energy wood". In the Processor module (4), all sequences modeled in FVS are processed through as many alternative Processor scenarios as desired—these may vary, for example, by wood valuation, harvest system used to implement treatments, or the set of available facilities to which wood can be delivered. The Treatment Optimizer (5) guides the user through developing one or more Optimizer scenarios specifying what effective management looks like and setting criteria for selecting the best of multiple effective silvicultural sequences, then summarizes some key landscape metrics, such as area treated by owner class, area feeding each processing facility, and quantities of wood and revenue produced, and cost incurred. It also summarizes outputs for each stand, and because each stand accounts for a statistically determined area of the forest, stand results can be aggregated however desired to describe averages or totals for fractions of the forested landscape that are of particular interest. Many more tables of policy relevant outputs can be produced by querying the tables that result from Treatment Optimizer and from intermediate modules such as Processor.

Figure 1: BioSum Workflow: From FIA Data to Policy Relevant Estimates and Insights


FIA Plot Data: representative sample of the forested landscape
Select plots in area of interest to generate stand info
Omit stands on reserved lands
Define & Apply in FVS packages of treatments to stands, project 3 cycles; drop inapplicable packages.
BIOSUM Framework and Data Flow
For all packages, stands, & cycles generate knowledge base of:
Stand descriptors for stands at each cycle via FVS/FFE
On-site activity costs via OpCost
Transport costs for products via Calculated Travel Times
Product quantities and values via Processor
Core Analysis
Define & apply criteria for package effectiveness --Improve hazard/ restoration/ habitat/ climate benefits etc. + economic
Define & apply rules to select best packages when >1 acceptable
OUTPUTS (by FVS cycle)
1. Acres by package
2. Treatment effects by stand and forestwide
3. Merchantable & Energy wood yield
4. Activity costs
5. Revenues from product sales
6. Best places to add processing capacity
7. Carbon implications

New at Version 5

An infusion of funding in FY18 provided by the PNW-RMA Program's Vegetation Monitoring Science and Analysis team BioSum enabled a temporary expansion of software development and analysis staff that delivered significant dividends in usability and capability compared to earlier versions of BioSum Manager:

- All FIA data is now translated into FVS input databases rather than text files, significantly easing the burden of readying data for analysis in FVS and enabling several new options for conveying additional, plot-specific information to FVS for use in the simulations (see below);
- Users can elect to import FIA down wood, duff and litter and field crew estimated Scott and Burgan surface fuel model into the BioSum project database and the FVS input database for use in simulations rather than rely solely on FVS default assumptions concerning surface fuels;
- Users can elect to import diameter and height measurements of trees collected at previous visits in order to calculate diameter and height growth for use in calibrating FVS projections; this will likely considerably improve the accuracy of FVS projections;
- Treatment Optimizer (formerly Core Analysis) now has access to any measured or calculated stand attribute for defining effective management based on its value at any simulation cycle or on a weighted average or total across all simulation cycles;

- Treatment Optimizer now has the capacity to identify the best management based on both stand condition and economic considerations calculated at any combination of simulation cycles;
- Most modules now include extensive, documented data consistency and integrity checks for analytical quality assurance;
- The harvest cost module, OpCost, was completely revamped and redesigned with an eye towards opening up the assumptions about suitable logging cost equations to the user by placing all harvest system definitions, and equations for the machines that comprise those systems, in Access database tables where they can be easily inspected, modified, disabled or supplemented. This enhancement allows control of which equations are used, enables adding new equations, and greatly facilitates adding new harvest systems, for example, to account for technology change;
- OpCost now includes a brand new set of equations that represent tethered harvest systems; we have found that tethered systems are frequently the lowest cost way to implement restoration management, especially on steep slopes.

Overview of BioSum Modules and Potential Analyses

Chapter 2: Database

A BioSum analysis begins with identifying an area of interest, typically in excess of 500,000 acres, to select plots to be included. This can include all the FIA inventory plots within a geographic area, or just a subset that meets user-specified criteria relating to, for example, ownership class, forest type, or wildland urban interface context. Because FIA data covers all forested lands, it is usually important to identify (and remove from consideration) plots within protected areas such as wilderness, parks and designated roadless study areas, for example. Some criteria for plot inclusion can be assessed from the inventory database (e.g., wilderness conditions will be attributed as reservcd=1 in the COND table); others can be derived via overlay with GIS layers. Some plots have more than one condition present – for example, two age classes of forest, or a forest and a non-forest condition, or forest controlled by members of two different landowner groups (e.g., a National Forest and a private holding) – so conditions are the basic analysis unit on which treatments are simulated. Conditions become stands when loaded into FVS.

Because the FIA inventory is a statistically representative sample of the whole forest, BioSum calculates an "expansion factor" for each condition that accounts for the number of acres in the forested landscape represented by the condition. These expansion factors are crucial for generating BioSum results that can be applied to the greater landscape. Data in FIADB formatted CSV files can be downloaded from the national FIA website (https://apps.fs.usda.gov/fia/datamart/datamart.html) and imported into an Access database to make it ready for use by BioSum. The database module imports this data into a "project" directory structure (a set of nested directories or folders) under which all information for a

BioSum analysis is stored and processed, including inputs, intermediate databases and final outputs.

Chapter 3: Travel Time Geoprocessing

There are two important calculations that require spatial analysis: 1) calculating the haul time from the plot to each potential processing site (this is used, with hourly truck operation cost and capacity, to develop unit transportation costs per acre for each stand), and 2) calculating the distance from the plot to the nearest road for use as a forwarding/skidding/yarding distance when modeling harvest costs. The timing within the overall BioSum workflow for these calculations is flexible, as long as it follows completion of the Database module and precedes the Treatment Optimizer. The analysis relies on a two point layers (plots and potential processing sites), and a line layer (roads) attributed with travel time per unit distance. This workflow can be carried out in ArcGIS or R, following the procedures outlined in this user's guide. The outputs are a table of truck travel times in units between the publicly-available approximate locations of each FIA plot and a user-supplied list of existing and potential processing facilities that accept merchantable wood or harvest residues.

Chapter 4: FVS

Silvicultural prescriptions to be modelled are described and labeled by the user in this module, then assembled into sequences of prescriptions (referenced as "packages" in BioSum parlance) that are applied over time. Multiple treatments can and should be specified so that in the later stages of analysis, the Treatment Optimizer module will have a range of choices from which to choose for each stand. It is generally the case that no single prescription is best for all stands, no matter how "best" is determined.

The actual prescriptions in the form of FVS keywords and parameters are developed separately within the FVS software environment in SUPPOSE or off-line. Prescriptions designed for forest restoration are typically specified in terms of which trees to prioritize for removal and/or retention (for example, based on size and species), desired residual stand density, and limits on the size of trees allowed to be harvested, although prescription design is really only limited by the analyst's imagination and expertise in translating their ideas into a keyword file.

An unlimited number of silivicultural sequences can be specified, each containing a prescription or grow-only (no action) choice applied at the outset of each of four growth cycles of five or ten years each. Harvesting systems can be specified separately for each prescription, if desired, or specified as applicable to all prescriptions at a later stage of the analysis when developing Processor scenarios. BioSum performs a number of translations on and reformats the FIADB data to ready it for projection and simulation in FVS (e.g., supplying site index values compatible with the site index equations used by FVS and mapping FIA species codes to their FVS counterparts), then generates one FVS readable Access database per FVS variant that can also be populated with FIA collected information on down wood and other surface fuels or Scott and Burgan surface fuel models, when available.

The last workflow phase in the FVS module imports FVS output into the BioSum project environment so that, for every tree in the FVS CUTLIST tables (which contain records of trees harvested by the prescriptions applied in each silvicultural sequence at each growth/treatment cycle), biomass and volume can be computed using the same species and region-specific FIA equation systems used in standard FIA compilation and reporting systems.

Chapter 5: Processor

BioSum's PROCESSOR module accomplishes two essential tasks to characterize, for each combination of stand and silvicultural sequence and for each growth or treatment cycle, wood production and associated value and costs: 1) computing per acre volume, weight and value of a) harvested merchantable wood, by user-specified species group and tree diameter class, and b) "dirty chips" (so-called because these chips derive from harvest residues delivered to the landing, and contain non-woody materials such as bark and foliage that make them unsuited for producing pulp, but potentially salable as fuel for bioenergy generation, depending on markets); and 2) estimating per acre treatment costs for harvest operations and any related activities such as treatment of surface fuels. Most of task 2 is accomplished using the OpCost software (Bell et al. 2017a), developed in R, with logic similar to Dykstra et al.'s (2009) Fuel Reduction Cost Simulator (FRCS; Fight, Hartsough and Noordjik 2006) which OpCost replaced. OpCost differs from FRCS in that it relies on parameters from more recent cost studies for the machines used in some of the supported harvest systems and is backed by a validation study (Bell et al. 2017b).

OpCost relies on machine production rates from the peer-reviewed harvest operations literature, and based primarily on elemental time analysis studies, to estimate harvest cost for all simulated forest operations activities in BioSum. The key inputs are similar to those in FRCS: harvest system, field observed slope and yarding distance (relying on distance from plot center to the nearest road as a proxy) at the stand level and, for four tree size classes, average volume per tree, number of trees harvested per acre, wood density, hardwood fraction, and residue fraction. OpCost estimates machine usage times for all machines in a harvest system, and translates these to cost estimates based on conventional cost control processes (D. M. Matthews 1942). OpCost also estimates "move-in" costs for getting equipment to a treatment site and ready to operate. These are ultimately combined to generate a total cost to harvest and process wood volume on a per acre basis.

Chapter 6: Treatment Optimizer

Treatment Optimizer integrates intermediate outputs from all the analyses generated via earlier modules in the workflow into user-crafted scenarios that address the landscape scale policy questions for which BioSum was designed. In Treatment Optimizer, analysts can interactively specify effectiveness criteria based on any of the FIA or FVS attributes at any simulation cycle or combination of cycles (e.g., weighted average values or totals over all cycles) to define what should be considered effective management. For example, effectiveness could depend on the post-treatment crown fire potential at cycle 1 being 20% lower than the pre-treatment crown fire potential, as indicated by one or more stand attributes. Alternatively, effectiveness could require that mean mortality volume over all four cycles under moderate fire conditions be lower for silvicultural sequence with active management than the mean mortality volume in a grow-only sequence. With the right choice of COMPUTE variables specified in the

FVS prescription keyword file to allow tracking this attribute, one might deem sequences that move 50% of the stand basal area into trees larger than 25 inches dbh by the end of cycle 4 to be effective if the goal is to accelerate the development of old growth characteristics. Up to four different stand attributes can be used in defining management effectiveness, and complex logic is supported, including making effectiveness contingent on improvements in more than one attribute, or limiting the allowable disimprovement in one attribute that accompanies the improvement in another.

Stand-sequence combinations that don't achieve the user-defined effectiveness threshold drop out of consideration, and those that remain are subject to heuristic optimization, where a user's overarching goal *after ensuring management effectiveness*, drives selection of the overall best treatment for each stand and the acres it represents. An optimization criterion could, for example, be minimizing crown fire potential, maximizing quadratic mean diameter in the residual (post-treatment) stand, maximizing mean canopy cover over 40 years, or minimizing net management costs (or maximizing net revenue). Users also specify attributes or treatment rankings to break ties that might occur in the optimization so that, for a given scenario, there will be a single best sequence identified for each stand that has one or more effective silvicultural sequences.

Treatment Optimizer can then summarize outcomes of the scenario for the whole forest landscape—for example, how many acres can be effectively managed, how frequently does each silvicultural sequence considered prove best, how much merchantable wood and collected harvest residues would be generated, and net revenue or cost, per cycle, for implementing the management program on all eligible acres. These kinds of summaries can be aggregated and reported, for example, by ownership, product species group, product tree diameter class, forest type, stand density class, treatment style, or cycle (decade). Aggregation of outputs is also possible by wood processing facility. Sequences and acres can be filtered by whether net revenues exceed any specified threshold, including negative values that indicate allowance for some degree of subsidy. Some of these tabular outputs are reported automatically in database tables; others can be easily generated via queries of the Treatment Optimizer table output.

Advanced Ad-hoc Analyses

For some questions, linking Treatment Optimizer results to attributes in the plot and condition tables, some of which may have been added to these tables via GIS overlay or other model application, can address a broader set of questions that link to location. For example, to name a few, how much of the wood from fuel treatments would be generated within the wildland urban interface? How much of the area requiring treatment is on steep slopes? How does site class impact treatment effectiveness and longevity? How does management success and cost relate to owner group? Some analysts may wish to explore post-treatment forest structure and fire resilience, compare post to corresponding pre-treatment values, compare habitat quality under a management sequence with what a grow-only simulation delivers, or track carbon pools over time, in managed and unmanaged sequences, with and without disturbance at different frequencies and intensities. All of the FVS table outputs that the analyst chooses to specify in their KCP files (that control FVS operation), including carbon, trees, potential fire and more are available for this kind of analysis by analysts comfortable developing database queries to stitch these different kinds of results together with the BioSum outputs. Moreover, because each stand is linked to a plot with a defined location in space, some users may wish to explore results geographically, for example, to identify regions where treatment effectiveness is generally high, economic feasibility is high, or pre-treatment hazard is especially high. It can also be useful to plot stands attributed (i.e., shaded) by product destination to convey the concept of potential woodsheds under a given configuration of processing sites.

An earlier version of BioSum was used to jointly optimize treatments and wood processing facility locations (Daugherty et al. 2008), and this may be of interest to some users. In that application, analysts extracted the data developed from the modules described in chapters 2-5, as input to a mixed-integer optimization algorithm to arrive at a joint optimization solution to the problem of which treatments to apply and which processing sites to build (and to what capacity). We used the same kinds of objectives for treatment effectiveness as can now be applied as heuristics in the Treatment Optimizer module.

Work Flow Considerations

With the exception of Geoprocessing, for which there is some flexibility as to the “when”, the modules outlined above must be executed sequentially because each successive step relies on analysis completed in the previous step. When performing iterative simulations, it is sometimes necessary to revisit earlier steps to implement alternative assumptions. For example, to model the effects of different assumptions about merchantable wood prices, the analyst would need to rerun the Processor module before running a new Treatment Optimizer Scenario (and linking to the newly created Processor scenario). To understand the effect of a different network of processing sites, the analyst would revisit geoprocessing operations before re-running Treatment Optimizer.

The time required to complete each module varies. Database can typically be completed quite quickly – in as little as an hour for a single state, depending on whether down wood and calibration data are also requested. Geoprocessing for a large area can requires several hours to complete, and has some iterative elements that may require multiple analyses to get a complete travel times table. However, this workflow is usually only done once. After silvicultural prescriptions have been generated as KCP files (the challenging part), the push of a few buttons is all that is required to initiate and complete the tasks in the FVS module, but some of them take several hours for a statewide project with many prescriptions specified. Running the prescriptions in FVS requires the most time; on some statewide projects, this has consumed a week of computer time, though that particular task can be divided among multiple computers and batched up so that it can run largely unattended.

Processor is easy to setup, and runs unattended for several hours when there are many prescriptions (e.g., 30) and multiple FVS variants for the project area. Treatment Optimizer runs quickly once the effectiveness and optimization criteria have been established by the user, usually completing in 2 to 5 minutes, though depending on how complex and ambitious, defining those criteria could take as little as 5 minutes or as long as an hour, the first time through. That rapid execution turnaround time makes it easy to modify and rerun scenarios to see the effect of different assumptions on the questions of interest.

A Word About Simulation

Simulation models like BioSum are designed to capture the essence of a “system” (in this case, the application of landscape-scale forest management) in an area, volume, value, cost and effectiveness accounting model intended to mimic the functioning of the system. If sufficiently realistic, simulation models can help identify solutions to critical problems, and robust models may even be useful in addressing problems not anticipated when they were developed. One motivation for developing this simulator was concern that much of current decision-making about fuel treatments, and forest management in general, is driven more by conjecture and belief, or by distortions introduced by simplistic performance benchmarks and the budgeting process, than by forest and fire science.

One of the under-appreciated aspects of simulation models is their ability to enhance our understanding of a system by quantifying the relationships within the system being modeled. In the case of using BioSum to evaluate forest restoration potential, we are forced to quantify achievement of fuel hazard and other restoration goals, and to be explicit about the silvicultural treatments we would consider implementing to them. We are also forced to examine closely the costs associated with implementing prescriptions and with hauling removed material to places where it can be utilized, providing some revenue to offset management costs.

This kind of simulation can be very useful for assessing the impacts of changes in both the policy environment (e.g., what kinds of fuel treatments are acceptable, how much funding is available to subsidize fuel treatments) and the economic environment (e.g., investment in new processing capacity, availability of new, low-cost techniques for removal of small diameter woody material). Simulation can be automated so as to produce a wide variety of possible scenarios and associated outcomes, without regard to current treatment plans, in the hope that a near optimal solution will be identified. Alternatively, it can begin from a “base case” that represents current plans and conditions, and be guided by an analyst familiar with the system and with the range of scenarios that might be worth considering.

Given the time and expense required to build and run a simulator like BioSum, some will ask, “Why bother?” One answer is that the complexity of managing large, multi-owner, forested landscapes is such that good decisions are difficult to make. Another is that bad siting decisions could have catastrophic consequences for biomass processing investors, the ecological integrity of the forested landscape, or both. While an experienced procurement forester may recognize and understand some of the simpler qualitative relationships in the system, predicting economic feasibility is far more difficult. Complex interactions can be explored in-depth using simulation, and intuitive expectations with respect to simpler relationships can be rigorously tested and refined.

In any case, simulation offers a safer means of experimenting with a system than implementing changes on a trial basis. No prudent land manager would embark on a fuel treatment program without some confidence that there would be markets for the materials removed (or that funding would be available to cover treatment costs) and that the treatments would be effective. No investor would build a bioenergy facility without confidence that a consistent supply of feedstock would be available to sustain it over a reasonable period of time. This kind of tool offers all involved an opportunity to conduct due diligence and to arrive at an analytically supportable basis for management decisions that commit funds and/or future actions. In this context, a simulation approach can lead to better management decisions.

Simulation modeling is both a science and an art because of the inescapable trade-off between model realism and data requirements. Relatively simple models such as “my fuel treatment planner” can help us understand treatment options, costs and effectiveness at a stand level, but are not of much help in answering questions about where to build processing capacity or forecasting how much funding would be needed to have an appreciable impact on fuel hazard at landscape scale. On the other hand, extremely complex models, capturing every nuance and possible permutation of the system, can be burdensome to parameterize and difficult to interpret. It is generally agreed that the best models are those which capture the elements of a system that is of critical concern to the clients needing the model, without being needlessly complex. BioSum’s design is the result of a conscious effort to balance complexity with realism, and to allow the user to make a number of decisions concerning the appropriate level of complexity. Simulation parameters for everything from effectiveness criteria to prescription sets to harvest and haul costs can be adjusted to enhance model realism. Sensitivity analysis will be an excellent way to examine the impacts of the assumptions specified by model users and to provide information about which may merit more intensive effort to perfect.

In developing BioSum, we tried to tailor the model to the needs of both land managers and those interested in building processing capacity. We hope that these users will find the end product of our collective effort to be worthwhile.

Learning BioSum

This analysis system is robust, comprehensive and flexible; however, it also has the potential to feel overwhelming from the perspective of a new user. Analysts should allow themselves time to come up to speed, and carefully read the documentation for each module to ensure that the analyses they conduct will be valid. While the treatment optimizer module is enticing and eagerly anticipated by most users, the results will only be valid if care has been taken in the preceding workflow.

New users are advised to start small—pick a small area and develop just a couple of prescriptions to build a test dataset to explore the kinds of analysis that are possible using Treatment Optimizer. Then, if all looks promising, collect and assemble the data, assumptions and parameters required for all modules to support an analysis of the target study area. For cases where this documentation does not address questions you have about how to run the model, BioSum experts at PNW-FIA can be contacted for advice. CONTACT INFO

Chapter 2. Database

The Database module is the first step in the BioSum workflow. This section describes how to navigate the BioSum menu, create a new project, open an existing project, add and delete forest inventory data, and manage files in a BioSum project.

Creating a New FIA BioSum Project

1. To create a new project, start the FIA BioSum Manager application and select <File><New Project> (see Figure 2.1).

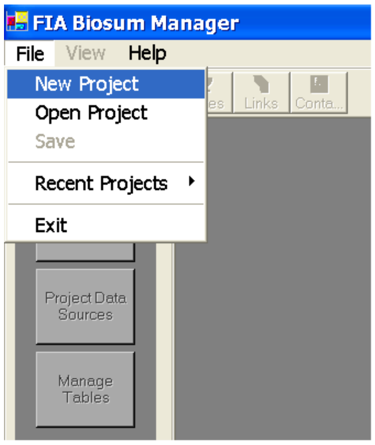
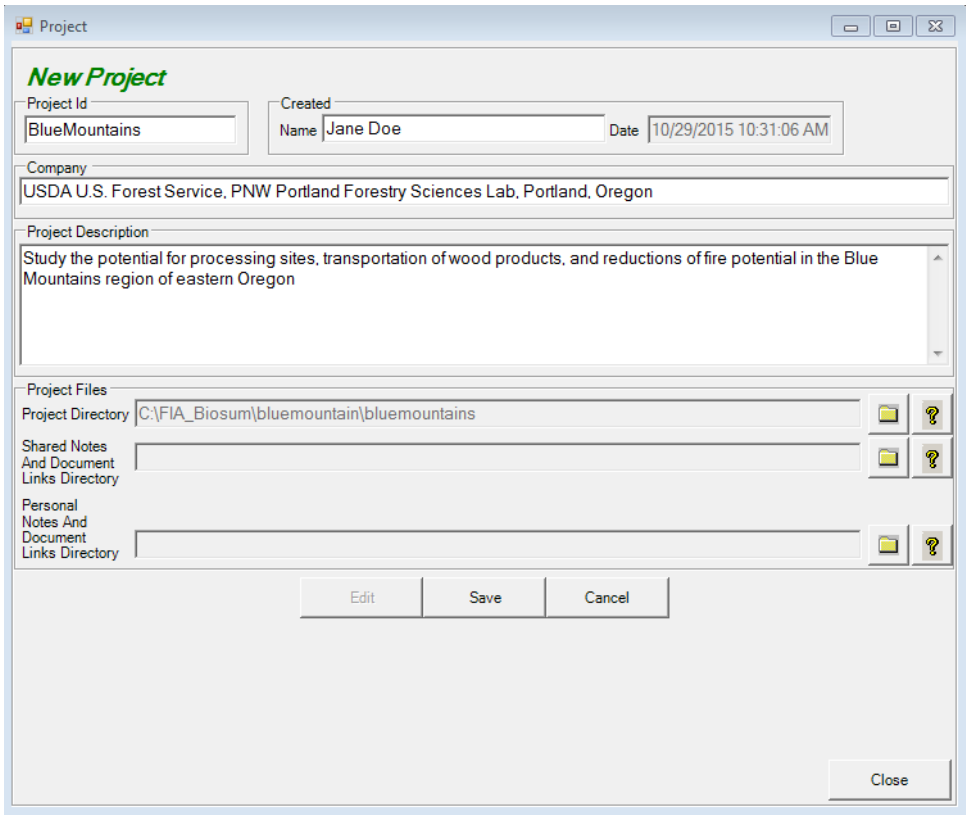


Figure 2.1 – Select New Project from the main menu.

2. Upon selecting <New Project>, the Project form will open (see Figure 2.2). The only required information is the Project ID field. By default, the project directory will be c:\FIA_Biosum\project id. For example, if the project id is 'BlueMountains' the default project directory would be c:\FIA_Biosum\BlueMountains. The default project directory can be changed by selecting the folder icon to the right of the Project Directory text box and navigating to a different directory.



Project Id: BlueMountains

Created — Name: Jane Doe — Date: 10/29/2015 10:31:06 AM

Company: USDA U.S. Forest Service, PNW Portland Forestry Sciences Lab, Portland, Oregon

Project Description: Study the potential for processing sites, transportation of wood products, and reductions of fire potential in the Blue Mountains region of eastern Oregon

Project Files

Project Directory: C:\FIA_Biosum\bluemountain\bluemountains

Shared Notes And Document Links Directory:

Personal Notes And Document Links Directory:

Figure 2.2 – Specify project name, description, and directory.

3. To share project notes and documents with others, click on the folder icon to the right of **the Shared Notes and Document Links Directory**. NOTE: To effectively use the shared document links option, users must have access to the shared directory, and use the same drive letter when mapping to the shared directory. To keep personal project related notes and documents, click on the folder icon to the right of the **Personal Notes and Document Links Directory.**
4. Click the <Save> button to create the new project in the BioSum Manager.
5. The BioSum Manager will create the project directories and copy the necessary project files. Select **<Yes>** if you want the document links depository to keep track of project documents. Select **<Close>** to exit the Project Properties window once the project has been created.
6. When a project is created, the project root directory and subfolders in Figure 2.3 are created.

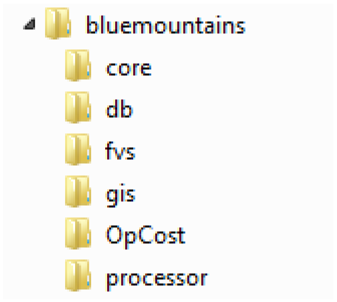
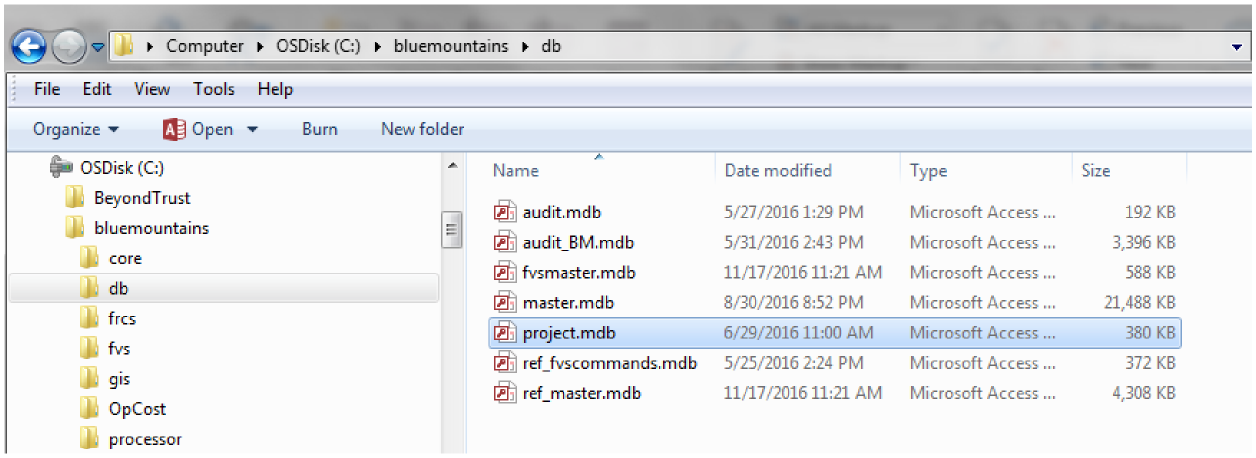


Figure 2.3 –Directory structure of a BioSum project.

Opening a Current FIA BioSum Project

1. To open an existing project, select **<File><Open Project>** on the main menu, or select **<Open>** from the toolbar, and navigate to the project root directory and db subfolder. Choose the file ‘project.mdb’ (see Figure 2.4). Alternatively, **<File><Recent Projects>** can be used to select from a pick list of projects that have been recently opened or saved.


Computer ▸ OSDisk (C:) ▸ bluemountains ▸ db
File Edit View Tools Help
Organize Open Burn New folder
OSDisk (C:)
BeyondTrust
bluemountains
core
db
frcs
fvs
gis
OpCost
processor

Name	Date modified	Type	Size
audit.mdb	5/27/2016 1:29 PM	Microsoft Access ...	192 KB
audit_BM.mdb	5/31/2016 2:43 PM	Microsoft Access ...	3,396 KB
fvsmaster.mdb	11/17/2016 11:21 AM	Microsoft Access ...	588 KB
master.mdb	8/30/2016 8:52 PM	Microsoft Access ...	21,488 KB
project.mdb	6/29/2016 11:00 AM	Microsoft Access ...	380 KB
ref_fvscommands.mdb	5/25/2016 2:24 PM	Microsoft Access ...	372 KB
ref_master.mdb	11/17/2016 11:21 AM	Microsoft Access ...	4,308 KB

Figure 2.4 – To open a project, navigate to and open the project database in BioSum.

2. Once a project is open, the BioSum Manager provides several tools to help manage a BioSum project. The tools include project properties (**Project** button), which allows edits to the project properties; project notes (**Notes** button), where notes and project details can be stored; the document links depository (**Links** button), which allows project-related documentation to be uploaded, managed, and deleted; and contact information (**Contacts** button), where project-related contacts can be stored. These tools can be accessed from the toolbar located at the top of the BioSum Manager main form. When one of these tools is opened, its window may need to be resized to display content that may otherwise be hidden.

The buttons located on left pane of the FIA BioSum Manager window (see Figure 2.5) provide access to the four principle task groups that comprise the workflow of a BioSum analysis: (1)

Database, (2) **FVS**, (3) **Processor**, and (4) **Treatment Optimizer**. The currently selected task group will be displayed at the top of the pane with its title in red (indicating it as the active task group). The tasks associated with the task group appear as gray buttons directly beneath the task group name. The **<Database>** group provides access to four tasks:

1. **Plot Data**- Add forest inventory data to, or delete data from, the currently active project.
2. **Wood Processing Sites**– View, edit, or delete wood processing sites. See Travel Time Calculation in chapter 3 of this user guide for details on how this information is used.
3. **Project Data Sources**– View the directory paths and names of the project database files and tables and the number of records they contain.
4. **Manage Tables**– View project MS Access database files: browse contents of database tables, open a database in MS Access, or compact a database file nearing the 2 GB Access database size limit.
5. **Manage SQLite Tables** – View project SQLite database files: browse contents of database tables or open a database in SQLite editor (e.g., SQLiteStudio).

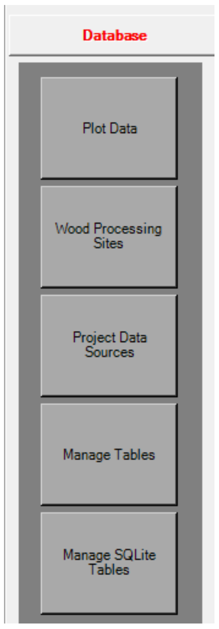


Figure 2.5 – Task buttons for the Database task group.

Adding Data

Input Data Sources

The FIA BioSum Manager loads annual inventory data from state-level FIADB SQLite databases available at https://apps.fs.usda.gov/fia/datamart/datamart.html as an "append" process that can expand the area covered by an existing project, for example, to add adjacent states.

It is highly advisable to save (for the life of the project) the state level source FIADB SQLite database from which the BioSum project was populated. That will ensure that the area of interest can be expanded to include other parts of the state, should that become desirable. FIADB is a "live" database that may update annually (or more frequently), and the sample population (POP) tables in updated versions of that database will become unsynchronized with the plot data (and the simulated projections of that plot data) in the BioSum project.

The "Adding Data" task will guide you through a multi-step process to add FIADB data to a project: 1) download FIA data from the national web site as an SQLite database; 2) tell BioSum the location of the SQLite FIADB database; 3) select an evaluation set to be analyzed (e.g., which years?); and 3) specify plot filters, if analyzing less than an entire state.

Plot filtering options include inputting all plots in a state FIADB file, filtering by menu selection for 1) particular counties and/or particular plots within counties, 2) plots with forested or non-forested conditions, or 3) filtering by a text file containing a list of plot identifiers that correspond to PLOT.CN values in the FIADB database.

Users can also choose whether to include two types of data in the process that loads FIADB data to BioSum databases: 1) inventory seedling data and 2) Down Woody Materials data.

Inventory plot data can be loaded in one operation, or incrementally. For example, for a multi-state analysis, the data from each state can be iteratively "appended" to the project.

When FIA BioSum appends data, it does not import every attribute from FIADB; it adds only the attributes needed by BioSum, matching the field names of the source table with the field names of the target table. As a result, custom or calculated attribute values can be included within a BioSum project only by adding these as columns in BioSum's plot, condition, or tree tables in the MASTER.MDB database, typically by running update queries that link the BioSum tables to other tables such as those in the FIADB or other databases.

Adding FIADB Data

1. To begin, click the **<Database><Plot Data>** buttons on the side menu shown in Figure 2.6.

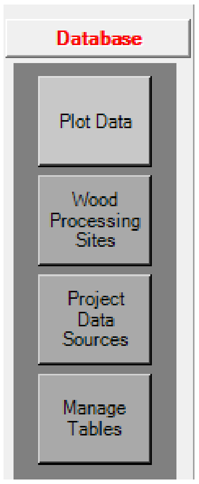


Figure 2.6 - Plot data button.

7. The **Plot Data** form will open. Click the **<Add Plot Data>** button depicted in Figure 2.7.

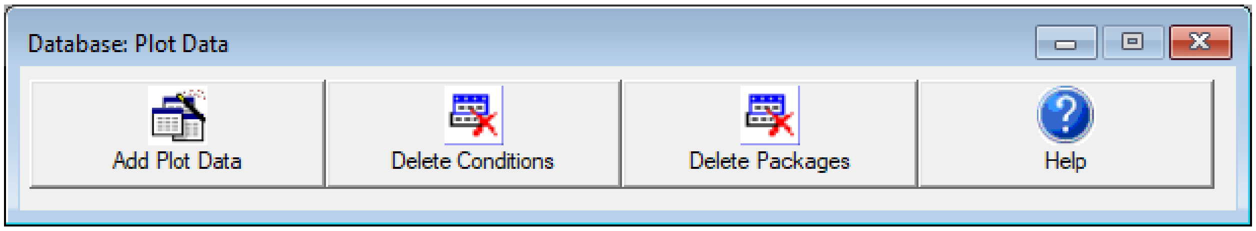


Figure 2.7 – Plot data management options.

Please note that windows that open during the BioSum workflow are sometimes hidden behind other windows displayed on the computer's desktop. A good way to find them is to hover the mouse cursor over the BioSum icon in the Windows Start bar—if there are additional BioSum windows open, you will see them as choices to which you can switch by selecting the mini-view of the desired window.

Note also that if adding plot data after previously deleting some or all of the plot data in a BioSum project, steps must be taken before adding the data to avoid corruption of the data in the project—see caveat/caution discussion under "Deleting Data" in the next section.

8. The first dialog box of the **Plot Data Input** form will appear (see Figure 2.8). Click the folder icon to browse to the location of a state level FIADB SQLite database, and then select the database to load.

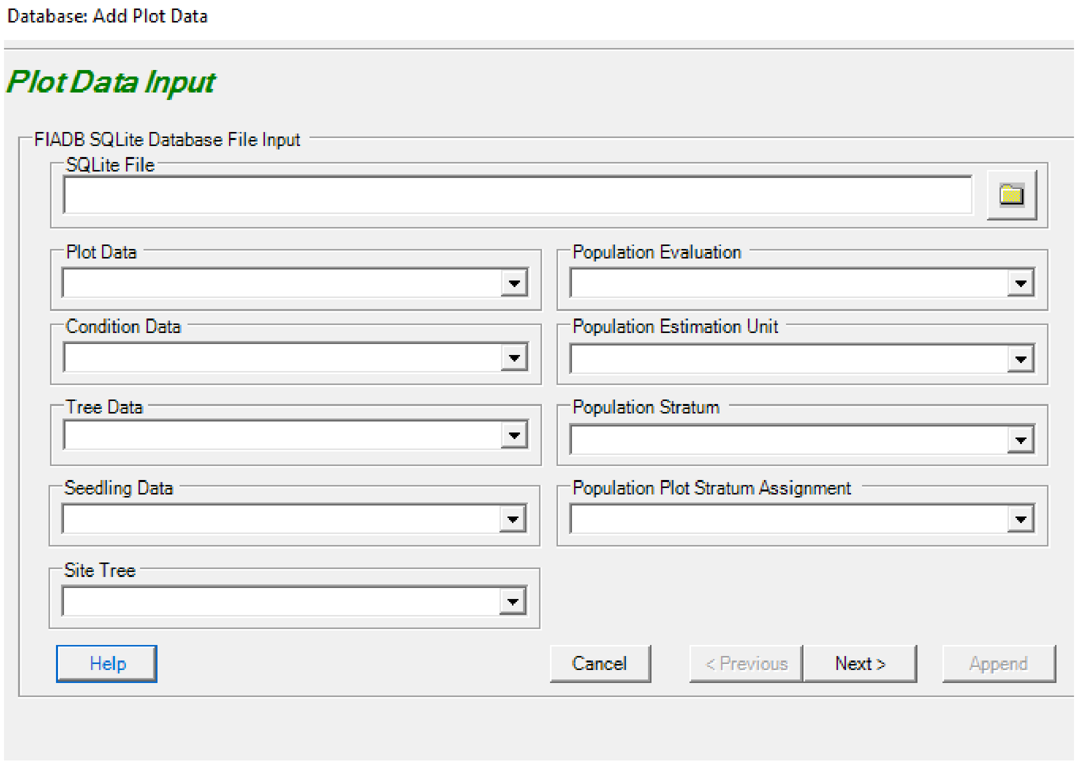
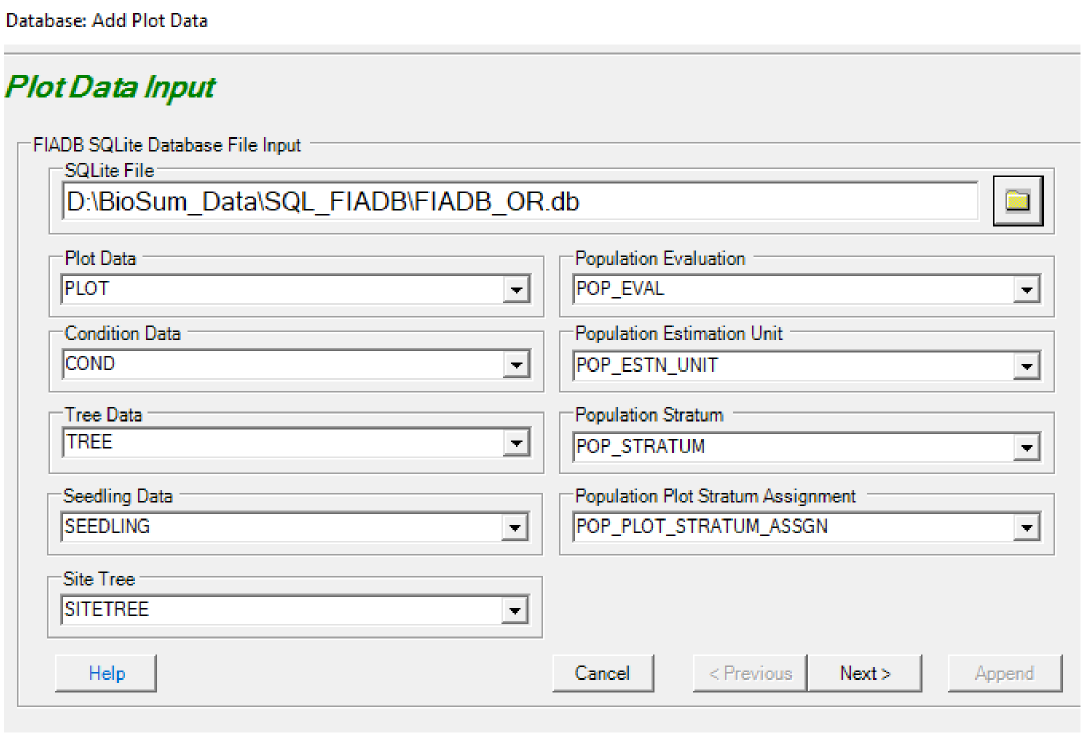


Figure 2.8 – Plot data input.

9. After clicking the desired database and then clicking **<Open>**, the 9 table name fields will populate automatically if the FIADB database contains tables with the standard table names for each type of data (see Figure 2.9).


Database: Add Plot Data

Plot Data Input

FIADB SQLite Database File Input

SQLite File: D:\BioSum_Data\SQL_FIADB\FIADB_OR.db

Plot Data: PLOT	Population Evaluation: POP_EVAL
Condition Data: COND	Population Estimation Unit: POP_ESTN_UNIT
Tree Data: TREE	Population Stratum: POP_STRATUM
Seedling Data: SEEDLING	Population Plot Stratum Assignment: POP_PLOT_STRATUM_ASSGN
Site Tree: SITETREE	

Figure 2.9 – Populate form with appropriate tables from the database.

10. A combination of filters and user choices, specified via a series of dialogs, determines which data are loaded into a BioSum project. Temporal Selection (the last dialog in the series, described in Step 8), determines the inventory data collection “window” on which BioSum analysis will be based. In the Western U.S., this “evaluation set”, represented by an EVALID, in FIA parlance, typically consists of the most recent 10 panels of field sampled plot data—for example, panels collected between 2010-2019 (inclusive).

    The first dialog(s) provide for plot and condition filtering, that in BioSum’s internal workflow are applied after selecting the desired evaluation set, for area of interest, forested status, and condition proportion threshold. These dialogs also provide for choosing 1) whether to load tree seedling data, which may, over time, affect stand density, competition, and likelihood of crown fire, and 2) whether to load down wood data that can, for example, be useful in representing surface fuels or to provide a more complete representation of carbon stores. If seedling and down wood data are to be available in FVS, they must be both loaded when building the project **and** specified as a desired element of the FVS input files when building those files in the FVS module.

    a. Area of Interest

For the temporal window selected, the “Input all Plots” radio button loads all plots in the state FIADB database. “Filter by Menu Selection” defers interactive selection of counties and plots to a later step, following Temporal Selection. “Filter by File” initiates browse and select for a user-created text file containing the PLOT_CN values associated with all plots to be included. For each option, the last step when reaching the last of the dialogs will always be to **Append** the data to the project’s master database. We use the term Append rather than Load because it is not uncommon to need to expand an existing project by expanding the area of interest, which entails loading additional plots. Caution: If appending plots to an existing project, it is crucial not to attempt to reload already loaded plots. BioSum will detect this as non-compliant and throw an error. Ensure that the list of plots to be appended does not include plots already loaded from FIADB to BioSum, or if this is not possible, then first run the condition zapper to eliminate data associated with conditions linked to any of the plots about to be appended. The condition zapping approach has been less thoroughly tested for this purpose, so avoiding appending duplicate instances of plots is the preferred and recommended approach.

Option 1– Input All Plots. Data from all plots in the state will be added (subject to other filters specified). To add all plots, select **<Input All Plots>,** after first selecting desired options for forested status and condition proportion (see below), then click **<Next>**. A list of FIADB EVALID’s will appear in the temporal selection dialog (see step 8 for guidance). Select the evaluation set of interest and click <**Append**>. Loading of population tables and inventory data may continue for several minutes (even several tens of minutes, depending on dataset size); thus, patience is essential and the temptation to cancel the process, based on a (most likely) incorrect assumption that the load is failing, should be resisted.

Option 2– Filter Plots by Menu Selection. Plots will be filtered to include only selected plots from selected counties, where counties are selected via county codes and plots by FIADB Plot number.

I. Select <**Filter Plots by Menu Selection**> and set options for forested status and condition proportion (see below) as desired, then click **<Next>**.
II. A list of FIADB EVALID’s will appear (see step 8 for guidance). Select the desired evaluation set and click **<Next>**. Loading of population tables may continue for several minutes (or potentially hours), even on computers with fast processors, so be patient.
III. When the **Filter by State and County** dialog appears, select desired counties (counties in only one state can be displayed for any given load operation) (see Figure 2.10). Click **<Append>** to add all plots within the selected counties. Alternatively, select specific plots within the chosen state(s) and counties, by clicking the **<Next>** button to

display the **Filter by Plot** dialog, select the desired plots, then click **<Append>**.

Option 3– Filter by File. Plots will be filtered based on Plot CN (control number) values listed in a text file that you create, one value per line and with NO header line and no delimeters such as quotes, even though the CN "value" found in PLOT (and in the FIADB database's FVS_STANDINIT_PLOT (which is NOT used by BioSum) as STAND_CN, is a text field, NOT the very large integer that it looks like (see Figure 2.11). Special care must be taken with any lengthy text attribute when working with such data in applications or computing environments such as R or Excel as these have a tendency to convert text data that LOOKS like a number into a numeric data type which, for long numeral strings, often truncates some of the digits, corrupting the data and breaking your analysis workflow in BioSum.

The contents of a file like JaJoSiPltCN.txt specified on the "Filter by File" line might start something like this:

```
29880599010497
40996214010497
40218925010497
40218934010497
```

To designate the text file, select <Filter by File (Text File Containing Plot_CN numbers)>, choose the text file, click <Open>, select the inventory of interest, and then click <Next> to load the population tables and generate the Eval ID selection screen. Select the evaluation set of interest and click <Next>. Loading may continue for several minutes, even on computers with fast processors, so be patient.

CAUTION: there are several different identifiers for plots and conditions in FIADB, BioSum and FVS and some users have become confused as to which one to use. For PLOT LOADING, it needs to be the CN value found in the PLOT table.

b. Forested status

The Forested and Non Forested check boxes act as follows. Checking Forested loads any plot with one or more forested conditions. Checking Non Forested loads plots with no forested conditions. Checking both loads all sampled plots. <u>Most users will likely need to load only plots that contain forested conditions</u>. The opportunity to load completely non-forest plots might be useful for an afforestation analysis; this capability has not been tested.

c. Condition Proportion

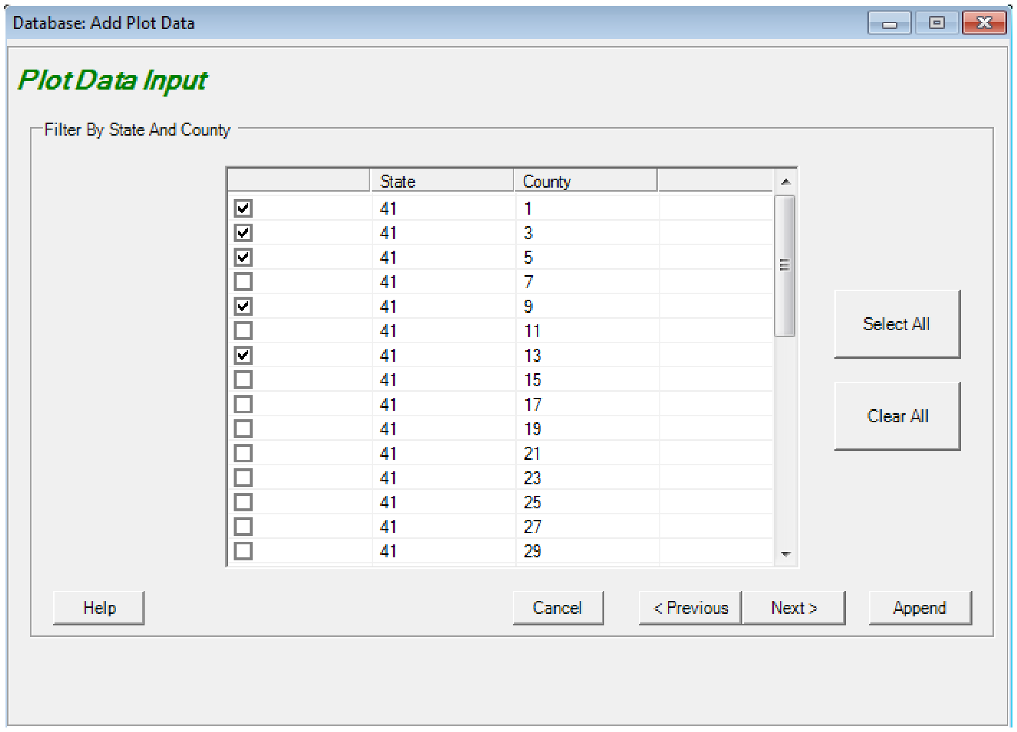
Select a value for the condition proportion threshold (expressed as a percent) between 1 and 99 using the dropdown list; the default is 25. At the default value, tree data for conditions with condition proportion (fraction of the plot the condition accounts for) of 0.25 or larger (equivalent to the size of one subplot) will be imported into the BioSum Project; for conditions with smaller condition proportions, the condition will be considered nonsampled, and the area accounted for by that condition will re-assigned proportionally to all the other conditions within the same stratum (for example, national forest - forested land) that have a condition proportion of 0.25 or greater. This helps to protect against modeling conditions containing so few trees that the degree to which the data adequately represents a "stand" for the purpose of exploring management options, is called into question. This threshold can be adjusted, if desired. Increasing the threshold towards 100% will exclude more conditions from contributing to the BioSum analysis while increasing the average count of trees per plot associated with the conditions that remain in the sample; decreasing towards zero will exclude fewer conditions, and increase the inclusion of conditions that may contain very few trees, potentially generating anomalous outcomes for some conditions.

d. Tree Seedling Data

Tree seedlings recorded in FIA plots can be loaded into the project by checking the "Include Seedlings" box before proceeding with an "**Append**". Seedling records will be written to the project's \db\master.tree table and can later be included/excluded from FVS Input files, thus we generally recommend including seedlings at this stage. The functionality to include inventory measured seedlings was recently added to BioSum after research determined that seedlings grown into saplings and trees in FVS could significantly affect stand density, competition, and flammability over longer simulations (e.g., >10yrs).

e. Down Wood Data

Down wood data from FIA plots, which includes coarse and fine wood loadings assessed from measured pieces of down wood on transects, duff and litter loadings developed from duff and litter depth measurements and field crew-assessed surface fuelbed category (also referred to as a fuel model) can be loaded to a project along with the plot, condition and tree data. If loaded into the project, it can be used later when building FVS input files, for example, to provide better information on surface fuels to be accounted for by FFE-FVS when estimating potential fire characteristics or accounting for woody carbon that does not reside in standing trees. Checking the "Use Down Woody Materials Data" box will load this data, if available in the FIADB database, into the project's \db\master_aux.accdb database as the tables DWM_COARSE_WOODY_DEBRIS, DWM_FINE_WOODY_DEBRIS, DWM_DUFF_LITTER_FUEL and DWM_TRANSECT_SEGMENT and into the dwm_fuelbed_typcd column of the COND table in the project's \db\master.accdb database. These data must be loaded if they are to be used in building the FVS input files. There is no reason not to load these data when building the project unless there is no possibility that these data will be needed in FVS, disk space is so limited that there is no room for the master_aux.accdb file if populated, or glitches are encountered when loading down wood.

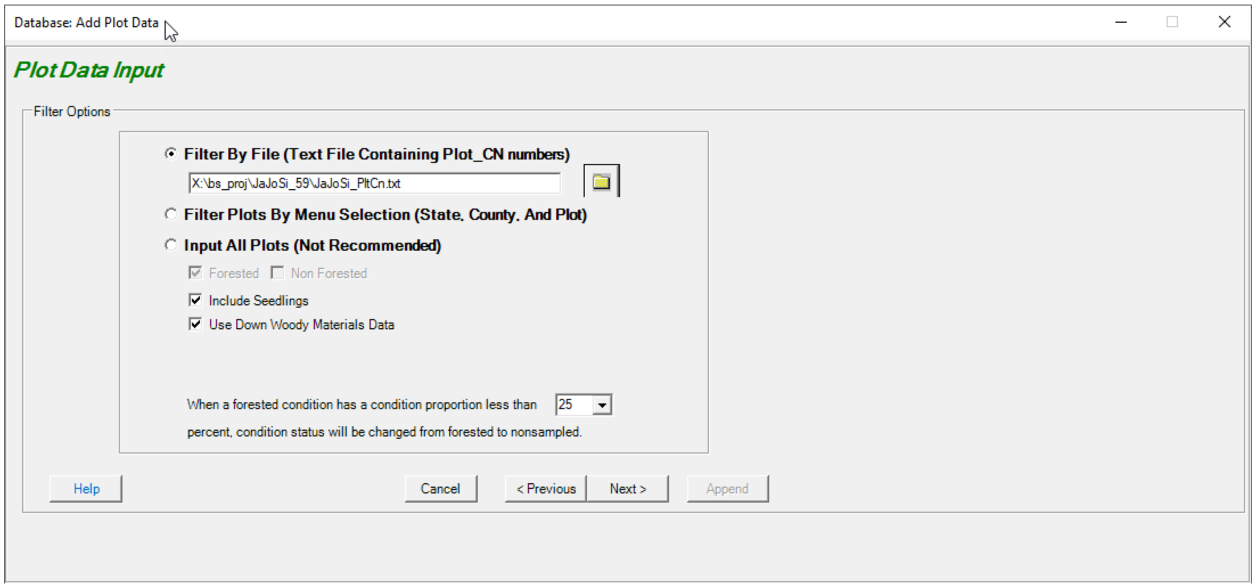

Database: Add Plot Data

Plot Data Input

Filter By State And County

	State	County
✓	41	1
✓	41	3
✓	41	5
☐	41	7
✓	41	9
☐	41	11
✓	41	13
☐	41	15
☐	41	17
☐	41	19
☐	41	21
☐	41	23
☐	41	25
☐	41	27
☐	41	29

Figure 2.10 – Filter by county.


Database: Add Plot Data

Plot Data Input

Filter Options

(●) Filter By File (Text File Containing Plot_CN numbers)

X:\bs_proj\JaJoSi_59\JaJoSi_PltCn.txt

( ) Filter Plots By Menu Selection (State, County, And Plot)

( ) Input All Plots (Not Recommended)

[✓] Forested [ ] Non Forested

[✓] Include Seedlings

[✓] Use Down Woody Materials Data

When a forested condition has a condition proportion less than 25 percent, condition status will be changed from forested to nonsampled.

Figure 2.11 – Filter CN using a text file.

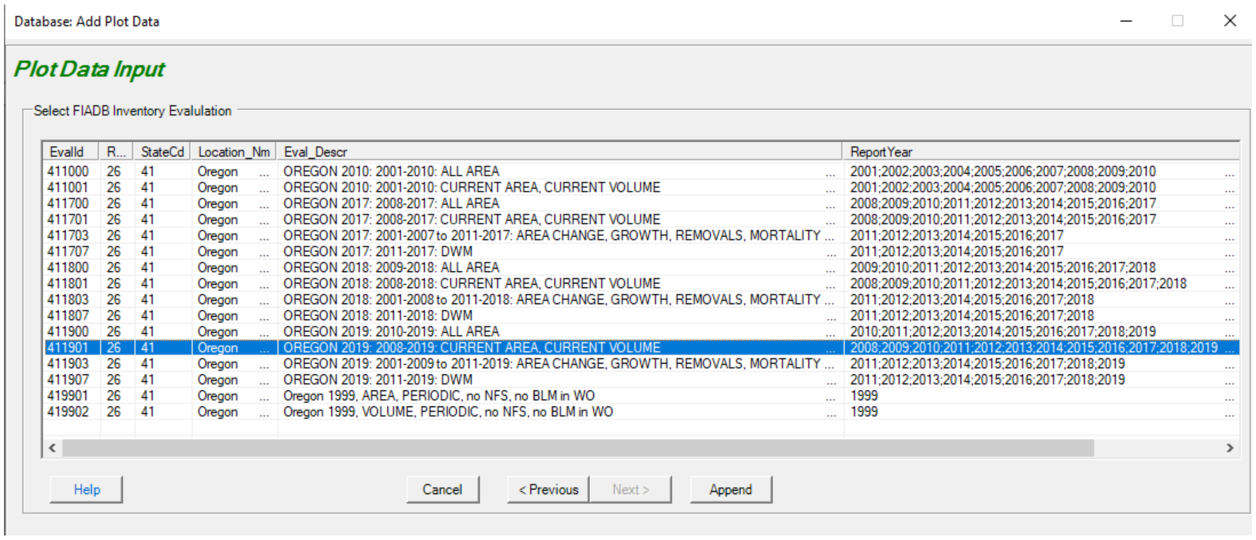
11. Choose an "EvalID" (Evaluation Identifier), which defines a set of FIA plots to represent a state, or portion thereof (if the 2nd or 3rd bullets are selected in the previous, filtering step), for a given time period and purpose, to load into the BioSum project. The status at the time an attempt is made to sample these plots in the field, and the status of conditions that comprise them (e.g., forested, non-sampled, etc.), and the post stratification data encoded in the associated "POP" (population) tables (e.g., POP_EVAL, POP_STRATUM) in the FIADB database, are analyzed to determine how many acres of forested landscape in that state should be attributed to each condition (stand, in FVS parlance).

    ONLY ONE EVALID SHOULD BE USED TO LOAD PLOTS FOR ANY PARTICULAR STATE. Plots can be loaded in sets, if desired. For example, a project encompassing Oregon's Multnomah, Clackamas and Marion counties could be loaded and, if it was later decided to add plots from Hood River county, that could be done via the same Plot Load process, as long as the same EVALID is selected at load time. If needing to load additional plots from another state, say, for example, Clark county, Washington, those could be loaded in another Plot Load operation, and a single EVALID from the Washington FIADB dataset can be specified to load those plots (and any other plots loaded from Washington at that or any future time). Typically, one would choose EVALIDs representing the same period of time for all the states loaded into a BioSum project, if only for ease of interpretation of BioSum results drawn from a multi-state analysis.

    The contents of POP tables in FIADB state databases changes over time as new Eval ID's are added and existing one retired. For BioSum, look for an Eval Description that references the time period of interest (for example, the most recent 10 years of available data, or the first ten years of data collected under the annual inventory protocols). Analysts are typically interested in the EVALID description that contains the phrase "Current Area, Current Volume", or in older EVALID tables, "Sampled Plots" – for example, the yellow highlighted row in Figure 2.12. These always end in "01". In this example, the "41" references the FIPS state code for Oregon, the "16" signifies it as an EVALID constructed in 2016, and the "01" indicates it is "Current Area, Current Volume".

    Loading plots associated with more than one EVALID (per state) generates potentially intractable problems in calculating valid plot expansions and in creating results tables that depend on managed forest area. The acres assigned to each FIA condition in BioSum depends on the EVALID selected because that determines the number of plots used to represent all the forest area in a state. For example, if some plots are loaded with an EVALID ending in "03" (a Growth, Removals and Mortality Evaluation -- GRM), more acres will be assigned to each condition in plots loaded with that EVALID than to those loaded using a EVALID for the same time period that ends in "01". This is because a GRM evaluation can only use plots that are sampled at both ends of a remeasurement interval-- denied access issues will affect more plots in a GRM Evaluation. When building statistical tables at the end of the BioSum analysis, plots from different evaluations cannot be combined. Moreover, there is risk of loading a plot twice (for different Evaluations), which will corrupt the BioSum project (which is completely intolerant of duplicate plot data).

For more information, please contact an FIA analyst in your region for advice before building the project. If building a project for the most recent ~10 years of data (2008-2019) available for Oregon, select EvalId 411901, then click **<Next>**.



EvalId	R...	StateCd	Location_Nm	Eval_Descr	ReportYear
411000	26	41	Oregon ...	OREGON 2010: 2001-2010: ALL AREA ...	2001;2002;2003;2004;2005;2006;2007;2008;2009;2010 ...
411001	26	41	Oregon ...	OREGON 2010: 2001-2010: CURRENT AREA, CURRENT VOLUME ...	2001;2002;2003;2004;2005;2006;2007;2008;2009;2010 ...
411700	26	41	Oregon ...	OREGON 2017: 2008-2017: ALL AREA ...	2008;2009;2010;2011;2012;2013;2014;2015;2016;2017 ...
411701	26	41	Oregon ...	OREGON 2017: 2008-2017: CURRENT AREA, CURRENT VOLUME ...	2008;2009;2010;2011;2012;2013;2014;2015;2016;2017 ...
411703	26	41	Oregon ...	OREGON 2017: 2001-2007 to 2011-2017: AREA CHANGE, GROWTH, REMOVALS, MORTALITY ...	2011;2012;2013;2014;2015;2016;2017 ...
411707	26	41	Oregon ...	OREGON 2017: 2011-2017: DWM ...	2011;2012;2013;2014;2015;2016;2017 ...
411800	26	41	Oregon ...	OREGON 2018: 2009-2018: ALL AREA ...	2009;2010;2011;2012;2013;2014;2015;2016;2017;2018 ...
411801	26	41	Oregon ...	OREGON 2018: 2008-2018: CURRENT AREA, CURRENT VOLUME ...	2008;2009;2010;2011;2012;2013;2014;2015;2016;2017;2018 ...
411803	26	41	Oregon ...	OREGON 2018: 2001-2008 to 2011-2018: AREA CHANGE, GROWTH, REMOVALS, MORTALITY ...	2011;2012;2013;2014;2015;2016;2017;2018 ...
411807	26	41	Oregon ...	OREGON 2018: 2011-2018: DWM ...	2011;2012;2013;2014;2015;2016;2017;2018 ...
411900	26	41	Oregon ...	OREGON 2019: 2010-2019: ALL AREA ...	2010;2011;2012;2013;2014;2015;2016;2017;2018;2019 ...
411901	26	41	Oregon ...	OREGON 2019: 2008-2019: CURRENT AREA, CURRENT VOLUME ...	2008;2009;2010;2011;2012;2013;2014;2015;2016;2017;2018;2019 ...
411903	26	41	Oregon ...	OREGON 2019: 2001-2009 to 2011-2019: AREA CHANGE, GROWTH, REMOVALS, MORTALITY ...	2011;2012;2013;2014;2015;2016;2017;2018;2019 ...
411907	26	41	Oregon ...	OREGON 2019: 2011-2019: DWM ...	2011;2012;2013;2014;2015;2016;2017;2018;2019 ...
419901	26	41	Oregon ...	Oregon 1999, AREA, PERIODIC, no NFS, no BLM in WO ...	1999 ...
419902	26	41	Oregon ...	Oregon 1999, VOLUME, PERIODIC, no NFS, no BLM in WO ...	1999 ...

Figure 2.12 – Select an Eval ID for loading plots to the project.

BioSum will report progress as data loads, and when finished, will display a dialog showing the number of plots, conditions and trees loaded. After clearing that dialog, exit the plot data input window via <**cancel**> or the close box in the upper right window corner (“X”).

Deleting Data

Previous versions of BioSum contained a “delete” button for removing plot data. Its purpose was to revise the plot set in a BioSum project BEFORE generating FVS input files, and for this reason, it only deleted data in the *projectname*\db\master.mdb that had been loaded from FIADB. However, for projects that had moved further along the workflow path into FVS inputs and outputs, processor and optimizer, such deletions could eventually lead to data corruption as tables downstream must sometimes link to data in master, and when that data no longer exists, problems surface! Thus, the plot deletion function has been removed and replaced with two “zappers” that delete data more consistently and completely from dozens or even hundreds of database tables where it resides, reducing the chances of data corruption while still allowing some condensation of BioSum projects that can otherwise become quite large while containing data that is no longer needed or wanted.

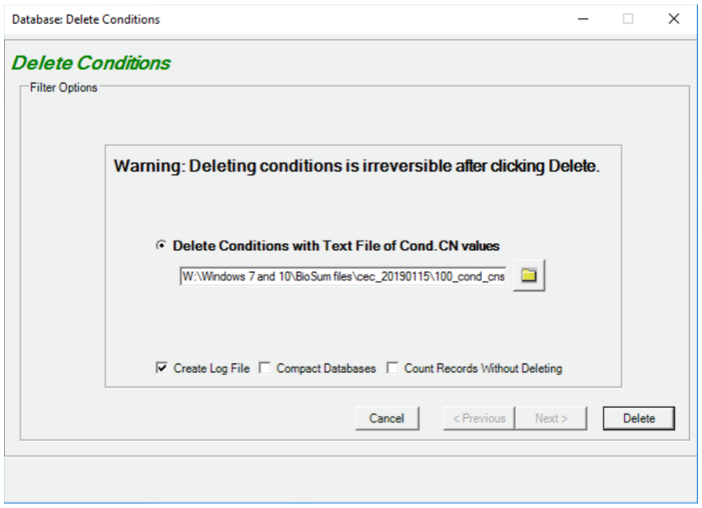
One zapper deletes conditions no longer needed in a BioSum project. This can be useful for subsetting a biosum project—for example, from a whole state dataset to a subregion within the state, perhaps to hand off to another user, and for thinning out a project to remove cases not needed for analysis—for example, conditions that are non-forest, in reserved areas where treatment need not be modeled, or for which no contemplated treatments can be implemented (e.g., if too little tree stocking).

The other zapper deletes unneeded silvicultural sequences (packages). This can be useful for removing, from all tables in the project databases, any silvicultural sequences that seemed like a good idea when developed, but which have been superseded by others or no longer make sense to keep.

The importance of backing up project data cannot be overstated, and if a deletion is contemplated, that is an especially good time to create a backup, given that deletions are irreversible. When major changes are needed in the FIA data to be used as the basis of a project that has already been analyzed with BioSum, it is generally safer to create a new project into which appropriate plot data can be loaded.

Delete Conditions

The Delete Conditions button opens a *Delete Conditions* user dialog for specifying the name of the file containing the list of COND.CN values (control number [CN]) of the records in the COND table for which data removal is desired.


Database: Delete Conditions

Delete Conditions

Filter Options

Warning: Deleting conditions is irreversible after clicking Delete.

(●) Delete Conditions with Text File of Cond.CN values

W:\Windows 7 and 10\BioSum files\cec_20190115\100_cond_cns

[✓] Create Log File [ ] Compact Databases [ ] Count Records Without Deleting

Figure 2.13 – Specify the filename containing the COND.CN values for the conditions to be deleted.

This list of COND.CN values is typically generated via queries in project\db\master.mdb and may include, for example, conditions with COND.RESERVCD=1, if removing wilderness areas and parks from potential management were the objective, or with COND.FORTYPCD<>371 if only conditions with a mixed conifer forest type were intended to be included in the BioSum analysis.

However this text file is produced, it is essential that 1) it contain the CN values found in the COND table (i.e., condition CNs not plot CNs), 2) that there be no delimeters (such as quotes) on this text attribute and no header line at the top of the file—just the text strings that make up the CN, one per line.

Special care must be taken with any lengthy text attribute when working with such data in applications or computing environments such as R or Excel as these have a tendency to convert text data that LOOKS like a number into a numeric data type which, for long numeral strings, often truncates some of the digits, corrupting the data and breaking your analysis workflow in BioSum.

The contents of a file like Conds2Drop.txt specified on the "Delete Conditions…" line might start something like this (no header, no delimiters):

```
447734010497
447735010497
447736010497
447737010497
15430368010497
```

CAUTION: there are several different identifiers for conditions in FIADB, BioSum and FVS and some users have become confused as to which one to use. For CONDITION DELETION, it needs to be the CN value found in the COND table (towards the far right), NOT the biosum_cond_id, or condid which are completely different identifiers used for different purposes.

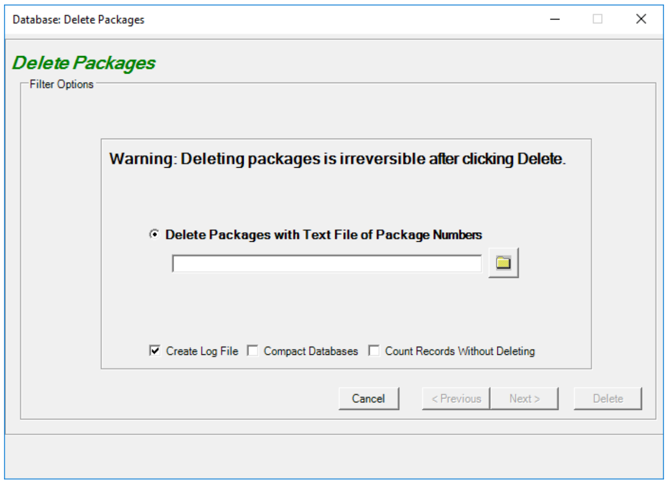
Three checkboxes offer access to options for 1) creating a log file listing the BioSum project tables searched for data associated with the specified condition records and the number of records deleted from each, 2) compacting each project database after deletions are accomplished to conserve disk space, and 3) for conducting an inspection and reporting of record counts associated with the targeted conditions, without actually performing the deletions.

Caution should be exercised when enabling the "compact databases" option because limitations of MS Access result in this step failing intermittently, such that some of the project databases do not successfully compact.

To proceed (irrevocably) with the deletion, select **<Delete>**. Caution: This function will delete ALL records associated with the submitted condition list, throughout the BioSum project directory structure, i.e., in ALL tables in ALL databases in ALL folders within *projectname*, regardless of table or database name or whether or not the database and table contain data used in the project—thus, even "backup" versions of databases or tables that a user has located within the *projectname* directory structure will have these conditions deleted by the end of the condition deletion operation.

Delete Packages

The Delete Packages button opens a *Delete Packages* user dialog for specifying the name of the file containing the list of package labels (these look like 3 digit numbers but are actually text, so it is important to include leading zeros, as in 001, not just 1 for package 1) of the packages for which data removal is desired. All rows in all tables that contain a package identifier matching one of the lines in the package label list provided to BioSum will be deleted.


Database: Delete Packages

Delete Packages

Filter Options

Warning: Deleting packages is irreversible after clicking Delete.

(●) Delete Packages with Text File of Package Numbers

[✓] Create Log File [ ] Compact Databases [ ] Count Records Without Deleting

Figure 2.14 – Specify the filename containing the Package labels for all packages to be deleted.

As with the condition zapper, three checkboxes offer access to options for 1) creating a log file listing the BioSum project tables searched for data associated with the records associated with the specified packages and the number of records deleted from each, 2) compacting each project database after deletions are accomplished to conserve disk space, and 3) for conducting an inspection and reporting of record counts associated with the targeted packages, without actually performing the deletions.

Caution should be exercised when enabling the “compact databases” option because limitations of MS Access result in this step failing intermittently, such that some of the project databases do not successfully compact.

To proceed (irrevocably) with the deletion, select **<Delete>**. This function will delete ALL records associated with the submitted package list, throughout the BioSum project directory structure, i.e., in ALL tables in ALL databases in ALL folders within *projectname*, regardless of table or database name or whether or not the database and table contain data used in the project—thus, even “backup” versions of databases or tables that a user has located within the *projectname* directory structure will have records associated with these packages deleted by the end of the package deletion operation.

Wood Processing Sites

The following description of workflow for populating the processing sites table has largely been superseded by a button in the Optimizer module as of version 5.88, when substantial improvements were made in travel time data management by providing a database of all wood processing facilities known as of 2019 and pre-calculating the travel times between them and all FIA plots in the BioSum covered states. Loading the travel times from these pre-calculated values is a cinch via a “Load GIS Data” button now implemented in the Optimizer main menu.

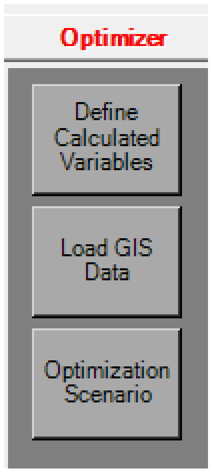


Figure 2.15- The Load GIS Data button in the Optimizer module can now replace the formerly manual assembly of a processing site table

To make use of this function, first download the file gis_travel_times_master.accdb (a zip compressed file which will need to be unzipped) from http://biosum.info/downloads/geoprocessing/gis_travel_times_master.zip, and place it in the hidden “C:\Users\<user_name>\AppData\Roaming\FIABiosum” folder. This database contains both psites and traveltimes tables and the data therein can be loaded into BioSum using the button. GIS yarding distance is also pre-calculated and loadable from the biosum_ref database (contained in the same hidden folder at gis_travel_times_master, described above) in the Database module whenever new plots are loaded from FIADB to a BioSum project.

Superseded Workflow

For those wanting to create their own wood processing facility list, or edit an existing one (e.g., one loaded via the “Load GIS Data” button) or to obtain a better understanding of how wood processing sites are represented, can refer to the following workflow documentation.

The Wood Processing Sites task allows entry and editing of wood processing sites-- the locations where harvested wood is delivered for conversion into products or bioenergy. Wood processing sites can be attributed as accepting logs, wood chips, or both and as final delivery terminals or as railheads where wood can be collected and forwarded by rail to a final delivery terminal. Wood processing sites can account for facilities that already exist, or that are under consideration to be built; either way, individual facilities can be toggled on or off during analysis.

1. Select the **<Wood Processing Sites>** button (see Figure 2.16).

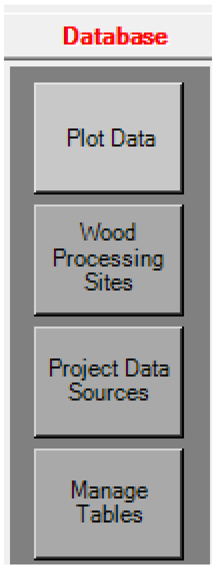
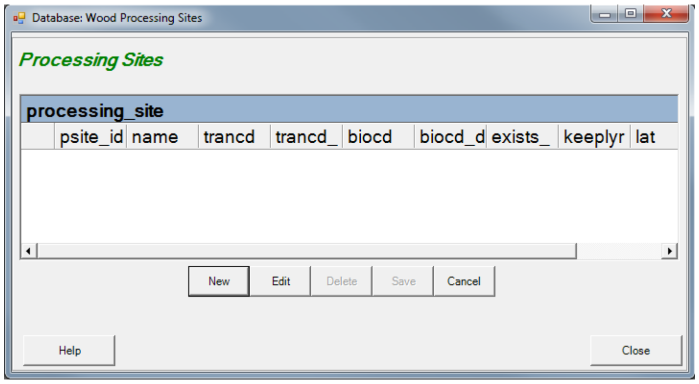


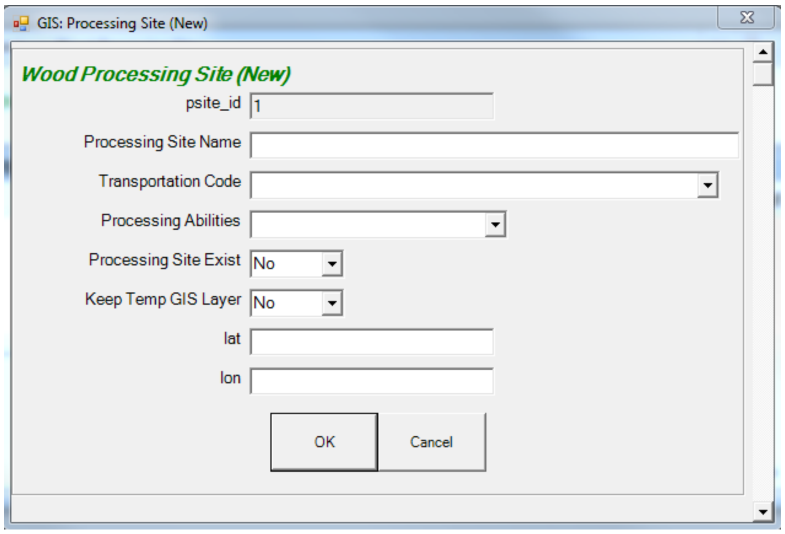
Figure 2.16- Wood Processing Sites button

2. To create a new wood processing site, click on **<New>** (see Figure 2.17). Enter the information on the new wood processing site, including the processing site name, the transportation code (road access, railroad access, or both), the material processed by the site (logs, chips, or both), whether the site currently exists, and facility location (latitude and longitude) (see Figure 2.18). Once a new site has been created, be sure to click **<Save>** before exiting the Processing Sites window shown in Figure 2.17.



	psite_id	name	trancd	trancd_	biocd	biocd_d	exists_	keeplyr	lat

Figure 2.17- Wood Processing Sites dialog

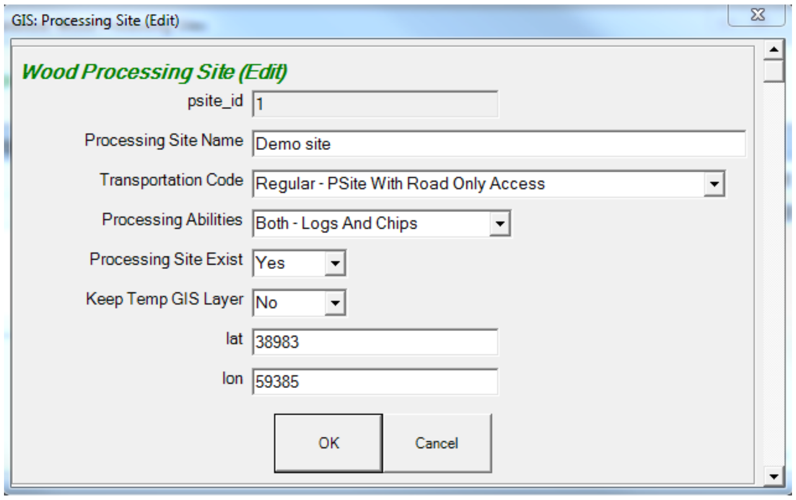


Wood Processing Site (New)

psite_id	1
Processing Site Name	
Transportation Code	
Processing Abilities	
Processing Site Exist	No
Keep Temp GIS Layer	No
lat	
lon	

Figure 2.18- Creating a new wood processing site

3. To edit a site, select the <**Edit**> button shown in Figure 2.17 and make changes to each field as needed (see Figure 2.19). Be sure to click <**Save**> before exiting the Processing Sites window to save any edits made.
4. Wood processing sites can also be deleted with this tool by selecting a site then clicking <**Delete**> (also in Figure 2.17), selecting <**Yes**> when the Delete Selected Processing Sites window pops up, then click <**Save**> to save the change.



Wood Processing Site (Edit)

psite_id	1
Processing Site Name	Demo site
Transportation Code	Regular - PSite With Road Only Access
Processing Abilities	Both - Logs And Chips
Processing Site Exist	Yes
Keep Temp GIS Layer	No
lat	38983
lon	59385

Figure 2.19- Editing a wood processing site

   a. Alternatively, site information can be populated or modified by directly manipulating the Access table processing_site within the gis_travel_times.mdb database which typically resides in the gis directory within the BioSum project.

Project Data Sources

1. To view the directory paths and names of the project database files and tables, determine whether or not the file and table names exist, and view table record counts click the

<Project Data Sources> button from the left-hand menu shown in Figure 2.20. This tool can also be used to change the information source for particular tables in a project (i.e., to point to a different database or table).

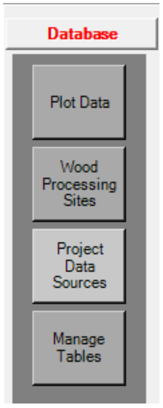
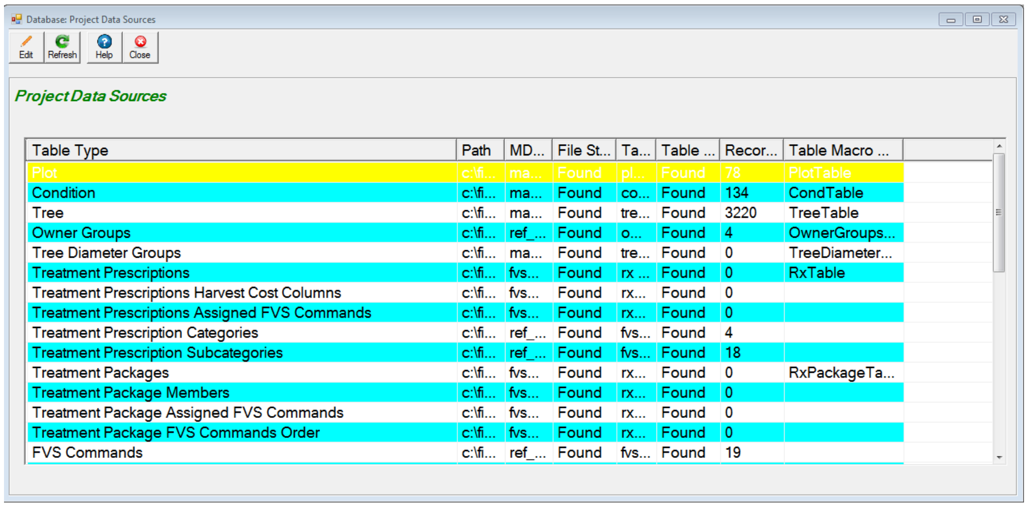


Figure 2.20 – Project data sources.

2. Each table has a unique identifier, or '*Table Type*.' The '*Path*' column displays the directory path for the corresponding MDB File. The '*MDB File*' column contains the file name for the Microsoft Access database. '*File Status*' indicates whether or not a file is found, while '*Table Name*' contains the name of the table. The '*Table Status*' column indicates whether the table is found in the MDB file. '*Record Count*' shows the number of records in the table. All columns may be updated by clicking on the **<Refresh>** button. Some of the 37 table types maintained as input data for a BioSum project are shown in Figure 2-21 and Table 2.1. Other BioSum tables, of both inputs and outputs, associated with the Processor and Core Analysis task groups, are not contained in the databases referenced here (those located within the db, fvs and gis subdirectories of a BioSum project).



Project Data Sources

Table Type	Path	MD...	File St...	Ta...	Table ...	Recor...	Table Macro ...
Plot	c:\fi...	ma...	Found	pl...	Found	78	PlotTable
Condition	c:\fi...	ma...	Found	co...	Found	134	CondTable
Tree	c:\fi...	ma...	Found	tre...	Found	3220	TreeTable
Owner Groups	c:\fi...	ref_...	Found	o...	Found	4	OwnerGroups...
Tree Diameter Groups	c:\fi...	ma...	Found	tre...	Found	0	TreeDiameter...
Treatment Prescriptions	c:\fi...	fvs...	Found	rx ...	Found	0	RxTable
Treatment Prescriptions Harvest Cost Columns	c:\fi...	fvs...	Found	rx...	Found	0	
Treatment Prescriptions Assigned FVS Commands	c:\fi...	fvs...	Found	rx...	Found	0	
Treatment Prescription Categories	c:\fi...	ref_...	Found	fvs...	Found	4	
Treatment Prescription Subcategories	c:\fi...	ref_...	Found	fvs...	Found	18	
Treatment Packages	c:\fi...	fvs...	Found	rx...	Found	0	RxPackageTa...
Treatment Package Members	c:\fi...	fvs...	Found	rx...	Found	0	
Treatment Package Assigned FVS Commands	c:\fi...	fvs...	Found	rx...	Found	0	
Treatment Package FVS Commands Order	c:\fi...	fvs...	Found	rx...	Found	0	
FVS Commands	c:\fi...	ref_...	Found	fvs...	Found	19	

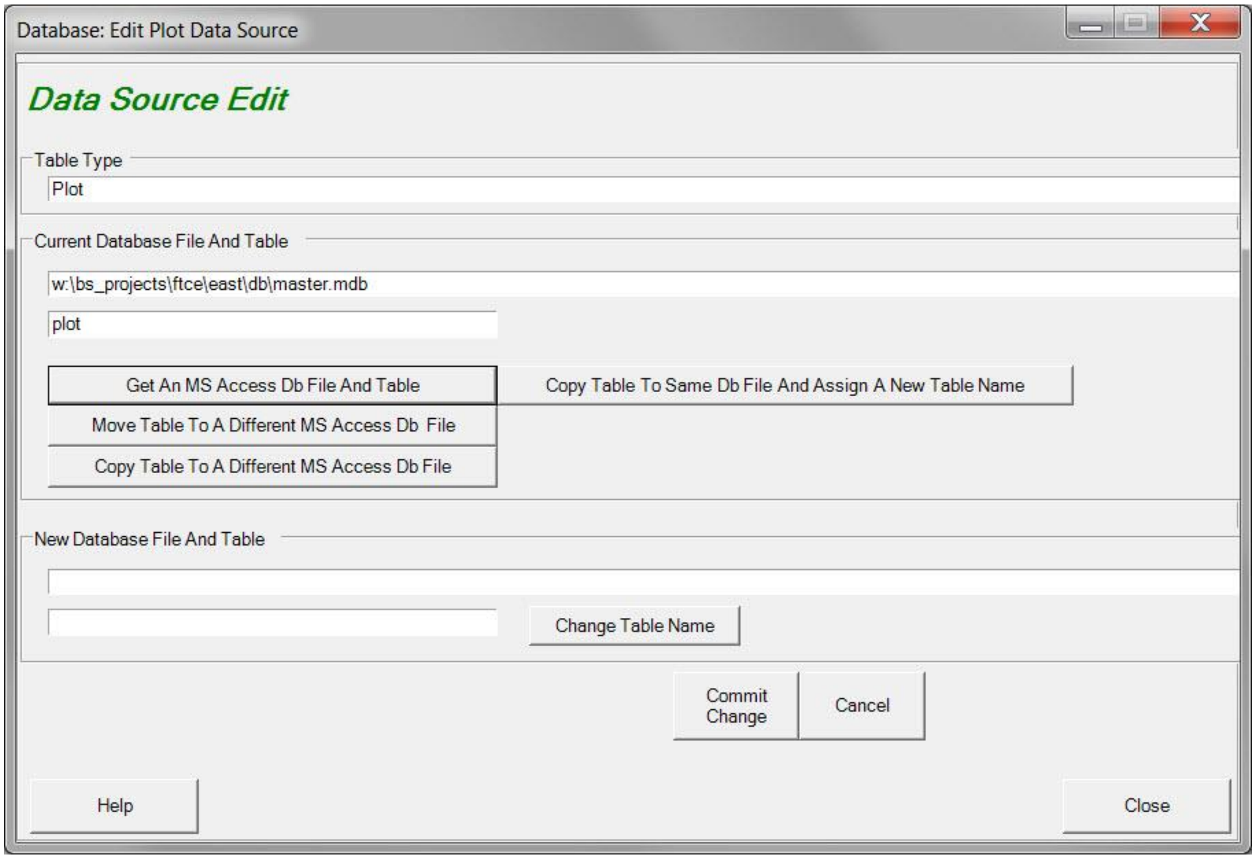
Figure 2.21- Project data source window

Table 2.1 – Summary of table types in project data sources.

Table Type	Description
Plot	Plot records.
Condition (aka stand)	Condition records.
Tree	Tree records.
Fire And Fuel Effects	Pre- and Post-treatment fire and fuel effects data generated by FVS.
Harvest Costs	Harvest cost data generated by OpCost.
Owner Groups	FIADB owner group values and descriptions.
Tree Diameter Groups	User defined groupings of tree species and diameters.
Treatment Prescriptions	User defined treatment labels. Treatments are assigned an A-Z character value.
Tree Species Groups	User defined tree species group number and tree species group label.
Tree Species Groups List	User defined tree species group assignments. Lists the tree species under each group number.
Tree Species	Lists tree species by FIA species code, FVS species code, FVS variant, FVS conversion value, group assignment, oven-dry weight conversion to dry to green weight ratios.
FVS Tree Species	Conversion matrix of FIA tree species to FVS tree species by FVS variant.
Tree Volumes And Values By Species And Diameter Groups	Processor summed output of tree volumes and values by the groupings listed in both the ‘Tree Diameter Groups’ table and the ‘Tree Species Groups’ table.
Travel Times	Travel times between plots and processing sites on the road network and between railheads and processing sites on the rail network.
Processing Site	Wood processing site facilities.
FVS Tree List For Processor	FVS tree cutlist output that has been formatted by the FIA BioSum Manager to be used as input into the Processor.
FIADB FVS Variant	User defined plot and FVS variant assignments for FIADB records. NOTE: Not used for PNW IDB because records already have FVS variant assignments.
Tree Species And Diameter Groups Dollar Values	Dollar value per unit volume for the tree diameter and species group assignments.

3. Data sources can also be edited using this tool, giving the user the ability to manage tables that are used in a BioSum project. To edit a data source, select a table type and click the <Edit> button on the top left of the screen as shown in Figure 2.21. This tool displays where the selected table is currently stored and allows the user to move the selected table to a differed mdb file, copy the selected table to a different mdb file, or make a copy of the table within the same mdb file, assigning a new name to the copy (see Figure 2.22). One useful feature of this tool is that it

allows the user to overcome the MS ACCESS 2GB file size limit by allowing a table to be copied to another MDB. Once the desired changes are made, select <Commit Change> then close the window with <Close> or the close control <x>.


Database: Edit Plot Data Source

Data Source Edit

Table Type: Plot

Current Database File And Table: w:\bs_projects\ftce\east\db\master.mdb, plot

Get An MS Access Db File And Table
Copy Table To Same Db File And Assign A New Table Name
Move Table To A Different MS Access Db File
Copy Table To A Different MS Access Db File
New Database File And Table
Change Table Name
Commit Change
Cancel
Help
Close

Figure 2.22- Editing data sources

Manage Tables

The **Manage Tables** task (see Figure 2.23) lists the project MDB files in a tree directory format, the tables in a MDB file, and the columns in a table. This tree directory can be utilized to locate and inspect all databases and tables in a BioSum project (see Figure 2.24). When a MDB file is selected, it can be compacted by clicking on the **<Compact>** button, or opened in Access by clicking on the **<Open in Access>** button. Opening the MDB file in Microsoft Access allows the user to view the tables in an MDB file, the data within each table, and make edits to the MDB file and associated data (though this is not generally advised). The user may also select a table and columns of their choice, and click on **<Browse>** to view the records. Once columns in a table are selected (boxes next to the column names are checked), the <**Browse**> button allows users to review the data populating those columns in the selected table, without risk of inadvertently introducing unwanted edits.

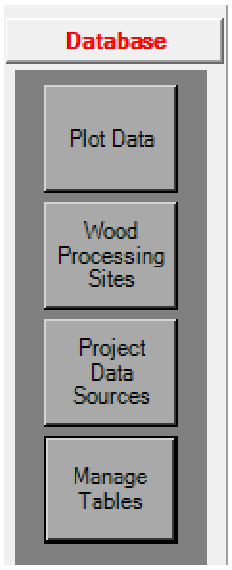
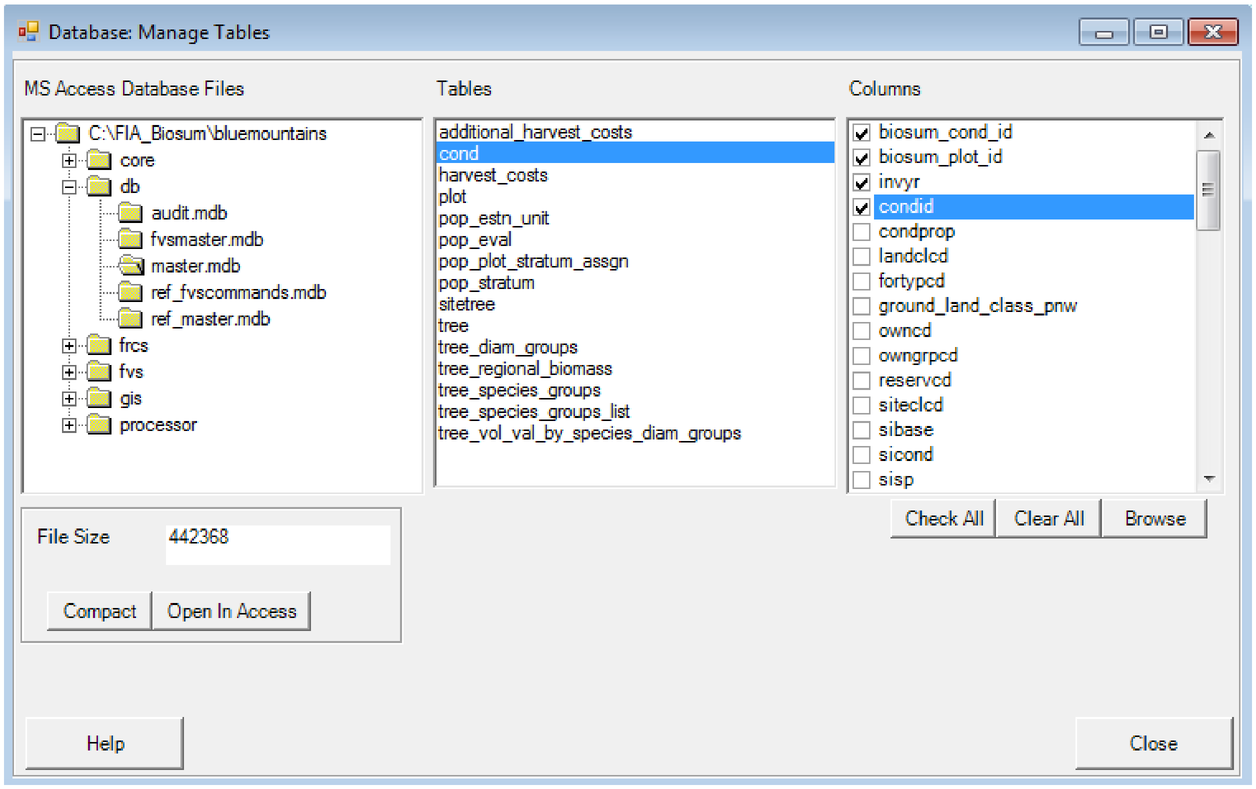


Figure 2.23- Manage Tables.



C:\FIA_Biosum\bluemountains
- core
- db
  - audit.mdb
  - fvsmaster.mdb
  - master.mdb
  - ref_fvscommands.mdb
  - ref_master.mdb
- frcs
- fvs
- gis
- processor

Tables: additional_harvest_costs, cond, harvest_costs, plot, pop_estn_unit, pop_eval, pop_plot_stratum_assgn, pop_stratum, sitetree, tree, tree_diam_groups, tree_regional_biomass, tree_species_groups, tree_species_groups_list, tree_vol_val_by_species_diam_groups

Columns: ☑ biosum_cond_id, ☑ biosum_plot_id, ☑ invyr, ☑ condid, ☐ condprop, ☐ landclcd, ☐ fortypcd, ☐ ground_land_class_pnw, ☐ owncd, ☐ owngrpcd, ☐ reservcd, ☐ siteclcd, ☐ sibase, ☐ sicond, ☐ sisp

File Size 442368

Figure 2.24– Manage Tables dialog.

Behind the Scenes

BioSum conducts a number of housekeeping chores and data preparation when it loads FIADB data into a project. This section describes some of the behind the scenes operations involved loading a project with data, based on questions that have been asked by users. As more questions are asked, it will become more complete.

Key variables are those that are unique for every row in a table. These key variables are essential when linking tables in a project, both by the BioSum software and when conducting diagnostics or putting tables together to tell a story. One variable that is key on the master.cond table, and which serves as a foreign key in many other tables, is biosum_cond_id. This same key variable is named stand_ID in FVS input and output tables. It is constructed from FIADB identifier attributes such as state, cycle, subcycle, plot number, and condition number, as follows:

BIOSUM_COND_ID			
Description: unique condition id field. Used as the primary key for the cond table as well as the primary key for the FVS STAND_ID field.			
Field Type: Character			
Length: 25			
Column	**Length**	**Description**	**Values**
01	**01**	**Inventory Id: Obsolete**	**Now always "1"**
02	**04**	**Year plot measured**	**4-digit year**
06	**02**	**State code**	**2-digit numeric**
08	**02**	**Nims Cycle**	**2-digit numeric**
10	**02**	**Nims Subcycle**	**2-digit numeric**
12	**03**	**County Code**	**3-digit numeric**
15	**07**	**Plot number (FIADB public plot number)**	**5-digit numeric**
22	**03**	**Obsolete**	**Now always "000"**
25	**01**	**Condition Number**	**1 to 5**

Chapter 3. Travel Time Geoprocessing

Caution: as of 29 January 2020, this version of chapter 3 is not necessarily glitch-free with respect to formatting, numbering of steps and referencing of figure and step numbers in the text

The purpose of geoprocessing in BioSum is to determine, for each plot, 1) the merchantable wood processing site that is cheapest to haul to, the energy wood facility that is cheapest to haul to, and the unit cost of hauling wood to both of those sites; and 2) the distance from the plot to the nearest point on the road network—a parameter that is important for estimating treatment costs.

This longest chapter in the User Guide has been rewritten several times over 15 years as technology and procedures have evolved and improved. Early versions made use of AMLs (ArcInfo macro language scripts), VBA (Visual Basic for Applications) scripts, python scripts and various GUIs for semi-automated processing. Unfortunately, as ESRI evolved their GIS software offerings, these investments were orphaned as software versions changed underlying architecture and scripting languages became unsupported. Several years ago, we decided not to re-attempt automation of this workflow and to instead provide detailed, step-by-step instructions that could be followed by a GIS analyst familiar with ArcGIS. Of course, these, too, will speed towards obsolescence as software continues to evolve and interfaces and GIS workflows are changed. For that reason, and to reduce barriers to getting started with analysis using BioSum, in 2019 we set out to conduct the geoprocessing for several western states so that you don't have to.

The **good news** then is that perhaps you won't need this chapter at all (or at least you may be well-positioned to skip most of it). For nine western states (Oregon, Washington, Idaho, Montana, California, Nevada, Utah, Wyoming, and Colorado[1]), the full BioSum software distribution package, when supplemented with a reference database downloaded from the spatial data page at biosum.info, provides the location-related outputs, created via the geoprocessing workflow described in this chapter, that you will need to perform a BioSum analysis. These include:

1. **Yarding distance** between plots in these nine states, plus Alaska and New Mexico, and the nearest point with road access (assumed landing site for transferring logs onto trucks), which is provided in the fiadb_fvs_variant table in the biosum_ref.accdb database (located, after the first time the BioSum program is executed, in the protected directory C:\Users\*[user_name]*\AppData\Roaming\FIABiosum).
2. A **processing_site** table, contained within the database gis_travel_times_master.accdb, lists all known current, former and planned mill and bioenergy facility locations. The facility list is believed complete for these nine states as of March, 2019, but users may want to update it as it ages, and new facilities are established and existing ones shuttered. The gis_travel_times_master.zip file can be downloaded from

[1] Caution: while spatial data support is provided for all 9 states, FVS model support (e.g., with respect to site index species and volume and biomass equations, is currently incomplete except for in California, Oregon, Washington, Idaho and most of Montana).

http://biosum.info/downloads/geoprocessing/gis_travel_times_master.zip; then the .accdb database file can be extracted using a utility such as 7zip.

3. A **plot_gis** table containing, for every plot in the 9 states that has been visited at least once by 2018, the approximate coordinates in lat-lon and in an albers equal area projection along with the distance derived from the ArcGIS NEAR function from the approximate plot location to the nearest road, in feet and in meters. This distance is already preloaded in the biosum_ref database that is accessed when FIADB data is loaded to a BioSum project, but the complete info resides in this table with gis_travel_times_master.accdb for user reference.
4. **Travel time** estimates from every plot to every wood processing facility location in the processing_site table that is located within 250 km (~156 miles) straight-line distance of the plot; these are stored in the **travel_time** table within gis_travel_times_master.accdb and can be imported to a BioSum project using the Load GIS tool available in BioSum's Optimizer module, provided the data source gis_travel_times_master.accdb is first copied by the user into the C:\users\[username]\AppData\FIABioSum folder (see Chapter 6).

The BioSum web site at www.biosum.info also includes the 9-states road data that were geoprocessed to generate these outputs; users seeking to duplicate this workflow in other states, following the instructions provided in this chapter, may find this road data, and the plot and processing site tables just described (from which a GIS analyst can easily derive GIS point layers), useful as a template. Raster and vector layers of all roads, with speed limit attribute, within these nine states, are posted on the web site. Users will need to combine Forest Service road data with state level road data for the state(s) they are working with into a single geodataset if choosing to conduct vector travel time analysis.

If

1. analyzing a study area that is entirely within this nine-state region, and
2. the facility and road layers (provided with the BioSum distribution) that were used to derive the travel time estimates are deemed acceptable,

then no GIS operations are required! The rest of this chapter may be skipped. Yarding distances will be loaded from a reference table when plots are loaded to BioSum, and the travel times and processing sites (psites) tables can be loaded with a button in the treatment optimizer module.

If opting to use the pre-calculated travel times from the 9 states data, you'll select this option much later in the BioSum workflow, when running optimizer (and before building an optimization scenario). There are three steps:

1) Copy database gis_travel_times_master.accdb, available from http://biosum.info/downloads/spatial_data.html, to C:\Users\<user name>\AppData\Roaming\FIABiosum, substituting your computer login ID for <user name>.

2) Click “Load GIS Data” (Figure 3.1). If the required database has not been copied to the correct location (as described in step 1), or if the database file name is not exactly as specified, an error message will display indicating that this file needs to be placed there.

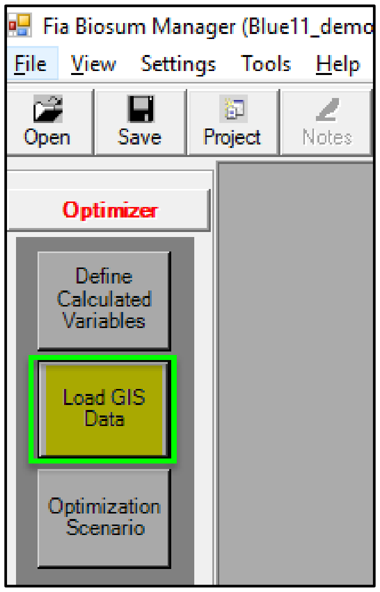
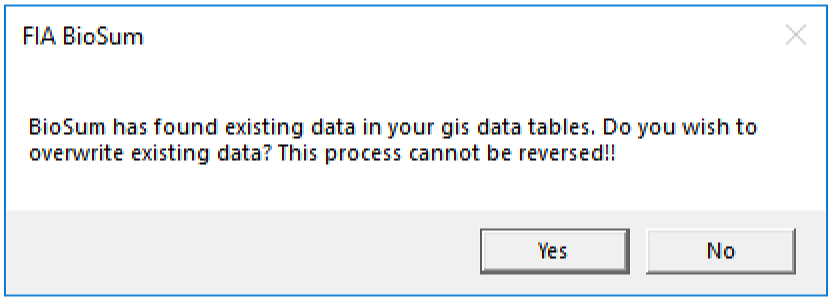


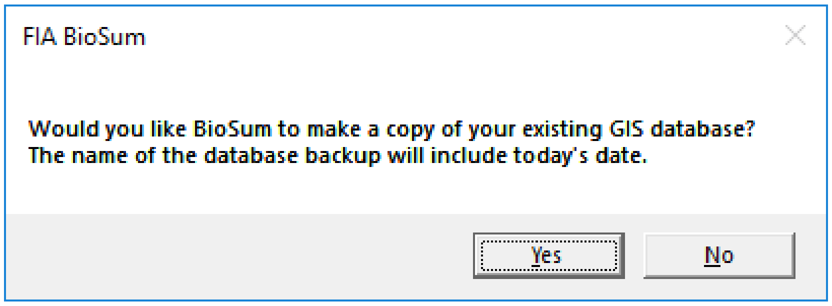
Figure 3-1. Choose to load pre-calculated travel times data via the “Load GIS data” button in Treatment Optimizer before running the first Optimization Scenario.

3) If the database is found, a message is displayed (Figure 3.2), warning that any existing information on travel times (in the <project root>\gis\db\gis_travel_times database) will be irrevocably replaced. For each plot, all travel time records and associated processing site records in gis_travel_times_master that are for processing sites with a travel time from the plot of less than 10.5 hours will be loaded.

   If an optimization scenario has already been defined, complete with a list of enabled processing sites (as determined by ticked check boxes in the first column of the Wood Processing Sites table displayed in the Treatment Scenario), the processing sites list will be updated to the contents of the one in gis_travel_times_master; if site numbering of the two tables is identical, then the enabled site list will not change, but it is worthwhile to double-check this. Select “Yes” to proceed, then choose whether or not you want BioSum to first backup the gis_travel_times database.


FIA BioSum

BioSum has found existing data in your gis data tables. Do you wish to overwrite existing data? This process cannot be reversed!!


FIA BioSum

Would you like BioSum to make a copy of your existing GIS database?
The name of the database backup will include today's date.

Figure 3-2. Warning that existing data on travel times and processing sites will be replaced and option to backup existing travel times database.

Rolling your own…

If criteria 1 and 2 are not met, this chapter provides detailed instructions covering the GIS-based spatial analysis for obtaining yarding distance and travel times, including:

1. Setting up a geoprocessing environment;
2. Creating a point feature class from the plot coordinates;
3. Using GIS to subset the population of plots downloaded for an entire state or set of states to delineate a study area and plot sample within this population;
4. Creating a processing site layer;
5. Preparing transportation layers;
6. “Relocating” plots to the nearest road (this is required for calculating both yarding distance and travel time);
7. Methodology for deriving plot yarding distance;
8. Methodology for calculating travel time (in decimal hours) from relocated plots (assumed sites of landings) to processing facilities via vector or raster transportation models.

Chapter 4. FVS

Overview

This module, the second in a sequence of four workflow modules managed by the BioSum Manager software, connects the FIA forest inventory data imported into BioSum in Module 1 with the Forest Vegetation Simulator (FVS; Dixon 2002) and its Fire and Fuels Extension (FFE; Rebain 2010). These models are developed and maintained by the USDA Forest Service, with detailed documentation and model software available at (https://www.fs.usda.gov/fvs/).

This overview begins with an extensive (9-page) discussion of how FVS modeling fits into the BioSum framework. A full understanding of this is essential to effective use of BioSum and the interaction between FVS and the BioSum framework is not entirely intuitive or obvious, owing to the complexity and flexibility of both components. If already familiar with building simulations in FVS, expect most of this chapter to be accessible on a first read-through. Those who need to learn FVS will likely want to return to this discussion after learning to use FVS using the resources, such as tutorials, available on the FVS web site.

This chapter delves a bit beyond the FVS module because the decisions made in working through this portion of the BioSum workflow (e.g., how prescriptions will be set up, and timing arranged, in the FVS keyword control code that the user must develop to represent management options) will determine what it is possible to accomplish in later stages of the BioSum workflow. Consistent with the bridging role played by this BioSum module (between FIA and FVS frameworks), terminology shifts here towards what is standard in the FVS model; hence, the forested *conditions* loaded in module 1 (Database) become *stands* from the perspective of FVS simulation. An FVS simulation (referred to as a *run* in FVS) can be applied to some or all of the stands (*forested conditions*) loaded in module 1 to assess the effects of a series of one or more silvicultural prescriptions, typically scheduled over four 5- or 10-year "cycles".

These sequences of silvicultural activities are referenced in this Users Guide, interchangeably, as silvicultural sequences or "packages" because they are, in essence, "package choices" involving a pre-specified sequence of what may be independently identifiable "treatments" occurring at different times along the silvicultural sequence. This opens the way for two fundamentally different approaches for representing silvicultural choices and their outcomes in a BioSum analysis, best understood via a simple example.

Suppose one wished to initially implement prescription Rx 100 – mechanical thinning across diameter classes to 75 sq ft residual basal area – treatment on a stand that has become overstocked relative to the size of its trees, followed by a separately described Rx 200 – mastication – treatment 20 years later to tamp down the ladder fuels that develop after opening the canopy. Under a "prescription assembly" paradigm, defining the thinning and mastication activities as separate building blocks, labeled Rx 100 and Rx 200 for this example, these could be assembled into a broad range of packages that could be developed. For example, Packages 1 (P001) could be constructed as Rx 100 (thin) at year 0 and Rx 200 (masticate) at year 20 and P002 as thin at year 10 and masticate at year 30. Mechanical thinning could be implemented in years 20 or 30, but there would not be time in a 4-decade simulation to implement the follow-on mastication 20 years later, so the analyst would need to decide whether to define and model those packages (which would model only the thinning, not the mastication during the simulation timeline). Moreover, Rx 200 might be incorporated into other packages that execute a q-factor prescription and Rx 100 might be combined in a package that refreshes and sustains treatment effectiveness with a broadcast burn, rather than a mastication.

An alternate, “chameleon” approach to the same silvicultural option pairing a thin across diameter classes with mastication 20 years later could define one package (P001) that contains one prescription (Rx 100) that is implemented before every growth cycle (at years 0, 10, 20 and 30 in this example). Typically, this prescription would be crafted to include some conditional trigger for when mechanical thinning takes place (e.g., basal area or stand density index exceeding a pre-specified threshold) that might or might not be true at year 0, but if not, might be in a future decade as the stand develops. Additional accounting variables and triggers could be specified in the prescription that implement the mastication treatment 20 years after the thinning (if still within the simulation timeline). It also opens the possibility of mastication being triggered not by a fixed interval following thinning but by a stand condition, e.g., based on a threshold canopy base height or trees per acre less than 5 inches in diameter, that might trigger the mastication activity to occur at 10, 20 or 30 years following thinning (or never).

There are valid reasons to select either approach to simulating management in BioSum, which will be discussed later, but each has advantages and disadvantages with respect to tracking of activities and timing on specific stands and the number of packages that must be defined (and simulated in FVS) to accommodate the options to be evaluated. BioSum’s modular architecture around packages and treatments makes both approaches viable for maximal flexibility. The user will want to consider how important it is to know, based on the treatment ultimately selected in BioSum as best for a given stand, exactly what happened in that stand and when it happened. While that’s easy under the prescription assembly paradigm, some work is needed in developing the accounting to make that possible under the chameleon paradigm.

Package effects, that is to say, the outcomes of applying the specified sequence of silvicultural activities (which may or may not involve the cutting of trees and in the simplest use case, might be scheduled to occur only in the first BioSum cycle), can be characterized in the Optimizer module. Within the structured workflow around which Optimizer was designed, the stand-level outcomes for each simulated silvicultural sequence can be assessed, and comparisons among sequences evaluated, relative to up to four stand-level descriptors generated by FVS or in the BioSum framework and, if desired, conditional on economic characteristics of the sequence. Examples of stand descriptors include metrics of crown fire potential (e.g, canopy base height) or metrics relating to stand health and habitat value (e.g., measures of stand density or canopy cover, average tree size, or share of trees of preferred species). Examples of economic characteristics include treatment costs and flows and value of wood derived from treatment. Moreover, when more than four attributes need to be considered, the authors have found it useful to construct composite stand indices that consider changes in multiple stand characteristics relative to objectives such as resistance to stand-replacing fire, which depends on both conveyance of fire into crowns, spread of fire among crowns and size and species mediated resistance characteristics of individual trees with respect to, for example, bark thickness (Jain et al 2020).

The four stand attributes selected to represent the effectiveness of management towards one or more goals can be combined via Boolean logic to rate effectiveness as a multi-dimensional accomplishment. An additional attribute can be defined to select the best among multiple effective silvicultural sequences according to rules developed by the user.

In the simplest case (PrePost analysis), effects of treatments implemented at BioSum cycle 1 can be assessed by comparing immediate treatment outcomes to targets (e.g., was canopy base height below 30 feet before treatment and above 30 feet after treatment?) or by considering the magnitude of change (absolute or proportional, or change in classes) between pre- and post-treatment values (e.g., was predicted, severe weather, mortality volume as a percent of live tree volume improved by 25 percent or better?).

A somewhat more nuanced use case (weighted-mean) considers effects of a time-weighted combination of values of such attributes under treatment vs baseline (typically, but not always, grow-only) trajectories from up to 8 time points over the simulation, nominally labeled as Cycle 1 Pre and Post, Cycle 2 Pre and Post, through Cycle 4 Pre and Post. By thoughtful selection of the weights, this weighted mean can be designed to represent a mean value over time, a total, or to select a particular time point for comparison (by setting the weights for the other time points to 0). Stand attributes are estimated or predicted using both standard outputs from and custom-computations generated in FVS and FFE, or, for the adventurous, exogenously outside of the BioSum/FVS workflow via Python or R scripts or database queries. Analysts are strongly cautioned to invest focused effort in QA, testing and documentation of any attributes relied upon for effectiveness determination, regardless of the workflow used to develop them.

Why is it necessary to understand all these details when starting the FVS workflow? It all comes down to defining clear objectives for what answers are sought from BioSum so that FVS keyword control code can be crafted to generate FVS output that will make those analyses possible. Using the wrong timing of FVS cycles or failing to generate needed stand attributes in the FVS output can be a time-costly and frustrating exercise with few options for achieving a remedy short of going back to FVS to revise prescriptions or, in some cases, going all the way back to one of the 1st steps in this workflow—generating an FVS input file from the BioSum project to serve as the basis for your FVS simulations. For example, if it is critical that seedlings, down wood, and/or growth, removal, and mortality information be included in the FVS input file and thus downstream FVS simulation settings and outputs, that is a choice that must first be made in the Database module when loading plots into the project via the FIADB, then also early in this module's workflow.

BioSum supplies most of the required input data for FVS, including the StandInit (condition-level data) and TreeInit (tree-level data) tables, via an input database named "FVSin", which BioSum loads into the *projectname*\fvs\data\variant\ directories. BioSum generates FVSin databases as an SQlite database (FVSin.db). For legacy use, it also generates an MS Access formatted database (FVSin.accdb) via a different process that was only fully valid in the states of CA, OR and WA—the Access FVSIn database is not recommended, not fully supported and is retained, for now, mainly to provide a band-aid process that permits loading of FIA compatible data in these states that are not part of the pubic FIADB, such as off-grid plots that are not part of the standard FIA stratification. FVSOnline accepts either database format as valid input. The FVS Suppose interface, now supplanted by FVSOnline, accepts only the Access format (FVSin.accdb).

BioSum also creates rudimentary "stub" keyword script (KCP) files (that should be thought of as templates to start from) containing instructions defining the time horizon of the FVS simulation, which FVS output tables should be produced, and BioSum-specific parameters for those output databases. A BioSum analyst well-versed in FVS—including its operation, internal functioning, and limitations— faces the not undaunting task of fleshing out these KCP files with scripting code (mainly keywords and associated parameters), for executing stand management activities at the appropriate times. It can't be overemphasized that only a very rudimentary introduction to operating FVS is provided in this documentation. Analysts not deeply acquainted with FVS should enroll in FVS training (which is available via the Forest Management Service Center at Ft. Collins) or, at minimum, reading all available FVS documentation and completing FVS tutorials, which are available on the FVS web site (https://www.fs.usda.gov/fvs/training/index.shtml). New users are strongly advised to carefully study the next section of this chapter ("It's about time") for some pointers concerning how to specify the time intervals that define FVS cycles before they edit the template KCP code into the prescriptions that they will model in FVS. Writing (and testing) FVS prescriptions is a demanding exercise for the uninitiated; we believe it's essential to allow sufficient time to run quality assurance (QA) on this part of the BioSum workflow, and to run the simulations themselves—a process that can take days for large project areas and large numbers of silvicultural sequences. The FVS simulation work is typically the most time-consuming part of a BioSum analysis.

After silvicultural sequences have been modeled in FVS, you'll return to this module to extract data from the FVS output databases and load it into BioSum databases that convey it 1) to Processor, the module which estimates the costs of each silvicultural sequence applied to each stand and the volume and value of products that result from implementing these treatments, and 2) to Treatment Optimizer, which relies on both FVS outputs and Processor outputs to select the best management for each stand and estimate the wood produced, and associated costs and revenues.

A Word About Time

The astute reader will have noticed that the 4 cycles are referenced above as generating management activity *nominally* (at least as we refer to timing in this Users Guide) at years 0, 10, 20 and 30 (when 10-year cycles are selected). In BioSum, the four time points are referenced this way (or as years 0, 5, 10 and 15 when 5-year cycles are selected). However, the "actual" years at which management occurs during the simulation, and that are reported out of FVS and loaded into BioSum, depend entirely on 1) the Timeint (and InvYear) parameters supplied when running FVS, and 2) how FVS output table records are assigned to the 8 BioSum Pre- and Post-cycle "time-points" (one can think of them as registers or a vector of values, that may or may not technically be "pre" or "post" treatment years, even though they are labeled that way) during the FVS Output load step at the end of this chapter's workflow. This architecture is maximally flexible for the analyst but is admittedly confusing for those first learning BioSum.

In the interest of flexibility, BioSum is designed with the capacity to retain the inventory year associated with the FIA data loaded to the project (though note that "measurement year" MEASYR sometimes differs slightly from INVYR) throughout the analysis workflow for those who wish to do so. To support this capacity, BioSum assigns a sequence number to each year of simulation output when FVS outputs are loaded into BioSum (as described at the end of this chapter). It is usually less confusing to the user to instruct FVS to override the INVYR extracted from the FIA data by setting the InvYear FVS variable to 1 in the FVS control code so that all plots are on the same schedule of activities associated with a silvicultural sequence, regardless of the year in which they were intended to be measured in the field (i.e., the INVYR). Setting the FVS InvYear in this way results in a plot with an INVYR of 2020 and another with an INVYR of 2022 both entering the FVS simulation with an FVS Year value of 1.

As you will learn later in this chapter, the analyst can and should specify the length of each FVS simulation cycle for a BioSum analysis. It will likely not end well (in terms of obtaining interpretable results) if FVS is allowed to choose a default timing for its projections, or if you specify a fixed, regular cycle length of 5 or 10 years. FVS output tables typically contain either pre-treatment attribute values or post-treatment values—only a few contain both. Evaluating what treatment has accomplished at a particular time requires simulation output immediately before and after treatment, and for most FVS table types, that can only be accomplished by inserting a 1-year projection between the “real” projections. Even if conducting a weighted mean analysis (i.e., the mean value of outcomes across the 20 or 40 year simulation period), a 1-year projection is often required so that you will have representation of what a treatment has accomplished in the years immediately following its implementation.

An additional complication is that virtually all the tables produced via the Fire and Fuels Extension (FFE) of FVS (including POTFIRE, FUELS, CARBON and many others) contain no pre-treatment stand attributes if treatment is implemented at the beginning of the first FVS cycle. Earlier in BioSum’s development, capacity was added to insert results from a “base year” simulation for the POTFIRE table output only, so as to include pre-treatment values for that table in a BioSum analysis. However, that solution makes the workflow more confusing—for example, the base year information gets inserted as year 1 and what was year 1 becomes year 2, year 2 becomes year 3 and so on, but only for that table, such that the timing does not appear to match other tables. Because of its complexity, this solution was not implemented for other Fire and Fuels Extension tables and this capability is now documented only in an appendix. The replacement solution to the problem, and the approach advised for all new BioSum projects, is to always simulate a grow-only projection for 1 year before implementing any treatments. That way, your pre- and post-treatment years will likely be years 2 and 3 (for non-FFE tables) and years 1 and 2 (for FFE tables).

The tables below illustrate the timing choices using a set of FVS Timeint values that are mirrored in the last (Period Length) column of the FVS SUMMARY table output. The first FVS cycle is a grow-only cycle as indicated by 0 removed trees per acre (RTPA=0) and evidence of slow increase in basal area, quadratic mean diameter, and timber volume (TCUFT) over that 1 year as the stand age turned 48. The prescription implemented here thins the stand in year 2 and projects forward 1 year to obtain the post treatment summary attributes in year 3 that clearly show decreases in TPA, basal area, QMD and volume. Moreover, the after treatment basal area (ATBA) in year 2 shows the reduction caused by the thinning (a reduction offset in the next year of growth as basal area grew from 124 to 128). Note that year 2 values for all columns except the After Treatment (AT) ones, show only modest growth from the previous year, because the Summary table shows pre-treatment values (except in the AT columns). The third FVS growth cycle takes this stand forward 8 years to the end of the first decade at year 11. This stand is clearcut in year 12, which is the second BioSum cycle (the ATBA is zero).

FVS_Summary										
YEAR	AGE	TPA	BA	QMD	TCUFT	RTPA	RTCUFT	ATBA	ATSDI	PRD-LEN
1	47	457	245	9.9	6893	0	0	245	450	1
2	48	457	252	10.0	7153	139	3648	124	243	1
3	49	349	128	8.2	3615	0	0	128	254	8
11	57	296	159	9.9	5031	0	0	159	293	1
12	58	290	163	10.2	5151	290	5151	0	0	1
13	1	345	0	0.1	0	0	0	0	0	8
21	9	324	15	2.9	140	0	0	15	44	1
22	10	321	22	3.6	228	96	84	14	40	1
23	11	224	19	4.0	213	0	0	19	51	8
31	19	216	65	7.4	1250	0	0	65	134	1
32	20	216	73	7.9	1481	0	0	73	147	1
33	21	215	81	8.3	1647	0	0	81	159	8
41	29	208	145	11.3	3648	0	0	145	254	1
42	30	207	153	11.6	3899	0	0	153	264	0

The POTFIRE table, one of the several that are created by FFE and share a different timing format, shows outputs for the same stand. Timing is different on account of this table containing only post-treatment values. Year 1 shows the stand before treatment has occurred (because treatment was modeled at year 2). The effects of the treatment are obvious—the fire type under severe weather change from conditional crown to surface, total flame length drops, canopy density is reduced and predicted mortality under severe fire weather is greatly reduced both on the basis of percent basal area and cubic foot volume. Note that FFE tables like this POTFIRE table have one less year of output than other tables (the last 1-yr projection does not show up here). Note also that year 12 shows -1 for canopy base height—FFE's code for a missing value, because having just been clearcut, there are no trees so canopy base height is not defined. When conducting QA, these are very important things to check for as you would probably not want a minus 1 to become part of a weighted average, for example. Interrogation of FVS output tables, such as FVS_Summary and FVS_Potfire, is critical toward ensuring prescription timing and parameters are occurring as intended by the analyst and produce logical values.

YEAR	SURF_ FLAME_ SEV	TOT_ FLAME _SEV	FIRE_TYPE_SE V	PTORCH_S EV	CANOPY _HT	CANOPY_D ENSITY	MORTALIT Y_BA_SEV	MORTALITY_ VOL_ SEV
1	4.5	64.0	COND_CRN	0.00	36	0.19	100	6893
2	5.7	5.7	SURFACE	0.01	38	0.09	20	667
3	5.8	5.8	SURFACE	0.04	38	0.09	20	682
11	5.9	5.9	SURFACE	0.00	44	0.10	16	756
12	7.9	7.9	SURFACE	0.00	-1	0.00	0	0
13	9.8	9.8	SURFACE	0.00	-1	0.00	100	0
21	7.0	10.3	PASSIVE	1.00	4	0.04	99	139

22	7.0	9.1	PASSIVE	1.00	4	0.03	99	143
23	6.8	10.0	PASSIVE	1.00	4	0.04	99	212
31	6.0	10.9	PASSIVE	0.99	8	0.06	99	1241
32	5.9	11.2	PASSIVE	0.99	8	0.06	99	1469
33	5.7	9.0	PASSIVE	0.98	9	0.07	98	1629
41	5.2	5.2	SURFACE	0.29	17	0.08	43	1546

As indicated earlier, some questions focus on the immediate accomplishments of management, and so call for structuring the timing of simulations and the loading of FVS Output into BioSum so that there are pre- and post-treatment values available in BioSum' Treatment Optimizer for evaluating treatment effectiveness, for a particular cycle or for all cycles. For example, FVS could be directed, as it was in the examples just shown, to treat in year 2, 12, 22 and 32, and report pre-treatment stand metrics (to be used in evaluating effectiveness) for these years and post-treatment metrics (from the Summary and other non-FFE tables) for years 3, 13, 23, and 33. A Treatment Optimizer scenario could explore treatment accomplishment in BioSum cycle 1 (year 3 vs year 2).

Other questions may be best addressed by comparing a weighted combination (e.g., average or total) of stand attributes over the course of several decades for one silvicultural sequence against another, or against a grow-only sequence. In that event, having FVS output representing pre- vs post-treatment time points may be less important. For example, FVS simulations can be crafted to implement treatment at year 1, and report stand metrics at years 2, 7, 12, 17, 22, 27, 32, 37 and 42, with each successive time of reporting what would be expected to be a decline in treatment effectiveness. Treatment optimization might then compare an equal weighted average across all 8 of these time points for each silvicultural sequence against a similarly weighted average for a grow-only sequence. Alternatively, one could weight the average so that the gain over grow-only at year 42 is the sole basis of effectiveness characterization (by setting the weight for year 42 to one and the weights for all other years to zero) if the only concern is the state of the forest at the end of 4 decades of management.

The take-home points here are:

1) Although the FVS module and this documentation that describes its use references years 0, 10, 20 and 30 to represent the 4 cycles accounted for in the BioSum analysis framework, in the FVS output databases, these may correspond to very different years, depending on the purpose and structure of the analysis.
2) While it may be tempting to launch into tasks with the BioSum manager, which are described in this chapter, it is very important to think through the analysis that you want to carry out so that you create an FVS input file and FVS keyword code that will support those objectives. Taking the time to understand how BioSum operates (e.g., by carefully reading this Users Guide in full) and, if possible, experimenting with the starter project before progressing through an intended BioSum project, is essential to ensuring a valid and useful analysis.

Assigning ”Additional CPA (cost per acre)” that apply to specific treatments

There’s one more topic to introduce before launching into the FVS Module workflow, because it surfaces in several places in the BioSum workflow and, like timing, needs to be planned for when developing KCP codes. BioSum’s Processor module (next in the workflow sequence) provides for calculating, for each stand, a per-acre cost of the harvest activity associated with a silvicultural treatment at a given time using the OpCost model – a script processed in R that transforms summaries of a cut tree list into machine time requirements, and ultimately, harvest costs per acre for that stand and treatment at that time. It is often necessary or desirable to also account for what BioSum labels as “Additional CPA” because they are over and above the costs modeled for the tree harvest, yarding, loading, chipping and mobilization/demobilization expenses accounted for by OpCost. Examples of costs that that may need to be tracked include those associated with surface fuel treatments (such as mastication, prescribed fire or pile burning); mechanical site prep; application of herbicide, fertilizer, lime or biochar; water-barring skid trails; and planting or seeding. Many of these are not harvest costs at all, but rather, costs of activities that are part of a prescription that may or may not include removal of trees and which may be incurred at the time of tree removal, or at another time. For example, a management trajectory that thins a stand in year 1 may return to implement a prescribed fire in year 21 of the simulation via the SIMFIRE keyword. In that case, there is no harvest cost in year 21 (because no trees are harvested), but there is an Additional CPA – the cost of the prescribed fire.

By thoughtful use of KCP scripting, these additional costs can be modeled as contingent on information generated during the FVS simulation of the prescription for a given stand—for example, pile and burning might be considered necessary when activity fuels (down wood resulting from thinning) exceed some threshold, say 15 tons per acre, but not otherwise. If FVS is coded to implement a pile and burn only when that threshold is exceeded, then the cost assigned to pile and burn should only be incurred if FVS calculates activity fuels as > 15 tons/ac and conditionally implements the pile and burn treatment.

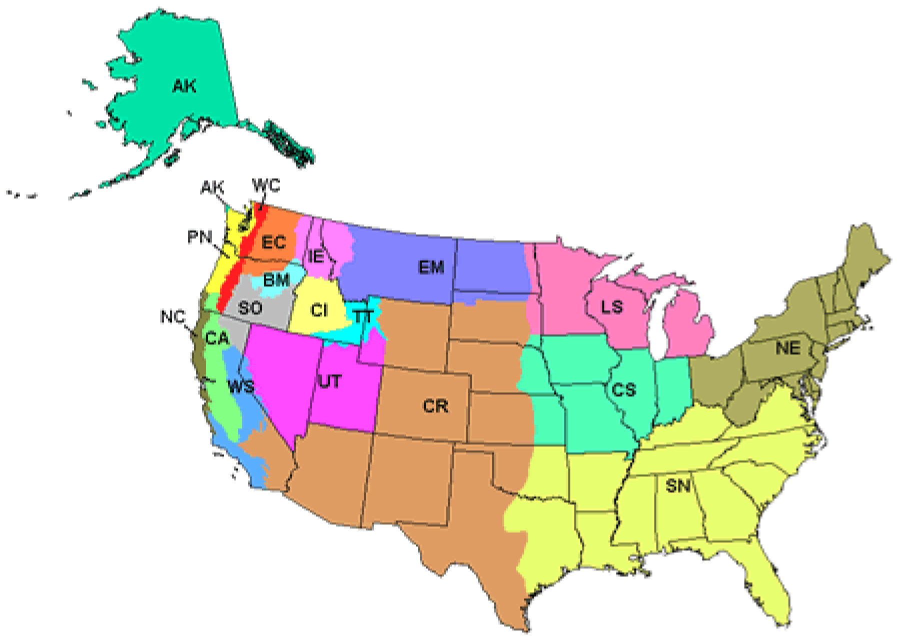
Accounting for Additional CPA requires intervention at three points in the BioSum workflow, chronologically: 1) when the prescription is defined or edited in the FVS module of BioSum workflow (accessed via the “Rx” button), 2) in the development of the KCP code that supports the simulation of that prescription in FVS, and 3) in the Additional CPA tab of Processor. Adhering to the guidance for these three workflow steps results in Treatment Optimizer generating the ‘Additional_kcp_cpa’ table containing a breakout of additional harvest costs and including these additional costs in the overall onsite harvest cost and net revenue calculations in the Processor and Optimizer modules.

Module Components

Finally, we are ready to learn the workflow of BioSum’s FVS module! This module includes six component tasks, all of which should be completed before proceeding to Processor: (1) Plot FVS Variants—review FVS variant and location code assignments for each plot, filling in values for plots not yet assigned variants, via an audit that draws from a reference database table, (2) Rx—define or edit each silvicultural prescription (treatment) that may be applied as part of one or more silvicultural sequences, (3) Rx Package—define or edit each package of treatments (a.k.a. silvicultural sequence) to be applied over up to four BioSum cycles, (4) Tree Species—inspect, audit, edit and assign FVS trees species codes to FIA trees for all species present in the project, (5) FVS Input Data (FVSin)—create both the input database (containing stand and tree lists) that will be used during FVS processing and the BioSum_Keywords.kcp (keyword control program) that will, when fleshed out by the analyst, control FVS operations, (6) Create FVSOUT MDBs – partitions FVS output data into individual Access databases by variant and RxPackage combination, and finally (7) FVS Output Data (FVSOut)— retrieve FVS tabular output and cut list tables into BioSum project databases, to feed to the Processor and Treatment Optimizer modules.

Plot FVS Variant

FVS simulations rely on equations and computation code that differ among geographic areas, known as "variants" in FVS parlance (Figure 4.1).



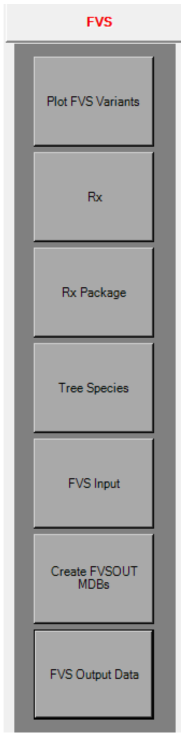
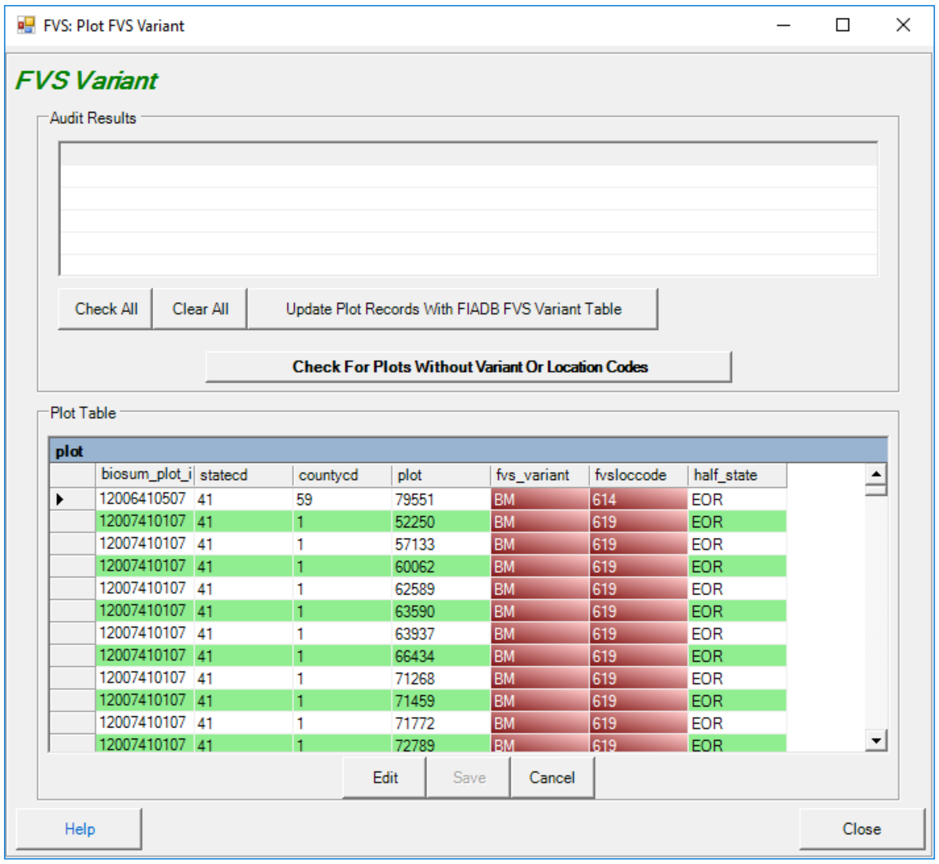


Figure 4.1 – FVS variant map. Source: https://www.fs.usda.gov/fvs/documents/overviews/FVSVariantMap20210525.zip

In general, the equations, such as those that predict diameter and height growth, were developed using growth data from trees within a variant and "location code", a finer-scale geographic unit. Surface fuel model assignment, site class calculation, mortality estimation, crown ratio and crown width parameters, and fuel moisture content assumptions are a few of the many other items that may differ among variants and location codes. It is <u>very</u> important that each plot, and the stands (conditions) that it contains be assigned to the variant and location code in which it geographically resides. The *fiadb_fvs_variant* table in the *ref_master* database contains the correct variant assignments and location codes (typically a concatenation of NFS Region and Forest numbers) for all plots in the FIA database in the states of CA, OR, WA, ID, MT and UT, which makes it easy to auto-assign the right values.

1. To assign the appropriate variant and location code to plot records or to review the current assignments, select **<Plot FVS Variants>.**
2. The **Plot FVS Variants** window will appear, with a list of plots and their existing fvs_variant and fvsloccode assignment in sienna-brown, editable columns. This table can be browsed, sorted (indexed) or filtered, via functions available from the right click menu when right clicking while the cursor is placed in the desired column. For example, one could specify a particular value to filter on in the county column (Filter by Entered Value) to list all plots in that county, or "filter by entered value" the fvs_variant column (leaving the entered value blank) to list all plots that lack a variant assignment. However, it is easier to find cases of plots that lack a variant assignment by running an "audit" on the plot. The prudent analyst will run audits whenever they are offered in the BioSum workflow to identify potential issues that, if ignored, could corrupt or invalidate the BioSum analysis.



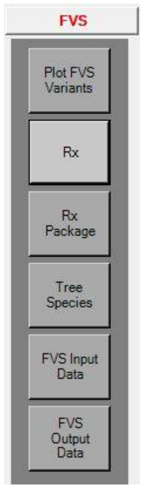
biosum_plot_i	statecd	countycd	plot	fvs_variant	fvsloccode	half_state
12006410507	41	59	79551	BM	614	EOR
12007410107	41	1	52250	BM	619	EOR
12007410107	41	1	57133	BM	619	EOR
12007410107	41	1	60062	BM	619	EOR
12007410107	41	1	62589	BM	619	EOR
12007410107	41	1	63590	BM	619	EOR
12007410107	41	1	63937	BM	619	EOR
12007410107	41	1	66434	BM	619	EOR
12007410107	41	1	71268	BM	619	EOR
12007410107	41	1	71459	BM	619	EOR
12007410107	41	1	71772	BM	619	EOR
12007410107	41	1	72789	BM	619	EOR

Figure 4.2 – Plot FVS Variants window containing a list of plots and existing FVS variant and location code assignments.

3. To run an audit, select the button labeled **<Check For Plots Without Variant Codes>,** which initiates a script that identifies plots without a variant assignment. A message box will indicate whether the audit passed or failed, and if it failed, those plots without variant assignments will appear in the table in the upper half of this task window, provided they exist in the fiafb_fvs_variant table in the database C:\Users\username\AppData\Roaming\FIABiosum\biosum_ref.accdb.
4. Select the records to be updated using the **<Check All>** button, or manually toggle the checkbox for individual plot records. After records are selected, update the plot table using the **<Update Plot Records With FIADB FVS Variant Table>** button to automatically make variant and location code assignments based on the values in the fiadb_fvs_variant table in the *ref_master* database. Inspect and sort the fvs_variant column of the Plot Table to make sure that fvs variants have been assigned for all plots. To confirm that all plots have been assigned a variant, the audit can be re-run by selecting **<Check For Plots Without Variant Codes>**.
5. If plots exist outside the states for which the fiadb_fvs_variant table contains variant assignments, it will be necessary to manually assign values to replace NULL values. It may be helpful to consult the FVS website (http:\\www.fs.fed.us\fmsc\fvs\) for information on assigning an appropriate variant code. When location code is omitted, FVS will revert to the default location code for the variant, which may or may not result in valid projections for those stands.
6. Click **<Save>,** then **<Close>** to save the FVS variant assignments and exit the task window.

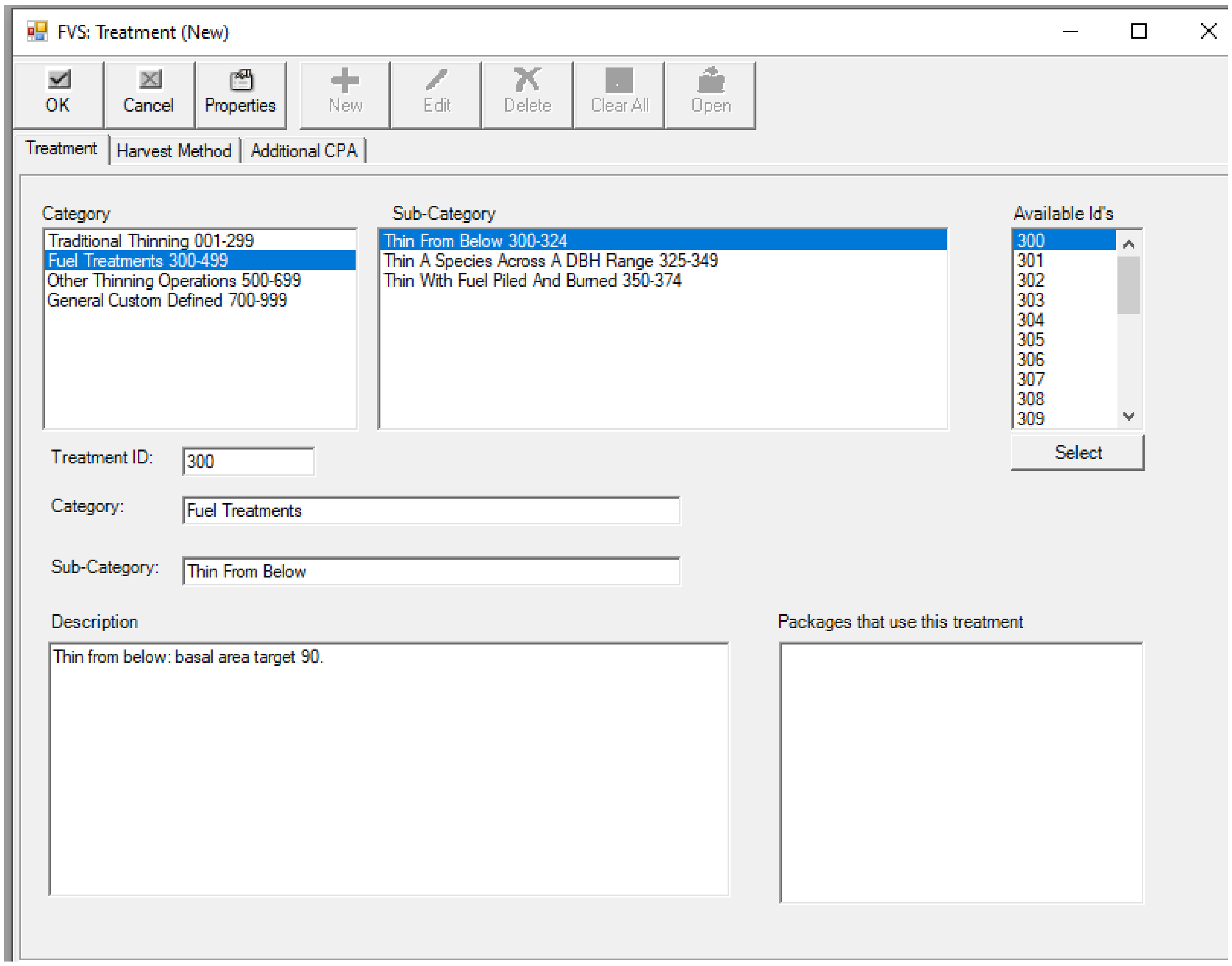
Rx (Prescriptions)

The next step is to define and label the silvicultural prescriptions (treatments) to be simulated in FVS. We refer to treatments as “Rx” throughout this guide. Treatments ultimately consist of a file of keywords and parameters provided to FVS to guide what happens at a particular time. For example, one might define treatment 200 as “thin across diameter classes (a constant harvest proportion applied to all diameter classes) to a residual density of 85 sq. ft basal area per acre, while cutting no trees larger than 36” in diameter, using mechanized whole tree harvest on gentle slopes and cable manual logging on steep slopes”. The harvest systems are not specified in the KCP file, but the implications of their use are, for example, via “yardloss” keyword statements indicating the quantity of wood residues left behind. To model a different residual density or upper diameter limit, you would create additional prescriptions (e.g., 201 with residual basal area=100, upper dimeter limit 36”, etc.). Sometimes two treatments will be identical except that one uses whole tree harvest systems and captures all or most forest residues for utilization (and/or piles them at the operation landing) while the other relies on a log length system that brings only merchantable logs to the landing, leaving residues in the forest, possibly to be treated later at additional expense. Treatments may also include activities that take place in the stand that are not harvest *per se*, for example, piling and burning or masticating harvest residues; however, these are usually specified as supplemental (to tree harvest) activities that incur additional harvest costs specified by the user. Once treatments are defined, they can be assigned to one or more prescription packages, which are simply sequences of either the same or different silvicultural prescriptions, implemented over 4 cycles. Note that some cycles in a package may have Rxs that are essentially "grow-only" and harvest no trees. By convention, it can be useful to define Rx 999 as grow-only, for repeated use in a grow-only silvicultural sequence.



Prescription definition workflow is as follows:

1. Define silvicultural treatments to be applied in FVS by clicking **<Rx>** to load the Rx task window.
2. For each treatment to be added to the project:
    a. Select **<New>** to open the **Treatments** window, which will display four tabs: Treatment, Harvest Method, Harvest Costs, and Associated FVS Command(s), which we explain below.
    b. **Treatment:** After selecting the treatment tab, choose a category, sub-category, and ID number. Each treatment must be assigned a value between 001 and 999. Treatment ids 001 to 699 represent pre-defined treatment categories, and treatment ids 700 to 999 can be used to label any custom-defined treatment. However, compliance with these conventions is optional—any prescription number can be assigned to any prescription. After choosing an Rx number for the new Rx, enter a brief description of that treatment. These descriptions usually prove to be far more important and useful than compliance with pre-defined treatment categories. Later in the BioSum workflow, these descriptions may be essential to help recall what the Rx was intended to accomplish. Apply your edits by clicking **<Select>.**



Category

Traditional Thinning 001-299
Fuel Treatments 300-499
Other Thinning Operations 500-699
General Custom Defined 700-999

Sub-Category

Thin From Below 300-324
Thin A Species Across A DBH Range 325-349
Thin With Fuel Piled And Burned 350-374

Available Id's

300
301
302
303
304
305
306
307
308
309

Treatment ID: 300

Category: Fuel Treatments

Sub-Category: Thin From Below

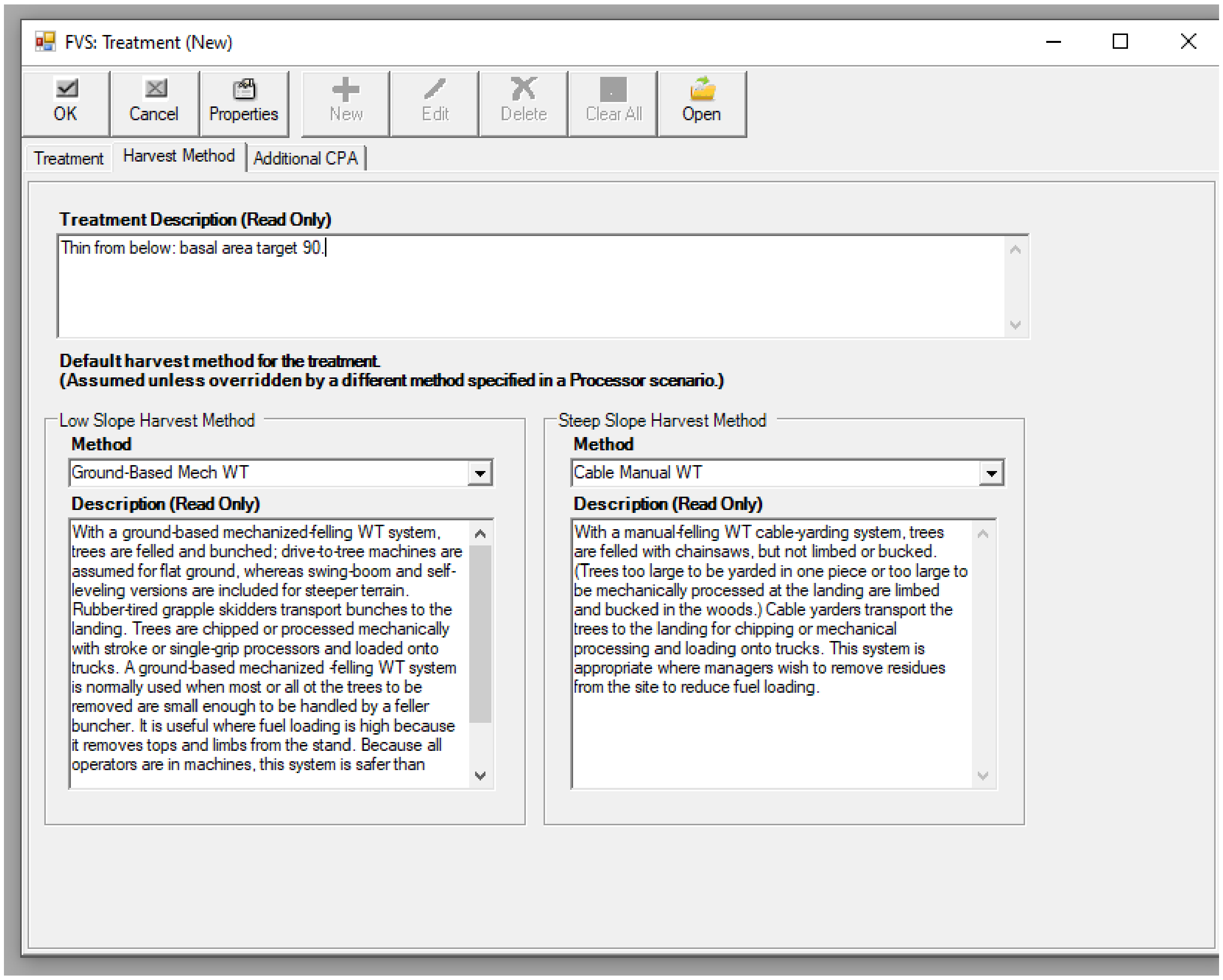
Description

Thin from below: basal area target 90.

Packages that use this treatment

Figure 4.3 – Treatment tab of the FVS:Treatment window. Treatment category, sub-category, ID #, and description are assigned at this stage.

c. **Harvest Method**: Choose the Harvest Method tab. Select a harvest method for low slopes and a harvest method for steep slopes. The percent slope threshold above which slope is categorized as steep will be specified later in the Processor module of BioSum (chapter 5). Once a harvest method is selected, a brief description of the method will appear in the Description text box. Refer to **Appendix A** for a list and description of harvesting methods available for selection.



Treatment Description (Read Only)

Thin from below: basal area target 90.

Default harvest method for the treatment.
(Assumed unless overridden by a different method specified in a Processor scenario.)

Low Slope Harvest Method

Method

Ground-Based Mech WT

Description (Read Only)

With a ground-based mechanized-felling WT system, trees are felled and bunched; drive-to-tree machines are assumed for flat ground, whereas swing-boom and self-leveling versions are included for steeper terrain. Rubber-tired grapple skidders transport bunches to the landing. Trees are chipped or processed mechanically with stroke or single-grip processors and loaded onto trucks. A ground-based mechanized -felling WT system is normally used when most or all ot the trees to be removed are small enough to be handled by a feller buncher. It is useful where fuel loading is high because it removes tops and limbs from the stand. Because all operators are in machines, this system is safer than

Steep Slope Harvest Method

Method

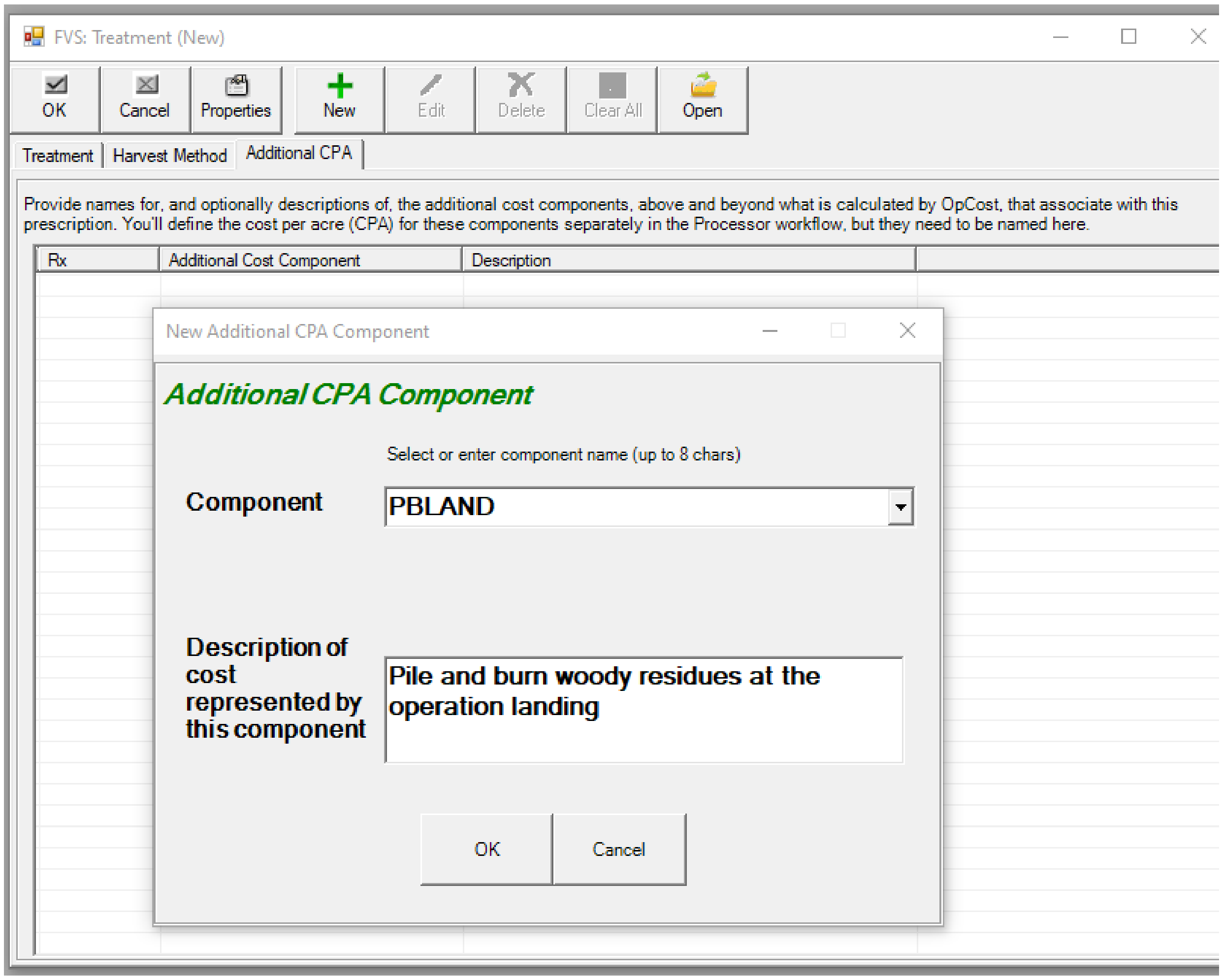
Cable Manual WT

Description (Read Only)

With a manual-felling WT cable-yarding system, trees are felled with chainsaws, but not limbed or bucked. (Trees too large to be yarded in one piece or too large to be mechanically processed at the landing are limbed and bucked in the woods.) Cable yarders transport the trees to the landing for chipping or mechanical processing and loading onto trucks. This system is appropriate where managers wish to remove residues from the site to reduce fuel loading.

Figure 4.4 – Harvest Method tab of the FVS:Treatment window. Select harvest methods for low and steep slopes.

d. **Additional CPA:** Add any additional cost components – i.e., those that are not accounted for by the OpCost model, such as the cost for surface fuel treatments (e.g., mastication, prescribed fire or pile burning); mechanical site prep; application of herbicide, fertilizer, lime or biochar; water-barring skid trails; and planting or seeding, as discussed earlier in the module Overview. Define a cost component by selecting **<New>** and entering the name for the cost component and a brief description in the window that pops-up. The analyst will subsequently need to set an activity flag (that will be passed along in FVS's COMPUTE table) with the same name as this cost component in the FVS KCP script that controls the simulation. To ensure compatibility with FVS scripting, the name/flag needs to be no more than 8 characters, start with a letter, and avoid spaces and special characters, so choose 'RXFIRE' rather than 'Rx Fire' or 'RX_FIRE'. The actual costs (in dollars per acre) of these components will need to be assigned in the Processor module (Chapter 5) and will be incurred in the accounting for management costs when the flag is set in the FVS simulation.



Provide names for, and optionally descriptions of, the additional cost components, above and beyond what is calculated by OpCost, that associate with this prescription. You'll define the cost per acre (CPA) for these components separately in the Processor workflow, but they need to be named here.

Rx	Additional Cost Component	Description

New Additional CPA Component

Additional CPA Component

Select or enter component name (up to 8 chars)

Component: PBLAND

Description of cost represented by this component: Pile and burn woody residues at the operation landing

Figure 4.5 – Additional cost per acre (CPA) tab of the FVS: Treatment window. Add or edit any additional cost components associated with this treatment.

e. When all desired information has been entered (treatment and harvest method are required; Harvest Costs is optional), select **<OK>** in the upper left corner to return to the Treatment List. Your treatment will not be saved in the project until clicking **<Save>** at the bottom of the treatment list.

3. To make changes to a treatment already displayed in the treatment list, select it (it will show as highlighted) and then click **<Edit>**.
4. When finished creating or editing treatments, click **<Save>,** then **<Close>,** to exit the **Treatments** window.

Note that one can add new prescriptions (and silvicultural sequences) to a project at any time; it is not uncommon to think of new prescriptions to try after seeing how those that were first modeled play out. You would simply come back to this module, define the new prescriptions and sequences, run them through FVS and load the FVS output from those sequences back into BioSum and proceed to Processor to process them.

What is a Silvicultural Sequence (a.k.a. RxPackage), conceptually?

Given that FVS is a very customizable model, there are more ways to design and conceptualize a simulation than can possibly be described in this Users Guide. Some thought must be given to what the meaning an analyst intends as by a silvicultural sequence, management alternative, prescription package, or whatever they choose to call a sequence of potential activities over time. BioSum expects to see FVS output that represents stand attributes of interest to the analyst at 8 time points—for example, Pre and Post management activity at each of 4 cycles of management opportunity (whether or not management activity actually occurred at those times). Thus sequence 001 could be defined as implementing Rx 250 at each of 4 cycles (Repeat Every Cycle); however, Rx 250 might be defined in the KCP file to conditionally cut trees only when a minimum basal area is exceeded, for example, in which case trees might be cut at only some, one or none of those cycles (Repeat Every Cycle Contingent). Or, sequence 001 could be defined as implementing Rx 250 at cycle 1 and not in the other cycles, sequence 002 could be defined as implementing Rx 250 only at cycle 2, etc. (Specific Cycle). Alternatively, sequence 001 could be defined as implementing Rx 250 at cycle 1 or any cycle at which some stand attribute threshold is reached, then implementing a different, “tune-up” treatment Rx 300 (perhaps one that cut no trees but implemented a prescribed fire to clear out ladder fuels arising from regeneration) 1 or 2 cycles after implementing Rx 250 (Mixed Rx Contingent). What a silvicultural sequence means then, and how results can be interpreted, are very much interdependent. For example, when comparing the effectiveness among treatments at cycle 1, a Specific Cycle “sequence” not timed to occur at Cycle 1 may never be effective (if effectiveness is judged at cycle 1), while a Repeat Every Cycle Contingent sequence may be effective in stands where actual treatment was triggered at cycle 1 and not for other stands where no actual treatment occurred at cycle 1.

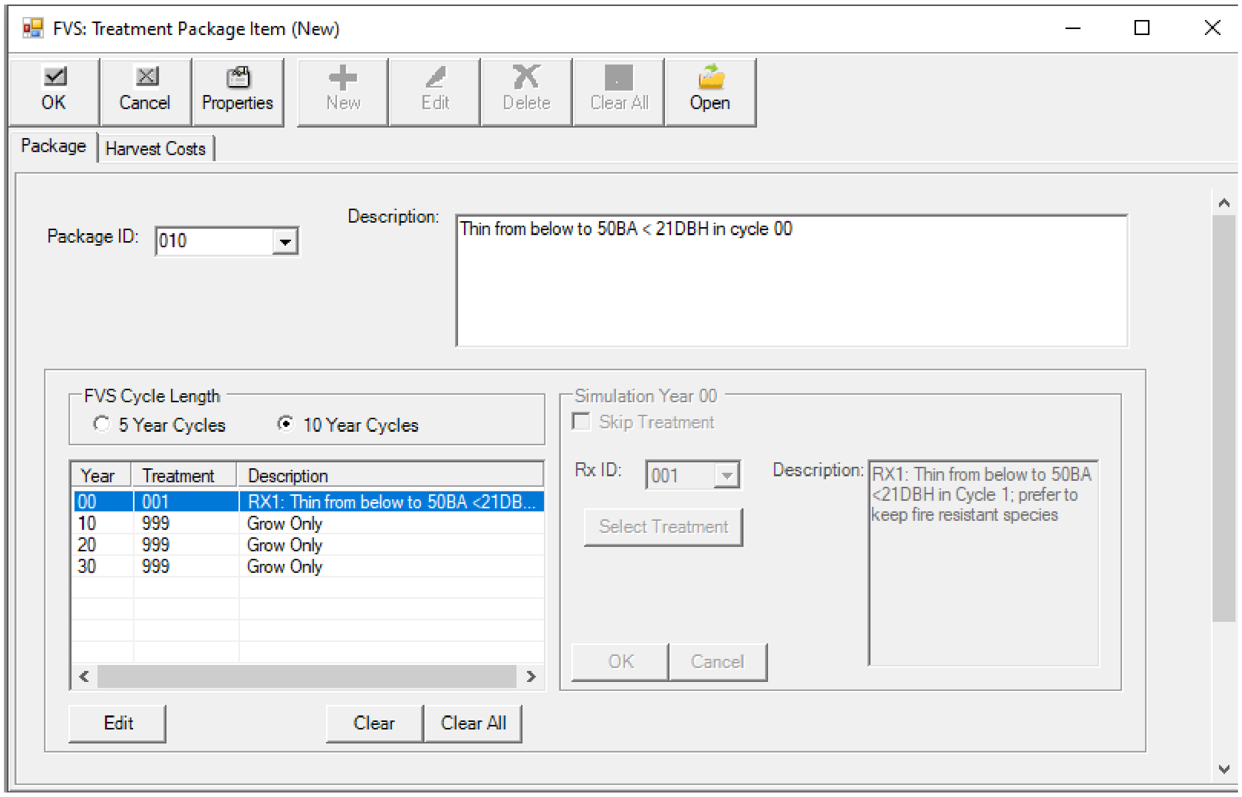
There is no one right answer to the question posed in the title of this section—different choices make sense for different purposes. However, analysts are advised to carefully consider and document the intention and definition of every sequence modeled and to keep these in mind when interpreting BioSum outputs later on.

Defining a Silvicultural Sequence (Rx Package) in BioSum

A silvicultural sequence (or Rx [prescription] package) is a series of treatments or management activities applied to a stand either at one time or over multiple times during the four, 5- or 10-year cycles of a BioSum analysis. In the Treatment Optimizer module, these silvicultural sequences can be evaluated and compared to one another and/or to a grow-only (no treatment) silvicultural sequence, depending on analysis objectives. An Rx package consists of a sequence of one or more of the Rxs already defined, with the possibility that some cycles will have no Rx assigned or will assign a prescription that is defined as grow only (these have the same net effect but result in different labeling of the project databases).

1. Assemble a sequence of silvicultural prescriptions to be applied in FVS by clicking **<Rx Package>** to load the Treatment Packages task window.
2. To create a new package:
    a. Select **<New>** to open the Treatment Package Item window, which has two tabs: Package and Harvest Costs.
    b. **Package:** To create a new package**:**
        i. Select an available package ID number from the **<Package ID>** drop-down list. It is usually easiest to start at 001, and increment from there as new Rx packages are defined.
        ii. Write a brief description of the package in the text box next to **<Description>**.
        iii. Next, specify whether your Rx package will utilize 5- or 10-year cycles by selecting the radio button next to the cycle length of choice in the **<FVS Cycle Length>** box. (Note: all packages within a BioSum project must have the same cycle length to generate valid results)
        iv. To add a treatment to a cycle year, select the year: 00, 10, 20, or 30, and click **<Edit>.**
        v. Uncheck the box next to **<Skip Treatment>** (by default, each cycle will be populated with a "skip treatment" via the check box, which will assign a null Rx coded as 000)**.**
        vi. Assign a prescription from the dropdown list next to **<Rx ID>.** You will see a list of prescription ids that have been defined for this project in a dropdown list; select one to assign. If a grow-only prescription was formally designated earlier as 999, that choice will be available here also.
        vii. Select **<OK>** to add the treatment to the package. The selected treatment ID will appear next to the desired year along with its description.
        viii. To add another treatment to the package, return to step iv. If no treatment is to be administered in a cycle year, no action is necessary; BioSum will assume no treatment as indicated by the "skip treatment" box being checked. However, it is equally valid, and perhaps preferred, to assign a grow-only (e.g., Rx 999) if treatment is not planned, and doing so may help avoid confusion over whether a treatment was inadvertently omitted when the sequence was defined (causing a 000 skipped treatment code [that appears as null in the BioSum interface] that will persist in all databases and kcp files associated with the package). **One very important point to emphasize** is that BioSum pays very close attention to these codes and assumes that a skipped treatment means just that. It is not adequate to enter a Rx for cycle 1 and leave skipped

treatment checked for the other cycles. Even if the FVS kcp file is structured as Repeat Every Cycle Contingent (as described above), and FVS generates a cut list in cycle 3 (because the contingency is met), BioSum will not read that cut list data from the FVS output if the Rx coded for cycle 3 is 000 (skipped treatment). If using contingent treatment specifications that may or may not actually harvest trees in a given cycle, it is essential that they be specified for every cycle in which FVS **may** generate a cut list if the contingency is met.


FVS: Treatment Package Item (New)

Package ID: 010

Description: Thin from below to 50BA < 21DBH in cycle 00

FVS Cycle Length: 5 Year Cycles [✓ 10 Year Cycles]

Year	Treatment	Description
00	001	RX1: Thin from below to 50BA <21DB...
10	999	Grow Only
20	999	Grow Only
30	999	Grow Only

Simulation Year 00

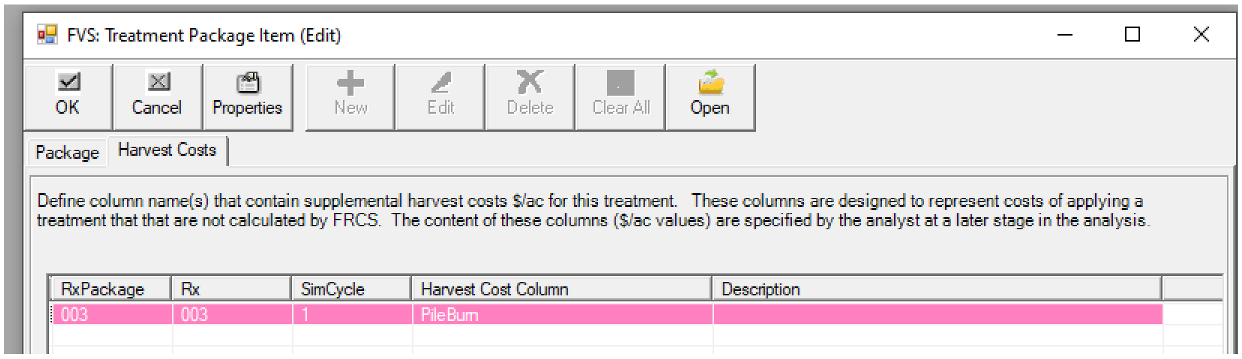
Skip Treatment: unchecked

Rx ID: 001

Description: RX1: Thin from below to 50BA <21DBH in Cycle 1; prefer to keep fire resistant species

Figure 4.7 – Package tab of the FVS: Treatment Package window. Define a package as a sequence of previously defined Rxs.

c. **Harvest Costs:** This window contains a list of harvest cost components (not the actual per acre costs, just the names of the components and the cycles in which they apply), assigned during **Rx** creation, for each prescription in the package. These items are read-only since they are associated with the treatment, not the package. Remember, the cost per acre values will be specified at a later stage in the analysis.


FVS: Treatment Package Item (Edit)

Define column name(s) that contain supplemental harvest costs $/ac for this treatment. These columns are designed to represent costs of applying a treatment that that are not calculated by FRCS. The content of these columns ($/ac values) are specified by the analyst at a later stage in the analysis.

RxPackage	Rx	SimCycle	Harvest Cost Column	Description
003	003	1	PileBurn	

Figure 4.8 – Harvest Costs tab of the FVS: Treatment Package window. View a list of harvest cost components associated with Rxs in this package.

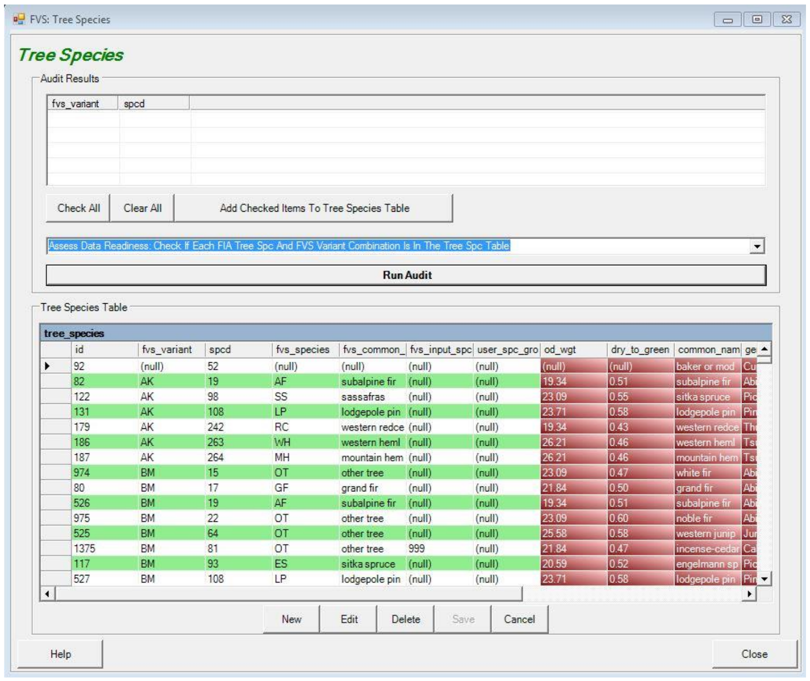
   d. Click **<OK>** to close the Treatment Package Item window. Note: the changes are not yet saved (see next step).

3. Repeat step 2 for each package you wish to create. When finished, click **<Save>** then **<Close>** to exit.
4. To make changes to a treatment package you have already created, simply select the package from the Treatment Package List table and click **<Edit>.** Select **<Delete>** to delete a package from your project. Remember that no changes, including deletions, are saved as final until clicking <Save> at the bottom of the Treatment Package List.
5. To display full documentation of prescriptions and packages in the project, (warning: this can take several minutes to complete) click **<Properties>** from the Treatment Packages window to open an html file in your default browser. The page will open in your browser and display details for every treatment assembled for this project database. Clicking the hyperlink **<Printer Friendly>** in the upper left hand corner will display a printable version of the summary table.
6. Click **<Close>** to exit the Treatment Packages window.

Tree Species

FVS predictions are customized to both geography and tree species, via allometric, regional and tree species specific equations for tree growth, mortality, and volume. These equations are embedded in the FVS framework and result in a geographic variant of FVS. Therefore, each tree in a BioSum analysis must be assigned an FVS tree species code. The FIA database, however, has a greater number of tree species than are specifically simulated in FVS. The *tree_species* table in the project's *ref_master* database contains FVS species code assignments for many FVS variant and FIA species combinations. If you have a tree species in your data that is not recognized by the FVS variant used for prediction, the *tree_species* reference table may be customized to "map" that species to one that FVS does recognize for that variant. This step in the FVS module offers an opportunity to ensure all trees in your study area are assigned an appropriate FVS species code. Trees that have not been assigned a FVS species code valid for the variant in which they are modeled can lead to unpredictable results, including invalid predictions of growth and biomass and volume that “disappear” from the analysis.

1. Open the Tree Species window by selecting **<Tree Species>** from the side menu.
2. The **Tree Species** window will appear, with the *tree_species* reference table visible at the bottom. This table can be browsed, sorted, or filtered to, for example, find specific tree and variant combinations, or reference oven dry weight and dry to green values for specific tree species. To sort or filter, right click in a cell under the desired column type and select the desired type of filtering.
3. To ensure every FIA tree species and FVS variant combination in your project has an assigned FVS species code, select **<Run Audit>** from the Tree Species window.



fvs_variant	spcd

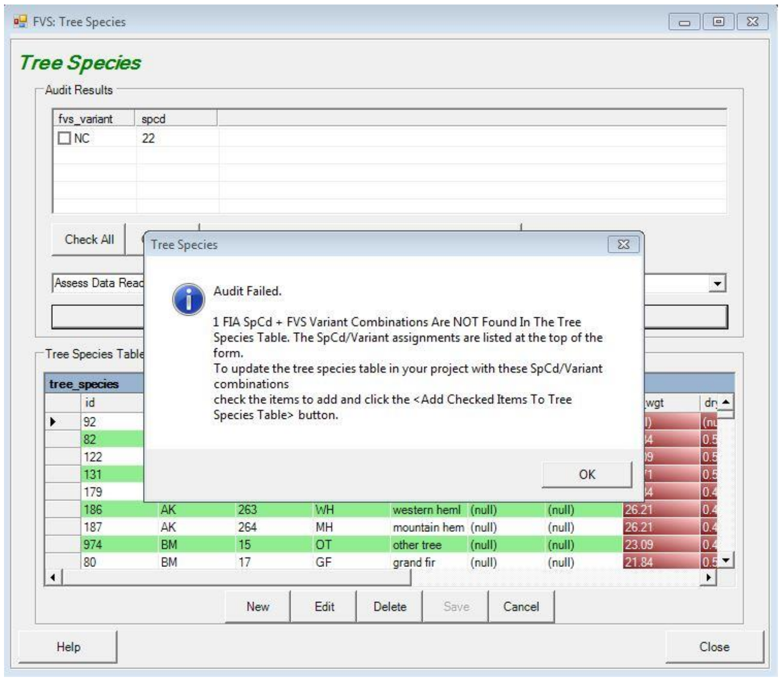
Assess Data Readiness: Check If Each FIA Tree Spc And FVS Variant Combination Is In The Tree Spc Table

tree_species

id	fvs_variant	spcd	fvs_species	fvs_common_	fvs_input_spc	user_spc_gro	od_wgt	dry_to_green	common_nam	ge
92	(null)	52	(null)	(null)	(null)	(null)	(null)	(null)	baker or mod	Cu
82	AK	19	AF	subalpine fir	(null)	(null)	19.34	0.51	subalpine fir	Abi
122	AK	98	SS	sassafras	(null)	(null)	23.09	0.55	sitka spruce	Pic
131	AK	108	LP	lodgepole pin	(null)	(null)	23.71	0.58	lodgepole pin	Pin
179	AK	242	RC	western redce	(null)	(null)	19.34	0.43	western redce	Th
186	AK	263	WH	western heml	(null)	(null)	26.21	0.46	western heml	Ts
187	AK	264	MH	mountain hem	(null)	(null)	26.21	0.46	mountain hem	Ts
974	BM	15	OT	other tree	(null)	(null)	23.09	0.47	white fir	Abi
80	BM	17	GF	grand fir	(null)	(null)	21.84	0.50	grand fir	Abi
526	BM	19	AF	subalpine fir	(null)	(null)	19.34	0.51	subalpine fir	Abi
975	BM	22	OT	other tree	(null)	(null)	23.09	0.60	noble fir	Abi
525	BM	64	OT	other tree	(null)	(null)	25.58	0.58	western junip	Jur
1375	BM	81	OT	other tree	999	(null)	21.84	0.47	incense-cedar	Ca
117	BM	93	ES	sitka spruce	(null)	(null)	20.59	0.52	engelmann sp	Pic
527	BM	108	LP	lodgepole pin	(null)	(null)	23.71	0.58	lodgepole pin	Pin

Figure 4.9 – Tree Species window. Run an audit to check for records without an FVS species code assignment.

4. A list of species\variant combinations without an assigned FVS species code will appear in the **Audit Results** table in the upper half of the window, and a pop-up window will appear with audit details. If there are no missing values in your project database, a pop-up window will appear stating that the audit has passed.   If your project contains tree species that do not have a corresponding FVS species code in the Tree Species table, an error message will be displayed to alert you that the audit failed (*Figure 4.10).*



fvs_variant	spcd
☐ NC	22

Audit Failed.

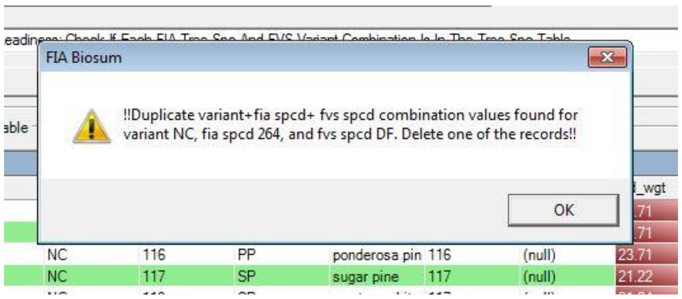
1 FIA SpCd + FVS Variant Combinations Are NOT Found In The Tree Species Table. The SpCd/Variant assignments are listed at the top of the form.
To update the tree species table in your project with these SpCd/Variant combinations
check the items to add and click the <Add Checked Items To Tree Species Table> button.

tree_species

id							wgt	dr
92								
82								
122								
131								
179								
186	AK	263	WH	western heml	(null)	(null)	26.21	0.4
187	AK	264	MH	mountain hem	(null)	(null)	26.21	0.4
974	BM	15	OT	other tree	(null)	(null)	23.09	0.4
80	BM	17	GF	grand fir	(null)	(null)	21.84	0.5

Figure 4.10 – Tree Species audit window. Audit will fail if a variant\species code combination is not found in the tree_species table.

5. To add the records with null values (missing or invalid FIA/FVS species code combinations) to the tree species table, select the check boxes next to each record, or use the **<Check All>** button to add all records with null values, and then click **<Add Checked Items To Tree Species Table>.** To clear any selected records in the Audit Results table click <**Clear All**>.
6. Once records are in the *tree_species* table, a record can be edited by clicking the **<Edit>** button to open the Procesor Tree Species Edit window to edit the FVS variant, FIA tree species code, common name, oven dry weight, and other descriptors associated with the selected entry.
7. Save edits to the *tree_species* table by clicking on the **<Save>** button. Changes to the table will not be updated until saved.
8. If duplicate tree species\variant combinations are found within the table, a warning will appear when changes are saved (*Figure 4.11*). Close the warning window using the **<OK>** button, select one of the duplicate records in the table, and click **<Delete>**. Be sure to **<Save>** your update.


FIA Biosum

!!Duplicate variant+fia spcd+ fvs spcd combination values found for variant NC, fia spcd 264, and fvs spcd DF. Delete one of the records!!

Figure 4.11 – If BioSum identified duplicate FIA species/FVS variant combinations, it will not allow the tree_species table to be saved. Duplicate records must be deleted before continuing.

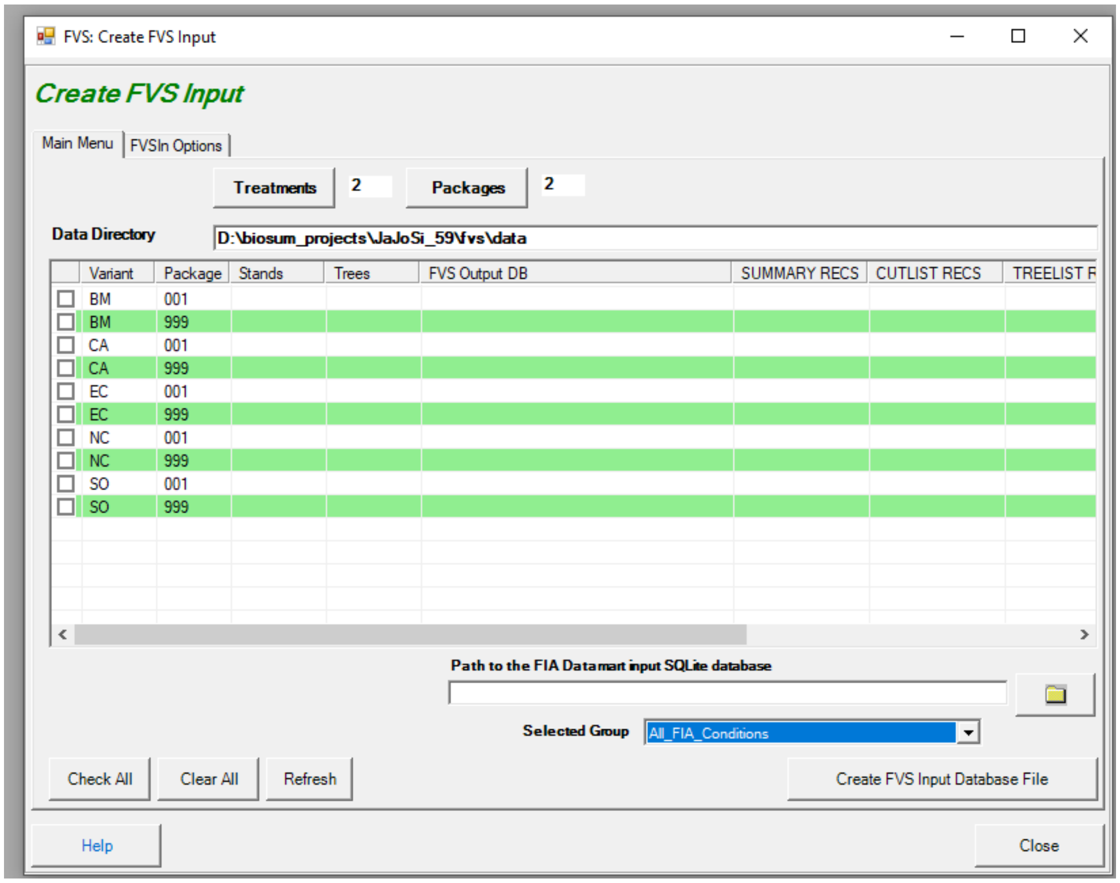
FVS Input Data

The next task is to create the files necessary for FVS growth projection: tree and stand data that serve as simulation input. While there are several pre-processing tools that can be used to create the input files needed to run a simulation in FVS, it is essential that when using BioSum, these **files be created following BioSum's FVS Input Data workflow** to ensure linkage between FIA data and FVS simulation output.

As noted in Step 24 of the Database chapter of this user guide, all choices about what data will be available to be passed to FVS, including calibration, down wood, and seedling data must be made at the time plot data is imported into a BioSum project. Otherwise, these data cannot be included in the SQLite FVS input files (i.e., FVSin.db) created by BioSum.

To create FVS input files:

1. Open the **Create FVS Input** window by clicking **<FVS Input Data>** button from the side menu.
2. The top section of the interactive window summarizes the BioSum project up to this point, showing a list of variant/Rx package combinations currently loaded into the project. Two interactive hyperlink buttons (Treatments and Packages) can be used to jump back to previously completed steps, if needed. To revisit these steps, simply click the corresponding button.



Treatments 2 Packages 2

Data Directory D:\biosum_projects\JaJoSi_59\fvs\data

	Variant	Package	Stands	Trees	FVS Output DB	SUMMARY RECS	CUTLIST RECS	TREELIST R
☐	BM	001						
☐	BM	999						
☐	CA	001						
☐	CA	999						
☐	EC	001						
☐	EC	999						
☐	NC	001						
☐	NC	999						
☐	SO	001						
☐	SO	999						

Path to the FIA Datamart input SQLite database

Selected Group All_FIA_Conditions

Figure 4.12 – Create FVS Input window.

3. **FVSIn Options tab:** The controls in this dialog affect what data is brought into the FVS input database files, which can significantly affect FVS simulation results. It is up to the analyst to understand how to tell FVS whether and how to use the information that ends up in this database. FVSIn options should be selected as needed at this point before proceeding further with the generation of FVSIn files.

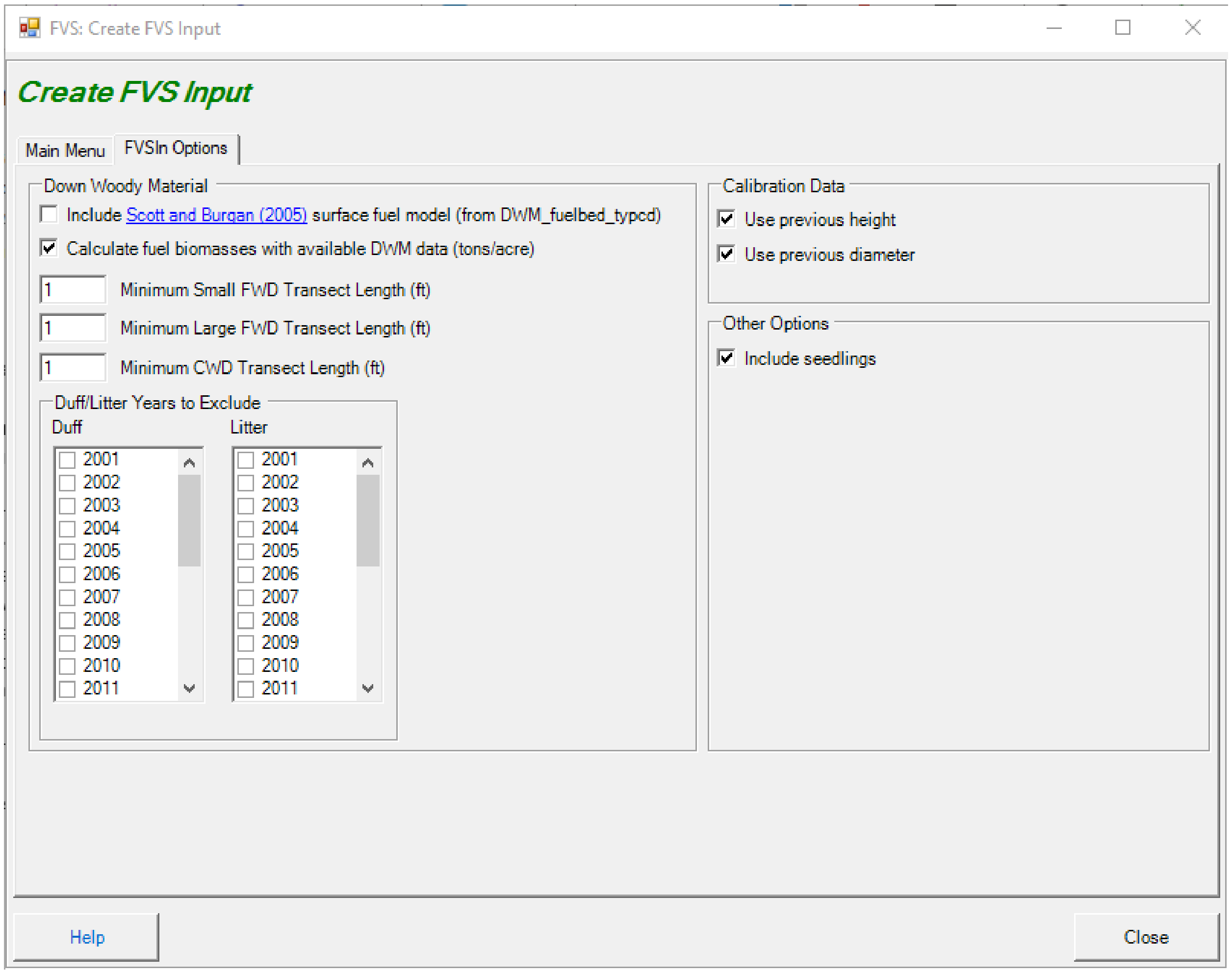
   For example, one can choose to rely on the surface fuel model recorded by FIA field crews (for the years it is available—2013 and later) or the calculated down wood biomass. The surface fuel model box is unchecked by default because if surface fuel is provided to FVS in the FVS_STANDINIT_COND table, FVS will rely on it and ignore any down wood data provided. Given that the down wood data are the basis of data in the CARBON table, omitting the down wood data from the FVSIn database could produce unwanted outcomes (i.e., carbon estimates based on broad region-wide averages calculated as FVS defaults rather than down wood quantities observed on the FIA plot).

   Down wood data can be filtered by transect length—for example, if a forested condition occurs on only one subplot, and on only a part of that subplot, the coarse wood transect length could be as short as 5 feet or less—a VERY small sample on which to base surface fuel biomass estimates, so the analyst might choose to set a higher threshold and pass coarse wood data to FVS only for conditions (stands) where the transect length on which down wood was sampled, exceeds that threshold. It's advisable to consult the field guides for each year of data you are interested in using, and/or the down wood tables, and/or an FIA analyst, for assistance in making these choices. The number, orientation and length of down wood transects has changed repeatedly over the past two decades. The default thresholds for minimum transect length are set at 1 foot because recent analysis in western Oregon revealed that almost any FIA observed measurement of down wood led to FVS assigned surface fuel models that produced predicted fire outcomes (under FFE) that better matched actual tree mortality outcomes than FVS assignments made without benefit of that measured down wood information.

Early in the annual inventory (2001-2003), confusion about duff and litter protocols (some due to the use of tenths of inches as a unit) leaves many analysts with little confidence that they are consistently and correctly recorded. These exclusion checkboxes (unchecked by default) provide the option to exclude loading of duff and/or litter data (which are used to represent fuels and forest floor carbon) for years where data is suspect (in which case FVS will assign what it "thinks" is a viable default for duff and litter mass for those years).

Checking the calibration data boxes (checked, by default) for previous height and/or diameter will result in those values being loaded for every tree for which they are populated in the FIADB database (i.e., trees present at the previous inventory visit). These can be found in the FVSIn database's FVS_TREEINIT_COND table's tree-level height and diameter growth fields (FVS_TREEINIT_DG; FVS_TREEINIT_HTG). When these values are populated, FVS will use them to calibrate its species-level growth models towards values empirically observed within each stand, with previous height data used to calibrate small tree growth (saplings) and previous diameter data used to calibrate growth of trees larger than 5 inches diameter. Under default FVS assumptions, at least 5 trees per size class and species, with remeasurement data available, are required to calibrate the small tree height or large tree diameter growth models. The CALBSTAT keyword can be used in FVS to change the minimum number of samples (trees with remeasurement data available) required to calibrate the growth models. Generally, including calibration data should increase the accuracy of FVS's growth projections, but it is up to the analyst to make this decision and to alter any related calibration parameters as appropriate (e.g., via the CALBSTAT keyword).

Finally, checking the "Include seedlings" box will populate the FVS input database files with seedling records from the inventory data. Seedlings (and saplings) are sampled on a plot size that is less than a tenth the size used for trees over 5 inches in diameter. While this may be adequate for characterizing densities of the smallest trees over a large landscape, it is arguably less so for representing them at stand level – the most granular scale on which BioSum evaluates management outcomes. It is fine to include them, but important to realize that they can generate apparently anomalous results—for example, 7 seedlings observed on the 6.8 foot radius microplot on one subplot for a condition that occupies only a quarter of the full plot will translate to a total of 2100 trees per acre (7 X 75)/(0.25). If many of these are modeled by FVS to survive and grow, this could translate to quite a high density of ladder fuels within a decade or two, so the analysis would depend very much on the accuracy with which FVS can do this. However, choosing not to include the seedling data may underestimate ladder fuel density (and the fire effects that would result) and also understate competition effects on growth, especially over longer simulations (e.g., >20 years).


FVS: Create FVS Input

Create FVS Input

Main Menu | FVSIn Options

Down Woody Material
- [ ] Include Scott and Burgan (2005) surface fuel model (from DWM_fuelbed_typcd)
- [✓] Calculate fuel biomasses with available DWM data (tons/acre)

1 Minimum Small FWD Transect Length (ft)
1 Minimum Large FWD Transect Length (ft)
1 Minimum CWD Transect Length (ft)

Duff/Litter Years to Exclude

Duff	Litter
[ ] 2001	[ ] 2001
[ ] 2002	[ ] 2002
[ ] 2003	[ ] 2003
[ ] 2004	[ ] 2004
[ ] 2005	[ ] 2005
[ ] 2006	[ ] 2006
[ ] 2007	[ ] 2007
[ ] 2008	[ ] 2008
[ ] 2009	[ ] 2009
[ ] 2010	[ ] 2010
[ ] 2011	[ ] 2011

Calibration Data
- [✓] Use previous height
- [✓] Use previous diameter

Other Options
- [✓] Include seedlings

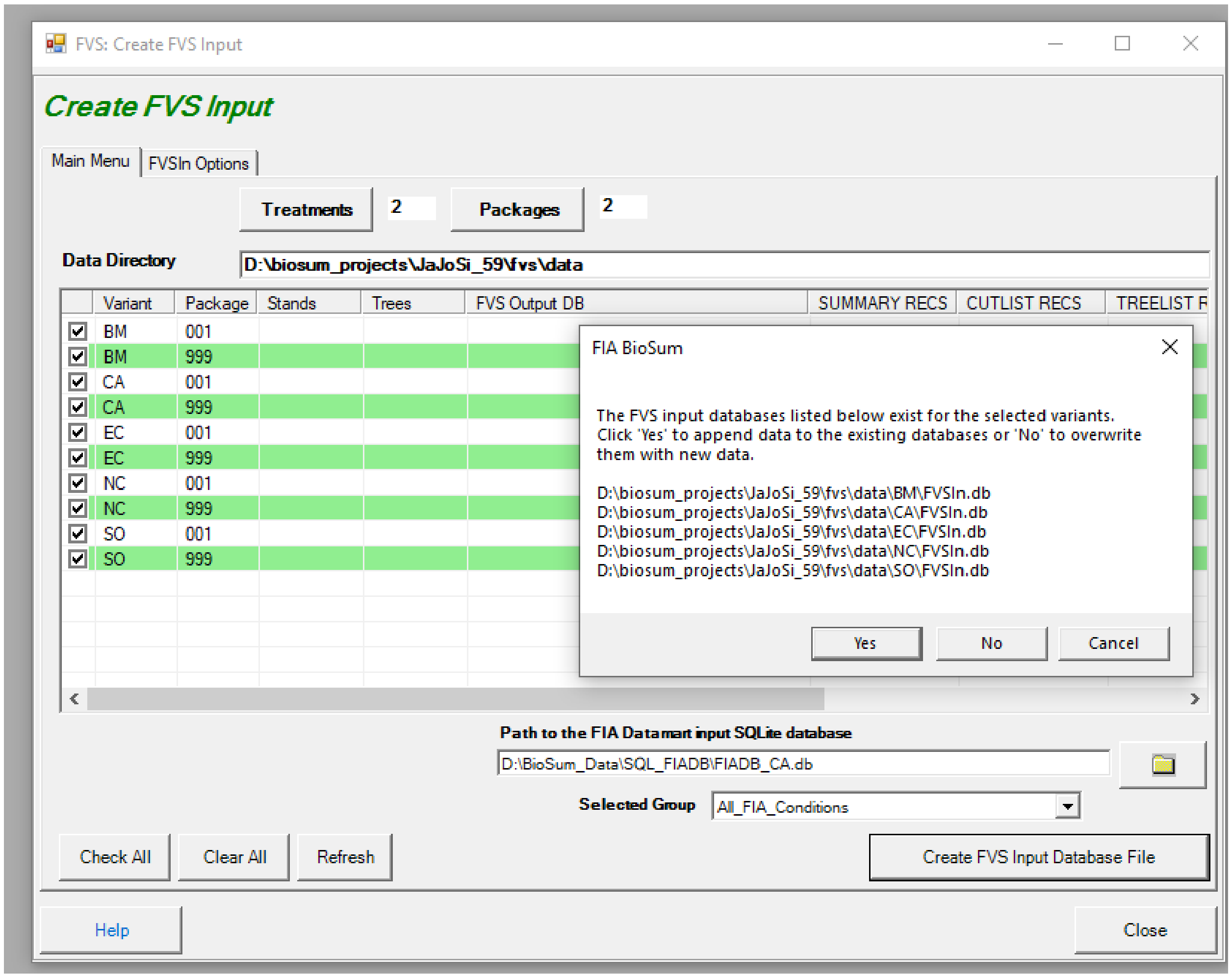
Help | Close

Figure 4.12.1 – The FVSIn options dialog for controlling how and whether field observations of down wood, surface fuel model, seedlings and measurements from previous visits (GRM data) are loaded to the FVSIn database for use by FVS.

4. Click on the Main Menu tab to return to the Rx package selections. Click <**Check All>** (in the lower left corner) to select all variant/Rx package combinations or select the desired packages by clicking on the check box next to each variant/package

combination. Stands (FIA conditions) that do not belong to one of the selected combinations will not be included in the FVS input files that BioSum generates.

5. Click the “Selected Group” dropdown menu and select the ALL_FIA_CONDITIONS option.
6. Click on the file button under “Path to the FIA Datamart input SQLite database” and navigate to the location of the state specific FIADB SQLite (.DB) database that was used to load FIA plots to the project under the already-completed Database module workflow. Click the <Create FVS Input Database File> button to initiate the input database file creation process. When the process is completed, the columns next to each variant/Rx package will populate.
    a. If FVS input databases already exist in a project, they can be either overwritten (replaced) or appended to. We recommend overwriting the files, unless the area of interest includes multiple states, in which case follow the directions below to append.
    b. For a multi-state project, step 6 must be repeated for each state. Starting with the 2nd state, use the append option (see example of dialog box below for choosing between the append [yes] and overwrite [no] options). Appending a state to an FVSIn will increase the record counts for Stands and Trees; if these don’t change, that may be a sign of a problem.



Treatments 2 Packages 2

Data Directory D:\biosum_projects\JaJoSi_59\fvs\data

	Variant	Package	Stands	Trees	FVS Output DB	SUMMARY RECS	CUTLIST RECS	TREELIST R
☑	BM	001						
☑	BM	999						
☑	CA	001						
☑	CA	999						
☑	EC	001						
☑	EC	999						
☑	NC	001						
☑	NC	999						
☑	SO	001						
☑	SO	999						

FIA BioSum

The FVS input databases listed below exist for the selected variants. Click 'Yes' to append data to the existing databases or 'No' to overwrite them with new data.

D:\biosum_projects\JaJoSi_59\fvs\data\BM\FVSIn.db
D:\biosum_projects\JaJoSi_59\fvs\data\CA\FVSIn.db
D:\biosum_projects\JaJoSi_59\fvs\data\EC\FVSIn.db
D:\biosum_projects\JaJoSi_59\fvs\data\NC\FVSIn.db
D:\biosum_projects\JaJoSi_59\fvs\data\SO\FVSIn.db

Path to the FIA Datamart input SQLite database D:\BioSum_Data\SQL_FIADB\FIADB_CA.db

Selected Group All_FIA_Conditions

FVS: Forest Vegetation Simulator

The 2nd module in the BioSum workflow facilitates simulation of silvicultural sequences with the Forest Vegetation Simulator (FVS), accounting for growth and silvicultural management. These instructions assume that the user is running the FVS growth model through the FVSOnline interface. All documentation and the most recent version of FVSOnline can be found on the FVS website (http://www.fs.fed.us/fmsc/fvs/software/complete.php).

Some caveats demand consideration before beginning the FVS workflow. FVS is a complex and highly customizable application that will generate much more realistic predictions when operated by an experienced user. Model performance can be greatly improved by customizing settings and/or calibrating models to more closely reflect inventoried stand conditions, rather than relying on defaults. For more information on modifying model performance by establishing baseline trends, refer to Vandendriesche (2010).

In addition to calibration and setting customization, users may want to account for regeneration in FVS growth projections if it is important to obtain accurate estimates of, for example, ladder fuels and the sustainability of stand structure as stands respond to treatments that open the forest canopy. Currently, the only areas for which FVS provides a “full” establishment model are eastern Montana, central and northern Idaho, and coastal Alaska. The remaining states are equipped with only a “partial” establishment model, meaning that only regeneration from stump resprouting or analyst-defined tree establishment (natural or planted) is included in the growth projections. Vandendriesche (2009) developed a method for estimating and including natural tree establishment within FVS simulations, but that method relied on an exogenous program application that is now defunct. An alternative natural tree regeneration imputation procedure that does not rely on exogenous program(s) has recently been developed. See Appendix C for an overview of how to include regeneration in FVS growth projections using this method.

Strategies for Specifying FVS Cycles and Intervals

The analyst running BioSum is responsible for understanding how both FVS and BioSum “understand” simulation cycles. The BioSum side of this is fairly straightforward, in that the program supports 5- and 10-year cycles (which can also be thought of as projection periods, or intervals), though <u>only one BioSum cycle length should be selected for a project</u>. As described in the introduction to this module, for simplicity, BioSum’s developers have always instructed FVS, via the KCP file, to start all stands at a nominal year 1, regardless of the year in which the data was collected; any other choice is likely to result in profound complexity and analyst confusion. This is accomplished with the InvYear 1 code as follows (with the “1” in column 20):

```
!!**************************************************
!!**Number Of Cycles and Years Between Cycles
!!**************************************************

InvYear           1
```

We also recommend a “throwaway” year (a 1-yr projection without management) to allow for obtaining a pre-treatment value of stand state variables from FFE patterned tables such as POTFIRE, CARBON and FUELS. Following that 1-yr projection, from the BioSum perspective, choosing 5-year cycles provides an opportunity for treatment to occur (depending on how the KCP file is coded) at any or all of year 2, 7, 12 and 17; choosing 10-year cycles provides for treatment implementation at years 2, 12, 22 and 32. As currently programmed, BioSum generates a kcp template with these parameters:

```
InvYear     1
Timeint     1     1 # PRE for FFE; “throwaway” grow-only
Timeint     2     1 # treat, grow 1yr, PRE for non-FFE
Timeint     3     8 # POST for all tables, grow to next PRE
Timeint     4     1 # PRE for FFE tables
Timeint     5     1 # treat, grow 1yr, PRE for non-FFE
Timeint     6     8 # POST for all tables, grow to next PRE
Timeint     7     1
Timeint     8     1
Timeint      9     8
Timeint     10     1
Timeint     11     1
Timeint     12     8
Timeint     13     1 # Needed for FFE output at year 41
NUMCYCLE      13
```

A 5-year cycle length would require a different set of keyword code, with the same general principles and a 1 1 3 repeating pattern instead of a 1 1 8. This pattern has the greatest flexibility because it arranges for FVS output to include PRE and POST treatment stand state values (such as metrics of strand volume and density, canopy cover, crown fire potential, carbon pools, etc.) for each of 4 BioSum simulation cycles (not to be confused with FVS projection cycles—the above example includes 13 of those), each with the possibility of implementing management activity. If no management activity needs to be modeled beyond BioSum cycle 1, a simpler code is possible, for example:

```
InvYear     1
Timeint     1     1 # PRE for FFE; “throwaway” grow-only
Timeint     2     1 # treat, then grow 1yr to post
Timeint     3     8 # →10 Years post-treatment
Timeint     4    10 # →20 Years post-treatment
Timeint     5    10 # →30 Years post-treatment
Timeint     6    10 # →40 Years post-treatment
NUMCYCLE      6
```

Any simulation for which it is important to know the state of the stand both before and immediately after treatment and that includes plans for use of output from FFE tables in that representation requires a pattern that includes back-to-back 1-year projections. If instead of the 6 FVS cycle pattern above a pattern of 10 10 10 10 were used with management occurring in the first FVS projection cycle (i.e., at year 1), then the FFE tables would contain no output describing pre-treatment stand conditions and one would need to extract different sets of columns (from the Summary table) or rows (from the STRCLASS table) to extract both pre and post treatment stand state from the non-FFE tables, which is not how the BioSum workflow is structured. Using the standard BioSum workflow, only the pre-treatment state in non-FFE tables would be loaded for each cycle, so if management were implemented in cycle 1, you would not see its effect on the state of the stand in those tables until 10 years of projected regrowth had occurred.

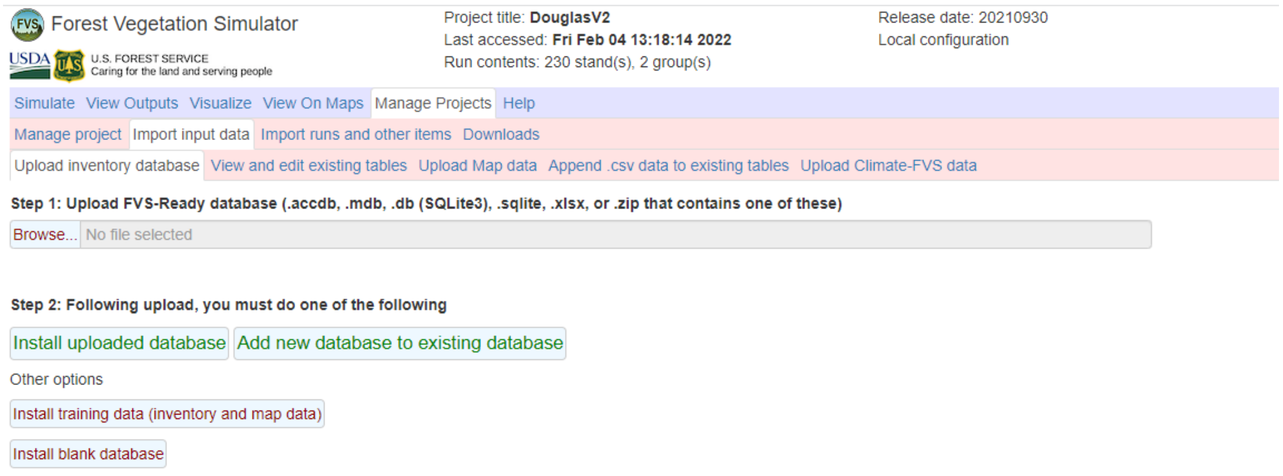
Ultimately, the choice of projection timing, made via edits to the KCP, is the responsibility of the analyst. The admittedly complex timing specifications are a result of a wrinkle in FFE’s architecture that causes the reported values for any given year in those tables, including stalwarts typically used in BioSum analyses that track crown fire potential, fuels and carbon, are calculated **after** any harvest has occurred and **before** any surface fuel treatment (e.g., via pile and burn or prescribed fire) has taken place.

In earlier versions of BioSum, the challenges posed by FFE's odd architecture were addressed by running a separate, 1-yr "base year" projection to generate a POTFIRE table without management and that output was then prepended to the POTFIRE table output from the main simulations as year 1, with the year attribute incremented by 1 for all rows in those tables and FVS output loading was modified accordingly. Though effective, this workflow was confusing and difficult to explain, and was only implemented for POTFIRE (not CARBON, FUELS or any of the other FFE tables) so we no longer recommend it. Instead, we recommend the throwaway year (1 year grow-only projection before commencing management simulation), which is effective for all FFE tables. The legacy BASE YEAR approach remains temporarily as a legacy option that is lightly documented in the appendix for those with legacy BioSum projects on which BASE YEAR was implemented.

Simulating Silvicultural Sequences (Rx Packages) in FVSOnline*

After FVS variants have been assigned, prescriptions and sequences defined, tree species mapping confirmed, FVS Input Data created, FVS calibration/regeneration considered, and POTFIRE base year values generated (if applicable), it is finally time to run the FVS growth projections for each silvicultural sequence and each variant in a project. Each combination of silvicultural sequence (aka Rx package) and variant, is referred to as a "run" in FVS terminology. This section describes major steps and best-practices associated with FVSOnline that are relevant to BioSum users. It is not an exhaustive guide to the FVSOnline interface for parameterizing and producing FVS outputs.

Loading Data
Once an FVSOnline project has been created, use the heading tabs to navigate to Manage Project < Import input data < Upload inventory database. Click the browse button to navigate to the location of your BioSum generated FVS input file (project/fvs/variant/FVSin.db) then click the "Install uploaded database" button. If your project includes multiple variants, additional variant specific FVSin files can be loaded into the project one at a time. To do so, navigate to the location of the next FVSin file using the browse button, and this time click the "Add new database to existing database" button instead. Continue this process for any additional variant specific FVSin files. You can always check what data is loaded into FVS at any time by consulting the View and edit existing tables tab.


Forest Vegetation Simulator

USDA U.S. FOREST SERVICE Caring for the land and serving people

Project title: **DouglasV2**
Last accessed: **Fri Feb 04 13:18:14 2022**
Run contents: 230 stand(s), 2 group(s)

Release date: 20210930
Local configuration

Simulate View Outputs Visualize View On Maps Manage Projects Help
Manage project Import input data Import runs and other items Downloads
Upload inventory database View and edit existing tables Upload Map data Append .csv data to existing tables Upload Climate-FVS data

Step 1: Upload FVS-Ready database (.accdb, .mdb, .db (SQLite3), .sqlite, .xlsx, or .zip that contains one of these)

Browse... No file selected

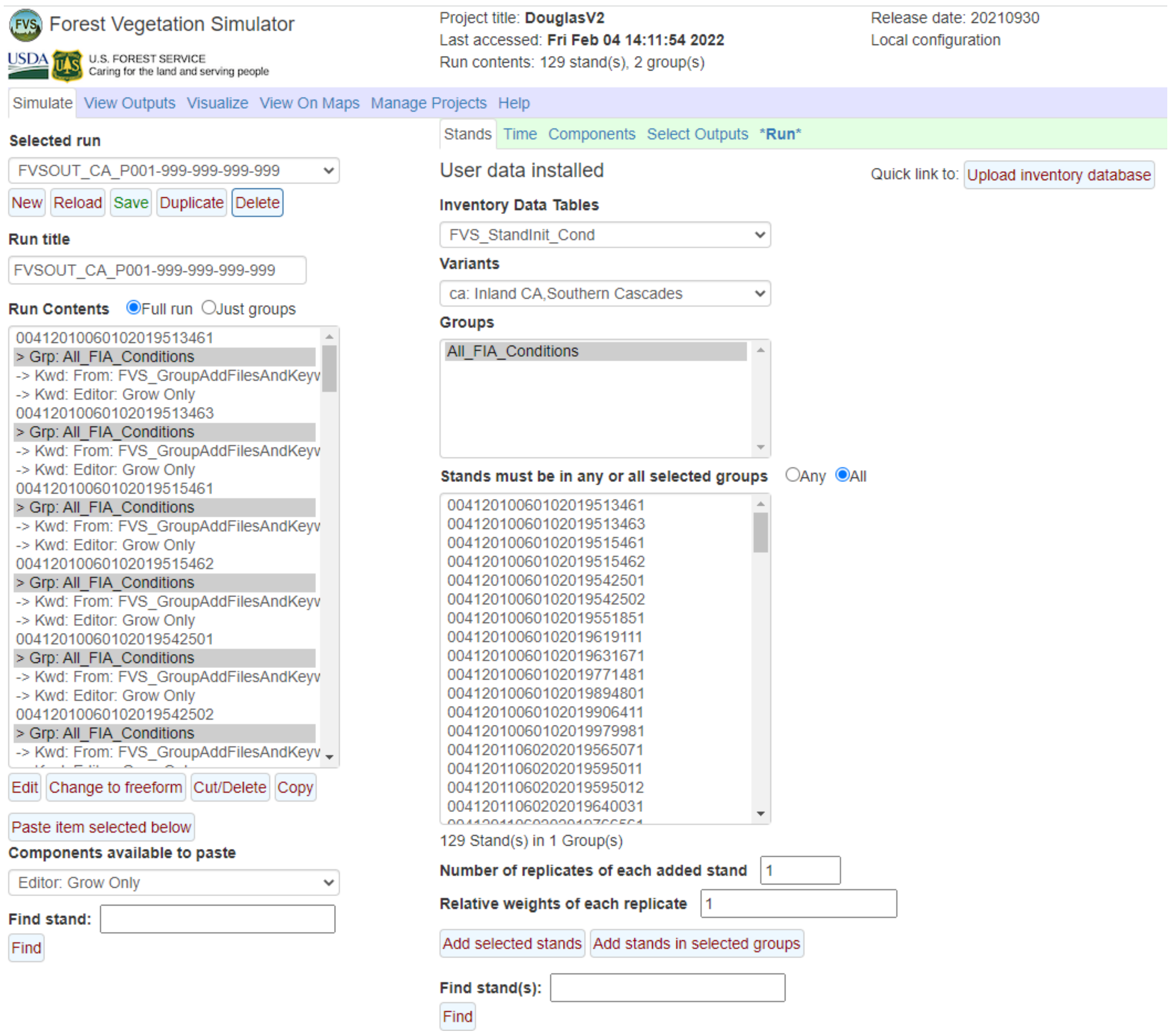
Step 2: Following upload, you must do one of the following

Install uploaded database
Add new database to existing database
Other options
Install training data (inventory and map data)
Install blank database

Figure 4.22 - The FVSOnline interface used to load (import) FVS input databases generated from BioSum

Crafting and Simulating FVS Runs (Variant-Specific Rx Packages)

To generate FVS outputs that can be loaded into and used by a BioSum project, FVS runs must be crafted and simulated for each variant/Rx package combination, to match the Rx packages defined in your BioSum project. This process is completed in the “Simulate” tab of FVSOnline.


Forest Vegetation Simulator
U.S. FOREST SERVICE
Caring for the land and serving people

Project title: **DouglasV2**
Last accessed: **Fri Feb 04 14:11:54 2022**
Run contents: 129 stand(s), 2 group(s)

Release date: 20210930
Local configuration

Simulate | View Outputs | Visualize | View On Maps | Manage Projects | Help

Selected run
FVSOUT_CA_P001-999-999-999-999
New | Reload | Save | Duplicate | Delete

Run title
FVSOUT_CA_P001-999-999-999-999

Run Contents [✓ Full run] Just groups

```
00412010060102019513461
> Grp: All_FIA_Conditions
-> Kwd: From: FVS_GroupAddFilesAndKeyv
-> Kwd: Editor: Grow Only
00412010060102019513463
> Grp: All_FIA_Conditions
-> Kwd: From: FVS_GroupAddFilesAndKeyv
-> Kwd: Editor: Grow Only
00412010060102019515461
> Grp: All_FIA_Conditions
-> Kwd: From: FVS_GroupAddFilesAndKeyv
-> Kwd: Editor: Grow Only
00412010060102019515462
> Grp: All_FIA_Conditions
-> Kwd: From: FVS_GroupAddFilesAndKeyv
-> Kwd: Editor: Grow Only
00412010060102019542501
> Grp: All_FIA_Conditions
-> Kwd: From: FVS_GroupAddFilesAndKeyv
-> Kwd: Editor: Grow Only
00412010060102019542502
> Grp: All_FIA_Conditions
-> Kwd: From: FVS_GroupAddFilesAndKeyv
```

Edit | Change to freeform | Cut/Delete | Copy

Paste item selected below
Components available to paste
Editor: Grow Only
Find stand:
Find

Stands | Time | Components | Select Outputs | ***Run***

User data installed

Quick link to: Upload inventory database

Inventory Data Tables
FVS_StandInit_Cond
Variants
ca: Inland CA,Southern Cascades
Groups
All_FIA_Conditions

Stands must be in any or all selected groups Any [✓ All]

```
00412010060102019513461
00412010060102019513463
00412010060102019515461
00412010060102019515462
00412010060102019542501
00412010060102019542502
00412010060102019551851
00412010060102019619111
00412010060102019631671
00412010060102019771481
00412010060102019894801
00412010060102019906411
00412010060102019979981
00412011060202019565071
00412011060202019595011
00412011060202019595012
00412011060202019640031
```

129 Stand(s) in 1 Group(s)
Number of replicates of each added stand 1
Relative weights of each replicate 1
Add selected stands | Add stands in selected groups

Find stand(s):
Find

Figure 4.23 - The Simulate tab of FVSOnline, where runs (i.e., the variant/Rx package combinations defined in BioSum) are crafted and simulated.

Naming Runs

How runs are named in FVS is extremely important for BioSum post-processing and must follow a specific naming convention, else you will incur errors when attempting to load FVS output data back into BioSum. The naming convention is as follows:

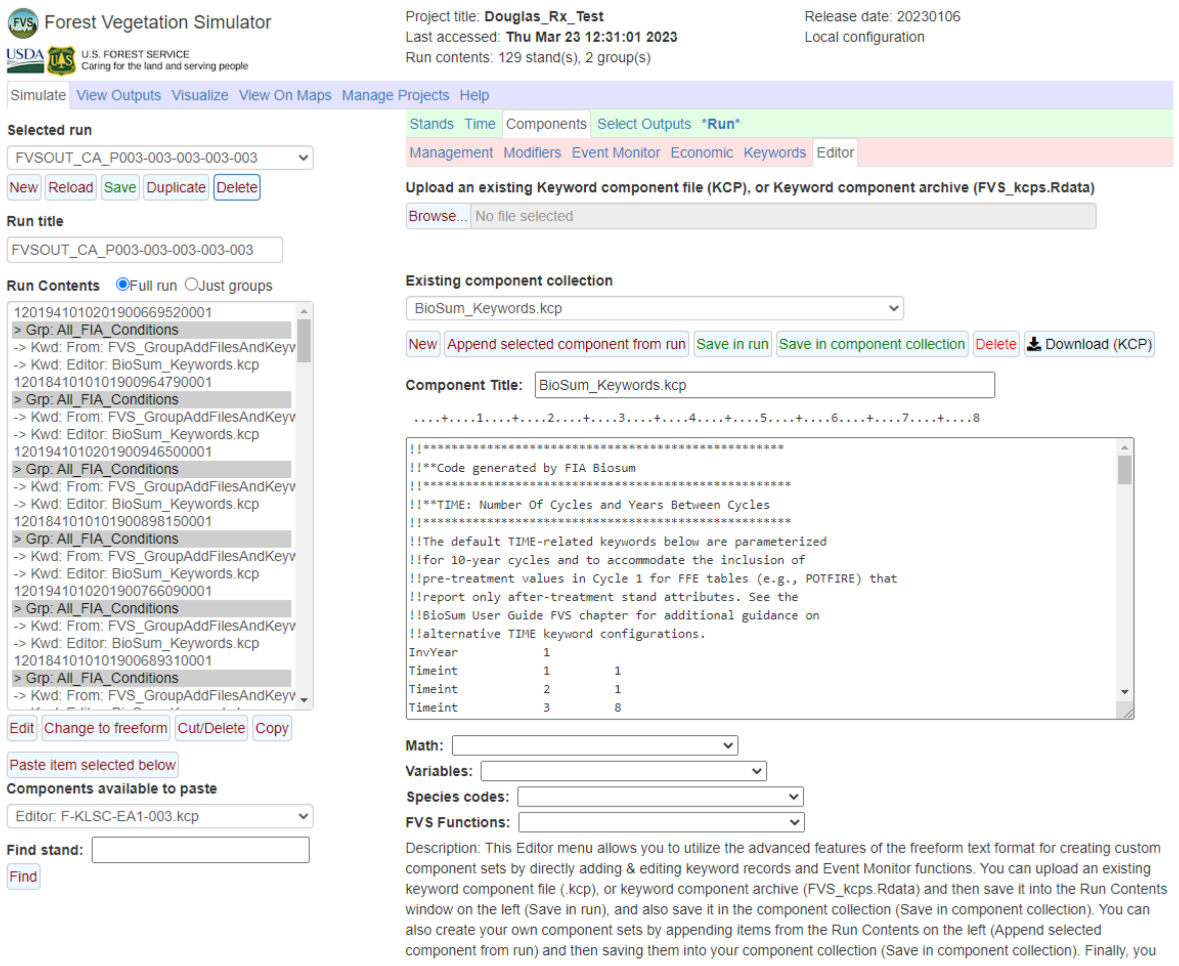
FVSOUT_(variant)_P(Rx package #)-(cycle 1 treatment #)-(cycle 2 treatment #)-(cycle 3 treatment #)-(cycle 4 treatment #)

A valid naming example would look like: FVSOUT_CA_P001-110-999-999-999
Where CA refers to the CA variant, P001 refers to Rx package 001, 110 refers to a thin-from below treatment in cycle 1, and 999 refers to do nothing (grow-only) for the remaining cycles 2, 3, and 4.

Type the run title name under “Run title” and click the “Save” button. A new run has been added to your FVS project and is ready to have input data appended to it. To add stands to a run, select the desired variant from the “Variant” drop-down menu, ensuring “All_FIA_Conditions” from the Groups box is highlighted. All stands (conditions) associated with the selected variant will be populated into the Stands table. Highlight specific stands you want to import to the run and click the “Add selected stands” button. The specified stands will be added to the Run Contents table. You are now ready to craft and append FVS keywords to stands that simulate the Rx treatments associated with your FVS run. You will also need to append a basic set of BioSum-related keywords to maintain compatibility of the FVS outputs with BioSum.

Adding Keywords to Runs

FVS keywords can be added to stands using two methods, (1) via GUI-guided command tables in the Components < Management, Modifiers, Event Monitor, Economic, and Keywords tabs or (2) via the Components < Editor tab, which allows for free-form FVS keyword scripting. Before applying Rx-related keyword to scripts, ensure BioSum-related keywords are first appended via the Keyword Editor. Click on the Components < Editor tab and click the <Browse> button under the “Upload an existing Keyword component file (KCP), or Keyword component archive (FVS_kcps.Rdata) header. Navigate to and select the BioSum Keywords KCP file saved in your project directory: project/fvs/data/BioSum_Keywords.kcp, then give the Component Title a name (e.g., BioSum_Keywords). The keyword component can be saved for later (useful when appending the same keyword script to many different runs) by clicking the <Save in component collection> button. The component can then be copied to all stands, by first ensuring all stands are highlighted from the Run Contents window, then clicking the <Save in run> button.


Forest Vegetation Simulator
USDA U.S. FOREST SERVICE Caring for the land and serving people
Simulate View Outputs Visualize View On Maps Manage Projects Help
Stands Time Components Select Outputs *Run*
Management Modifiers Event Monitor Economic Keywords Editor
New Reload Save Duplicate Delete
Browse...
New Append selected component from run Save in run Save in component collection Delete Download (KCP)
Edit Change to freeform Cut/Delete Copy
Paste item selected below
Find

Project title: **Douglas_Rx_Test**
Last accessed: **Thu Mar 23 12:31:01 2023**
Run contents: 129 stand(s), 2 group(s)

Release date: 20230106
Local configuration

Selected run
FVSOUT_CA_P003-003-003-003-003

Run title
FVSOUT_CA_P003-003-003-003-003

Run Contents [✓ Full run] Just groups

```
1201941010201900669520001
> Grp: All_FIA_Conditions
-> Kwd: From: FVS_GroupAddFilesAndKeyw
-> Kwd: Editor: BioSum_Keywords.kcp
1201841010101900964790001
> Grp: All_FIA_Conditions
-> Kwd: From: FVS_GroupAddFilesAndKeyw
-> Kwd: Editor: BioSum_Keywords.kcp
1201941010201900946500001
> Grp: All_FIA_Conditions
-> Kwd: From: FVS_GroupAddFilesAndKeyw
-> Kwd: Editor: BioSum_Keywords.kcp
1201841010101900898150001
> Grp: All_FIA_Conditions
-> Kwd: From: FVS_GroupAddFilesAndKeyw
-> Kwd: Editor: BioSum_Keywords.kcp
1201941010201900766090001
> Grp: All_FIA_Conditions
-> Kwd: From: FVS_GroupAddFilesAndKeyw
-> Kwd: Editor: BioSum_Keywords.kcp
1201841010101900689310001
> Grp: All_FIA_Conditions
-> Kwd: From: FVS_GroupAddFilesAndKeyw
```

Components available to paste
Editor: F-KLSC-EA1-003.kcp

Find stand:

Upload an existing Keyword component file (KCP), or Keyword component archive (FVS_kcps.Rdata)
No file selected

Existing component collection
BioSum_Keywords.kcp

Component Title: BioSum_Keywords.kcp

```
....+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
!!**************************************************
!!**Code generated by FIA Biosum
!!***************************************************
!!**TIME: Number Of Cycles and Years Between Cycles
!!***************************************************
!!The default TIME-related keywords below are parameterized
!!for 10-year cycles and to accommodate the inclusion of
!!pre-treatment values in Cycle 1 for FFE tables (e.g., POTFIRE) that
!!report only after-treatment stand attributes. See the
!!BioSum User Guide FVS chapter for additional guidance on
!!alternative TIME keyword configurations.
InvYear         1
Timeint         1         1
Timeint         2         1
Timeint         3         8
```

Math:
Variables:
Species codes:
FVS Functions:

Description: This Editor menu allows you to utilize the advanced features of the freeform text format for creating custom component sets by directly adding & editing keyword records and Event Monitor functions. You can upload an existing keyword component file (.kcp), or keyword component archive (FVS_kcps.Rdata) and then save it into the Run Contents window on the left (Save in run), and also save it in the component collection (Save in component collection). You can also create your own component sets by appending items from the Run Contents on the left (Append selected component from run) and then saving them into your component collection (Save in component collection). Finally, you

Figure 4.24 - Using the keyword editor to copy BioSum keywords to all stands for a single run.

IMPORTANT: Specifying time correctly via select keywords is extremely important for BioSum post-processing, yet how it is specified will vary depending on the project, Rx packages, and whether POTFIRE tables are required. Consult the **Strategies for Specifying FVS Cycles and Intervals** section above. Those requiring documentation of legacy BASE YEAR implementation are advised to consult **BioSum User Guide Appendix D** and edit the "TIME: Number Of Cycles and Years Between Cycles" keywords as necessary. By default, the time keywords in the Biosum_Keywords.kcp file specify a 40-year simulation (four 10-year BioSum cycles). In this case, treatments should occur specifically in FVS cycles 2, 5, 8, 11 (simulation years 2, 12, 22, 32; BioSum cycles 1, 2, 3, 4).

Full documentation regarding FVS keywords and parameterization can be referenced in the Keyword Reference Guide for the Forest Vegetation Simulator (Van Dyck and Smith-Mateja 2000). FFE related keywords can be referenced in the Fire and Fuels Extension to the Forest Vegetation Simulator (Rebain 2010)

Specifying Output Tables

For each run, you will need to specify what information (tables) are output by FVS. This can be done by editing Output Table Keywords in the BioSum-Keywords.kcp. BioSum requires the FVS_Cutlist and FVS_Summary tables – these tables are produced by default. Consult FVS documentation on whether other output tables are useful for your analysis and edit the BioSum_Keywords.kcp file “Output Table Keywords” section appropriately. FVS output table keywords can be referenced in the User’s Guide to the Database Extension of the Forest Vegetation Simulator (Crookston 2003). Keep in mind that as more tables are selected for export, the larger the final FVS output file will be. This can drastically increase post-processing time spent in BioSum and occasionally lead to system memory issues when Access files become too large (e.g., >2gb). Thus, it is good practice to determine what tables (information) are needed for a BioSum analysis beforehand to prevent the export of unneeded tables and data.

Set additional harvest cost flags in FVS via the FVS_COMPUTE table (optional)

As explained in Step 2d of the treatment definition workflow and in the section on Additional CPA at the beginning of this chapter, when coding silvicultural treatments (Rx) in FVS via .kcp keyword files, any treatment components that incur operational costs that are not accounted for by OPCOST will need flags set (one per activity) in the FVS_COMPUTE table during the FVS simulation for BioSum to correctly track those costs. The FVS_COMPUTE table contains all user-defined and calculated variables. Examples of additional harvest costs that may need to be tracked include those associated with surface fuel treatments (such as mastication, prescribed fire or pile burning); mechanical site prep; application of herbicide, fertilizer, lime or biochar; water-barring skid trails; and planting or seeding. Recall that if additional CPA flags were specified with the treatment when it was defined, the activity flags were associated with the treatment as 8-character labels (that avoided spaces and special characters for compatibility with FVS conventions for variable names in COMPUTE).

These flags need to be set to 1 in the FVS_COMPUTE table for the record corresponding to the cycle/year in which an activity occurs, and to 0 otherwise (if the cost should not be incurred at that cycle). This is best accomplished via keyword statements integrated into the treatment keyword (kcp) scripts and/or as individual keyword addfiles appended to the simulation. At its most basic, such kcp code looks like:

```
COMPUTE        0
VARNAME = 1
END
```

Such flag-setting must be applied for each treatment component that incurs additional costs per acre (in essence, all of the additional CPA components defined for each Rx that is a component of the silvicultural sequence [package] represented by the kcp code being modified).

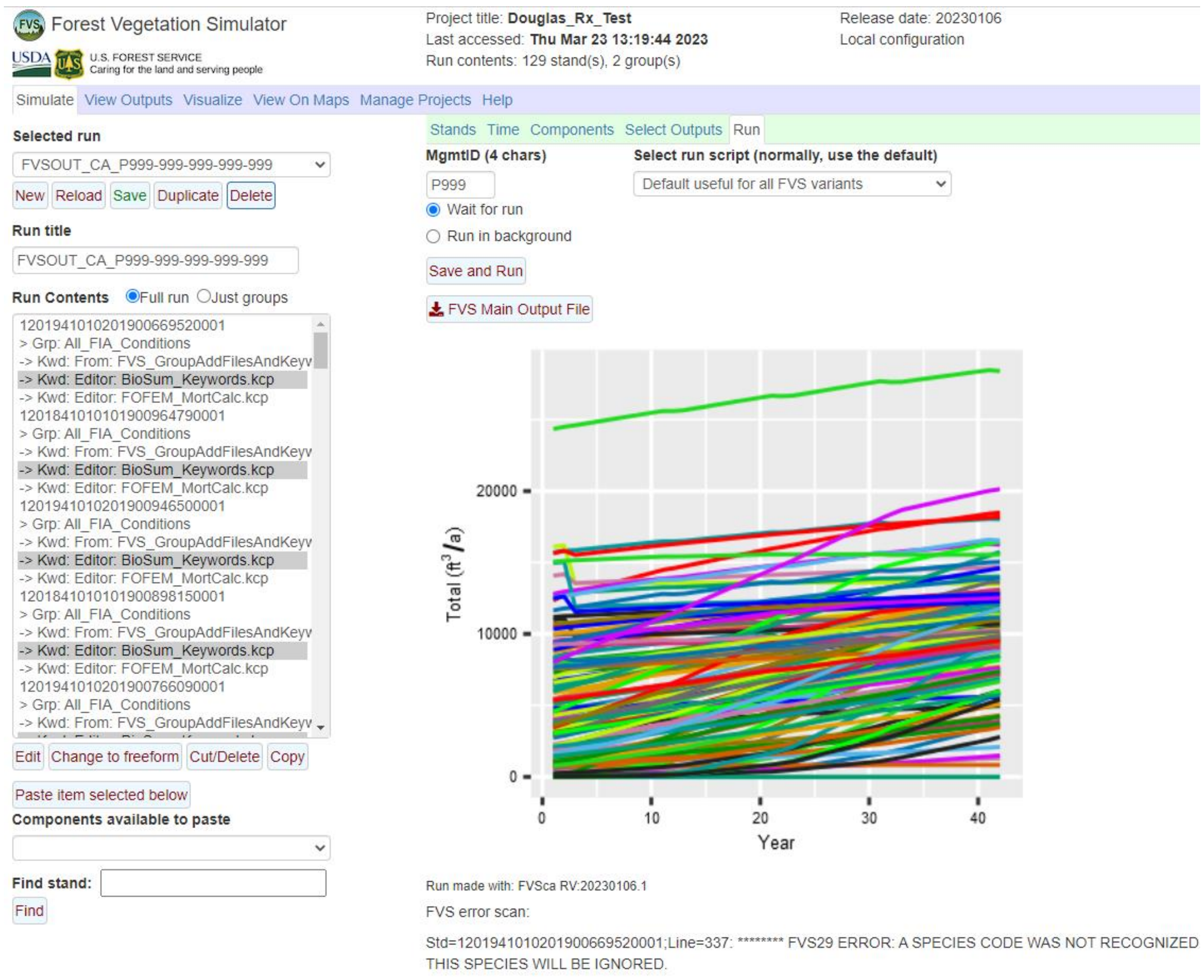
Below is an example where the component flag (broadcast burn; BRDBURN) keywords have been integrated into a treatment keyword (kcp) script. BRDBURN values are only set to 1 when the treatment is conditionally executed.

```
*Conditional statement triggers treatment if FVS cycle = 2
IF
CYCLE EQ 2
THEN
*The thinning (harvest) component of the treatment
YardLoss        0    PARMS(.05, 1, 0.1)
ThinSDI         0    PARMS(200, 1, All, 7, 30, 1)
*The broadcast burn component of the treatment
FMIN
SimFire         0    PARMS(3, 2, 55, 1, 100, 4)
FlameAdj        0    PARMS(1.0, 0.2)
FMortMlt        0    PARMS(0, All, 10, 999)
END
*Sets the BRDBURN variable value to 1 in the FVS_COMPUTE table when treatment is
applied
COMPUTE         0
BRDBURN = 1
END
ENDIF
```

Important Timing Note: The FVS_COMPUTE flag associated with a treatment component that incurs additional harvest costs must be set only for FVS cycles that correspond to BioSum cycles 1-4 in which management activity is scheduled to occur. These are the FVS cycles for which output is loaded into the pre-treatment registers of the Summary PREPOST table during the FVS Output Load using sequence number assignment). For example, if FVS cycles 2, 5, 8, and 11 are selected as the pre-treatment sequence numbers for the Summary output for BioSum cycles 1, 2, 3, and 4, the flag for broadcast burn must be activated in FVS cycles 2, 5, 8, or 11 (if, in fact, the additional CPA should be incurred for the associated BioSum cycles). The broadcast burn treatment itself can be programmed to occur in a different FVS cycle, however (e.g., the year/cycle immediately after tree harvest, or even later—what's essential is that the flag be set to 1 for a pre-treatment cycle).

Executing Run Simulations

Once a run is correctly named, has stands appended to it, and has Rx-related and BioSum related keywords appended to stands, the run is ready for simulation execution. Navigate to the *Run* tab. Ensure you save run parameters by clicking the Save button under the Selected Run dropdown menu. Optionally, give the run a Management ID (MgmtID) if desired. Runs can either be executed one at a time or batch processed. To run one at a time, ensure the "Wait for run" radio button is selected and click the Save and Run button. When this option is checked, the run will take longer to process, but will provide a list of potential FVS errors that occurred. This is useful for diagnosing whether FVS correctly read all input keywords and/or database values. To batch process, ensure the “Run in background” radio button is selected, select how many CPU cores should be used while processing (i.e., parallel processing) and click the Save and Run button. This option can significantly reduce processing time by utilizing parallel processing, assuming the user’s computer has many CPU cores. With this option selected, potential FVS errors will not be displayed, thus it is pragmatic to first use the “Wait for run” option to initially diagnose errors, then the “Run in background” option to maximize processing speed.


Forest Vegetation Simulator
USDA U.S. FOREST SERVICE Caring for the land and serving people
Simulate View Outputs Visualize View On Maps Manage Projects Help
Stands Time Components Select Outputs Run
New Reload Save Duplicate Delete
Edit Change to freeform Cut/Delete Copy
Paste item selected below
Find
Save and Run
FVS Main Output File
Total (ft³/a)
Year
20000
10000
0
0
10
20
30
40

Project title: **Douglas_Rx_Test**
Last accessed: **Thu Mar 23 13:19:44 2023**
Run contents: 129 stand(s), 2 group(s)
Release date: 20230106
Local configuration

Selected run
FVSOUT_CA_P999-999-999-999-999

Run title
FVSOUT_CA_P999-999-999-999-999

Run Contents [✓ Full run] Just groups

```
1201941010201900669520001
> Grp: All_FIA_Conditions
-> Kwd: From: FVS_GroupAddFilesAndKeyw
-> Kwd: Editor: BioSum_Keywords.kcp
-> Kwd: Editor: FOFEM_MortCalc.kcp
1201841010101900964790001
> Grp: All_FIA_Conditions
-> Kwd: From: FVS_GroupAddFilesAndKeyw
-> Kwd: Editor: BioSum_Keywords.kcp
-> Kwd: Editor: FOFEM_MortCalc.kcp
1201941010201900946500001
> Grp: All_FIA_Conditions
-> Kwd: From: FVS_GroupAddFilesAndKeyw
-> Kwd: Editor: BioSum_Keywords.kcp
-> Kwd: Editor: FOFEM_MortCalc.kcp
1201841010101900898150001
> Grp: All_FIA_Conditions
-> Kwd: From: FVS_GroupAddFilesAndKeyw
-> Kwd: Editor: BioSum_Keywords.kcp
-> Kwd: Editor: FOFEM_MortCalc.kcp
1201941010201900766090001
> Grp: All_FIA_Conditions
-> Kwd: From: FVS_GroupAddFilesAndKeyw
```

Components available to paste

Find stand:

MgmtID (4 chars)
P999

Select run script (normally, use the default)
Default useful for all FVS variants

[✓ Wait for run]
Run in background

Run made with: FVSca RV:20230106.1

FVS error scan:

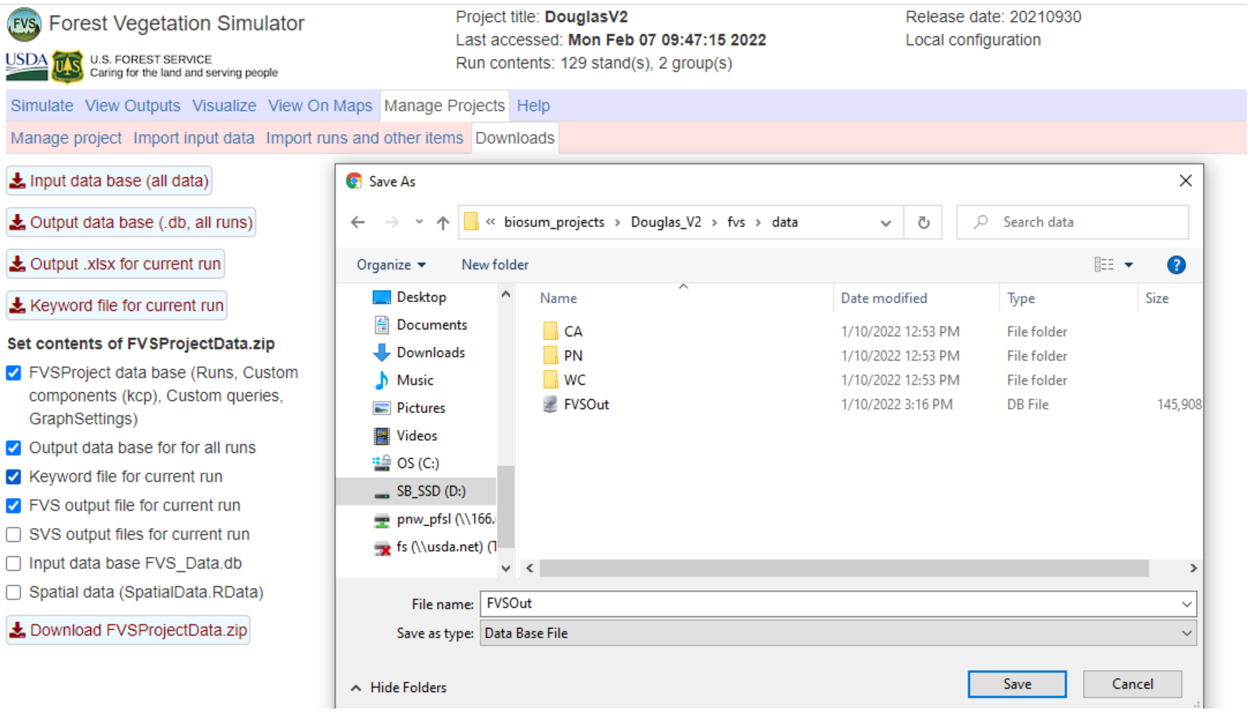
```
Std=1201941010201900669520001;Line=337: ******** FVS29 ERROR: A SPECIES CODE WAS NOT RECOGNIZED
THIS SPECIES WILL BE IGNORED.
```

Figure 4.26 – Executing a grow-only Rx package for the CA variant from the *Run* tab of the FVSOnline interface. In this case, a keyword error (species code not recognized) was detected and logged.

Exporting Output Data

Once all FVS runs (Rx package and variant combinations; POTFIRE base year and variant combinations) in a project have been simulated, the user is ready to export the FVS output data that will be loaded into BioSum. Navigate to the Manage Projects < Downloads tab and click the

“Output data base (.db, all runs)” button. Navigate to the project/fvs/data folder of your BioSum project. The output file must be named “FVSOut” and be saved in the project/fvs/data folder. Click save.


Forest Vegetation Simulator
USDA U.S. FOREST SERVICE Caring for the land and serving people
Simulate View Outputs Visualize View On Maps Manage Projects Help
Manage project Import input data Import runs and other items Downloads
Input data base (all data)
Output data base (.db, all runs)
Output .xlsx for current run
Keyword file for current run
Download FVSProjectData.zip
Save As
Organize New folder
Desktop
Documents
Downloads
Music
Pictures
Videos
OS (C:)
SB_SSD (D:)
pnw_pfsl (\\166.
fs (\\usda.net) (
Search data
Save
Cancel
Hide Folders

Project title: **DouglasV2**
Last accessed: **Mon Feb 07 09:47:15 2022**
Run contents: 129 stand(s), 2 group(s)

Release date: 20210930
Local configuration

Set contents of FVSProjectData.zip

- [✓] FVSProject data base (Runs, Custom components (kcp), Custom queries, GraphSettings)
- [✓] Output data base for for all runs
- [✓] Keyword file for current run
- [✓] FVS output file for current run
- [ ] SVS output files for current run
- [ ] Input data base FVS_Data.db
- [ ] Spatial data (SpatialData.RData)

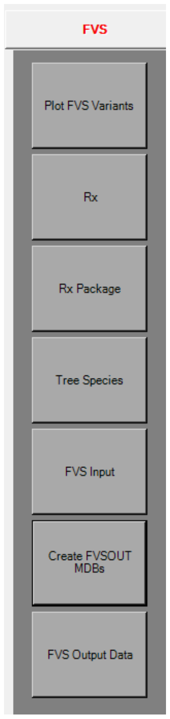
« biosum_projects › Douglas_V2 › fvs › data

Name	Date modified	Type	Size
CA	1/10/2022 12:53 PM	File folder	
PN	1/10/2022 12:53 PM	File folder	
WC	1/10/2022 12:53 PM	File folder	
FVSOut	1/10/2022 3:16 PM	DB File	145,908

File name: FVSOut
Save as type: Data Base File

Fig 4.27 - Location of the “Output data base (.db, all runs)” button used to export the FVS output file and location where the FVSOut.db file must be saved within the BioSum project directory.

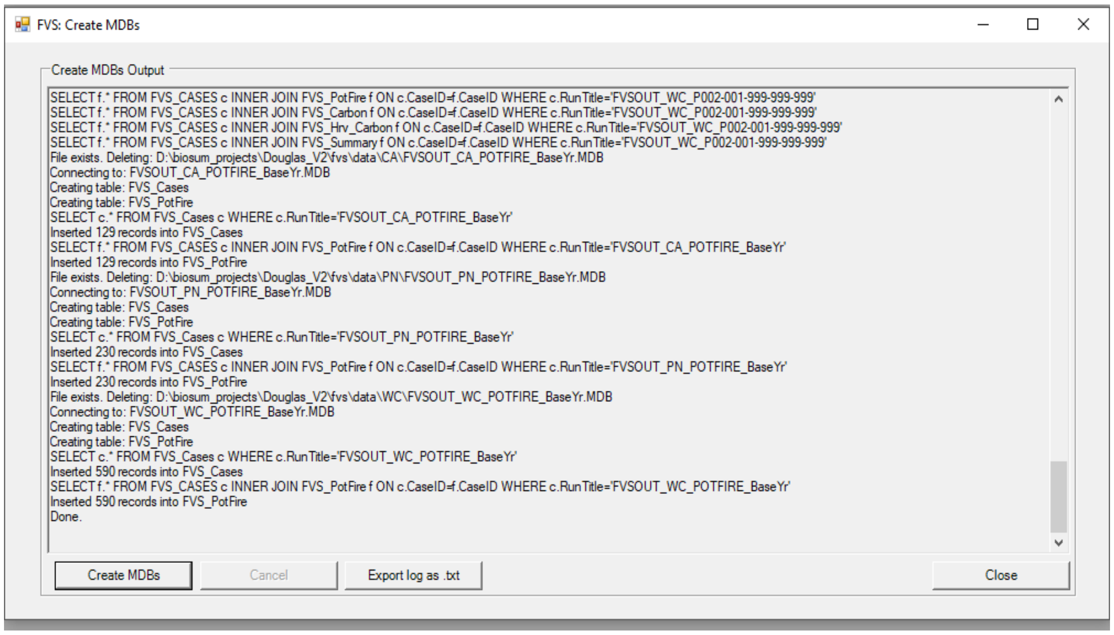
FVS Output Data



The last task in the BioSum’s FVS module reformat FVS output as input to the BioSum Processor and Treatment Optimizer modules. The “cut list” data (list of trees harvested in management operations) are used to calculate potentially available biomass and merchantable wood volume and value, and costs of implementing treatment packages; the other FVS outputs provide pre- and post-treatment stand attributes that can be a basis for evaluating management effectiveness. With the project open in BioSum Manager:

1. Click **<FVS>** on the left side-bar, then **<Create FVSOUT MDBs >.**
2. From the newly opened FVS: Create MDBs window, click the Create MDBs button. This process will write data from the single FVSOut.db file produced in the previous section into many .mdb (Access) databases. A .mdb database will be written and populated with data from each run executed in FVSOnline. Databases are stored in the project/fvs/data/variant subfolders. If this process has already been run previously in a project, databases will be overwritten rather than appended, preventing duplication of outputs. The process may take 1-30 minutes (or longer) to complete, depending on the size of a project (i.e.,

number of Rx packages, variants, and stands). The dialog box will actively track execution of SQL statements used to write databases; you may close the window when the dialog box displays “Done”.



```
SELECT f.* FROM FVS_CASES c INNER JOIN FVS_PotFire f ON c.CaseID=f.CaseID WHERE c.RunTitle='FVSOUT_WC_P002-001-999-999-999'
SELECT f.* FROM FVS_CASES c INNER JOIN FVS_Carbon f ON c.CaseID=f.CaseID WHERE c.RunTitle='FVSOUT_WC_P002-001-999-999-999'
SELECT f.* FROM FVS_CASES c INNER JOIN FVS_Hrv_Carbon f ON c.CaseID=f.CaseID WHERE c.RunTitle='FVSOUT_WC_P002-001-999-999-999'
SELECT f.* FROM FVS_CASES c INNER JOIN FVS_Summary f ON c.CaseID=f.CaseID WHERE c.RunTitle='FVSOUT_WC_P002-001-999-999-999'
File exists. Deleting: D:\biosum_projects\Douglas_V2\fvs\data\CA\FVSOUT_CA_POTFIRE_BaseYr.MDB
Connecting to: FVSOUT_CA_POTFIRE_BaseYr.MDB
Creating table: FVS_Cases
Creating table: FVS_PotFire
SELECT c.* FROM FVS_Cases c WHERE c.RunTitle='FVSOUT_CA_POTFIRE_BaseYr'
Inserted 129 records into FVS_Cases
SELECT f.* FROM FVS_CASES c INNER JOIN FVS_PotFire f ON c.CaseID=f.CaseID WHERE c.RunTitle='FVSOUT_CA_POTFIRE_BaseYr'
Inserted 129 records into FVS_PotFire
File exists. Deleting: D:\biosum_projects\Douglas_V2\fvs\data\PN\FVSOUT_PN_POTFIRE_BaseYr.MDB
Connecting to: FVSOUT_PN_POTFIRE_BaseYr.MDB
Creating table: FVS_Cases
Creating table: FVS_PotFire
SELECT c.* FROM FVS_Cases c WHERE c.RunTitle='FVSOUT_PN_POTFIRE_BaseYr'
Inserted 230 records into FVS_Cases
SELECT f.* FROM FVS_CASES c INNER JOIN FVS_PotFire f ON c.CaseID=f.CaseID WHERE c.RunTitle='FVSOUT_PN_POTFIRE_BaseYr'
Inserted 230 records into FVS_PotFire
File exists. Deleting: D:\biosum_projects\Douglas_V2\fvs\data\WC\FVSOUT_WC_POTFIRE_BaseYr.MDB
Connecting to: FVSOUT_WC_POTFIRE_BaseYr.MDB
Creating table: FVS_Cases
Creating table: FVS_PotFire
SELECT c.* FROM FVS_Cases c WHERE c.RunTitle='FVSOUT_WC_POTFIRE_BaseYr'
Inserted 590 records into FVS_Cases
SELECT f.* FROM FVS_CASES c INNER JOIN FVS_PotFire f ON c.CaseID=f.CaseID WHERE c.RunTitle='FVSOUT_WC_POTFIRE_BaseYr'
Inserted 590 records into FVS_PotFire
Done.
```

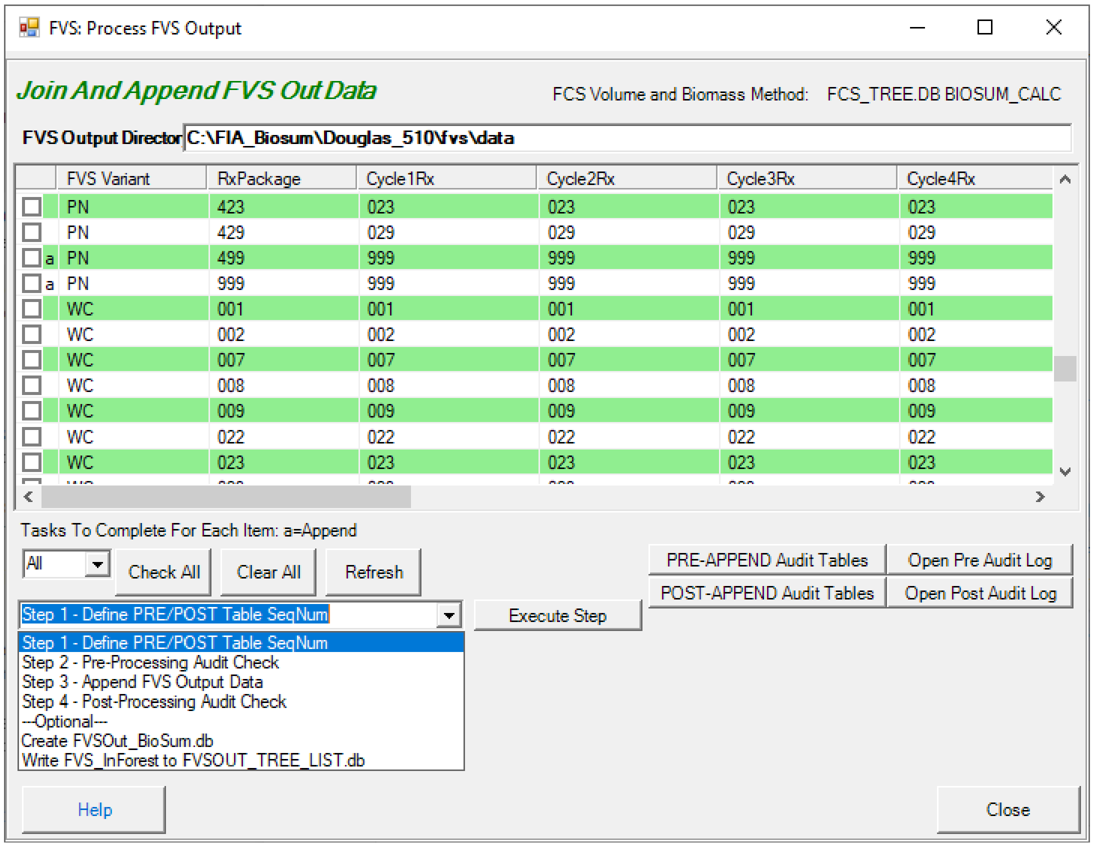
Figure 4.28 - The Create FVSOUT MDBs window in the BioSum interface with completed processing shown in the dialog box.

3. Next, click the **<FVS Output Data >** button from the left side of the BioSum interface; The Join and Append FVS Out Data window opens showing a list of all variant/Rx package combinations that have been modeled in FVS, similar to the window displayed when creating FVS input files. Additional information is also provided (viewable by scrolling to the right), such as the output database file name, Rx package specifics, whether the file was successfully found, and the number of records in the FVS summary, cut tree, standing tree, and potential fire tables.

   There are 4 required and 2 optional tasks managed here, with the required tasks to be executed in sequence. Only Steps 2-4 and the pre- and post-append and audit tables and logs buttons respect the status of the checkboxes adjoining FVS variant and package identifiers—the other selections in the drop down “step list” apply to all variants and packages.

   Defining the sequence number assignments (Step 1) builds the configuration tables (in [*project_name*]\db\fvsmaster.mdb) that control FVS output loading. See “Strategies for Specifying FVS Cycles and Intervals” earlier in this chapter for details on how to assign sequence numbers. Variant-package combinations showing an “a” next to the checkbox have not yet been appended or the source data in what was exported for that

combination from FVSOut contains a more recent time stamp than what is currently loaded in the BioSum data structures.
Manually select output databases to be appended or click <Check All> to select all variant/Rx package combinations, then click Execute Step (when Step 1 is highlighted).


FVS: Process FVS Output

Join And Append FVS Out Data

FCS Volume and Biomass Method: FCS_TREE.DB BIOSUM_CALC

FVS Output Director C:\FIA_Biosum\Douglas_510\fvs\data

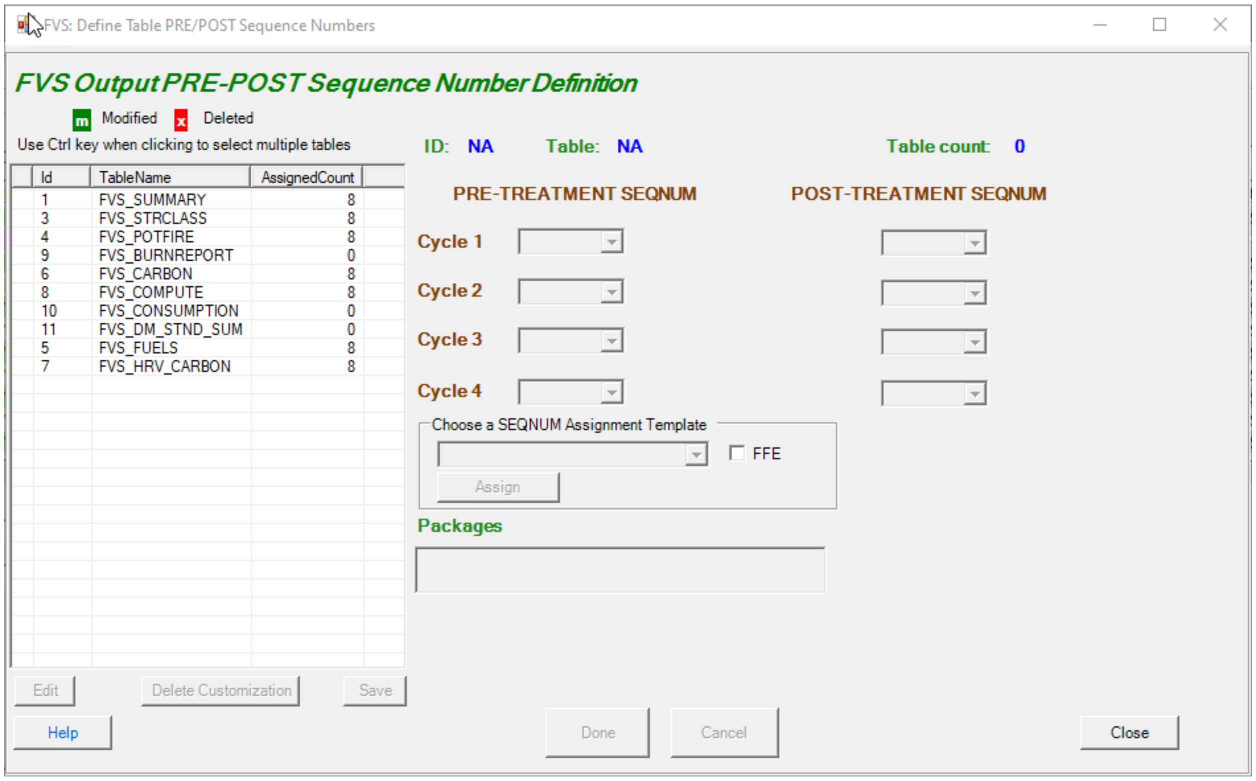
		FVS Variant	RxPackage	Cycle1Rx	Cycle2Rx	Cycle3Rx	Cycle4Rx
☐		PN	423	023	023	023	023
☐		PN	429	029	029	029	029
☐	a	PN	499	999	999	999	999
☐	a	PN	999	999	999	999	999
☐		WC	001	001	001	001	001
☐		WC	002	002	002	002	002
☐		WC	007	007	007	007	007
☐		WC	008	008	008	008	008
☐		WC	009	009	009	009	009
☐		WC	022	022	022	022	022
☐		WC	023	023	023	023	023

Tasks To Complete For Each Item: a=Append

All
Check All
Clear All
Refresh
PRE-APPEND Audit Tables
Open Pre Audit Log
POST-APPEND Audit Tables
Open Post Audit Log
Step 1 - Define PRE/POST Table SeqNum
Execute Step
Step 1 - Define PRE/POST Table SeqNum
Step 2 - Pre-Processing Audit Check
Step 3 - Append FVS Output Data
Step 4 - Post-Processing Audit Check
--Optional--
Create FVSOut_BioSum.db
Write FVS_InForest to FVSOUT_TREE_LIST.db
Help
Close

Figure 4.29 – Join and Append FVS Out Data window. Displays a list of all variant and package combinations.

The table grid displays names of all tables found in the FVSOut.db in the project_name\fvs\data folder and the number of sequence number assignments (AssignedCount) currently specified for each table. Those with zero will not be brought forward into PrePost databases.


FVS: Define Table PRE/POST Sequence Numbers

FVS Output PRE-POST Sequence Number Definition

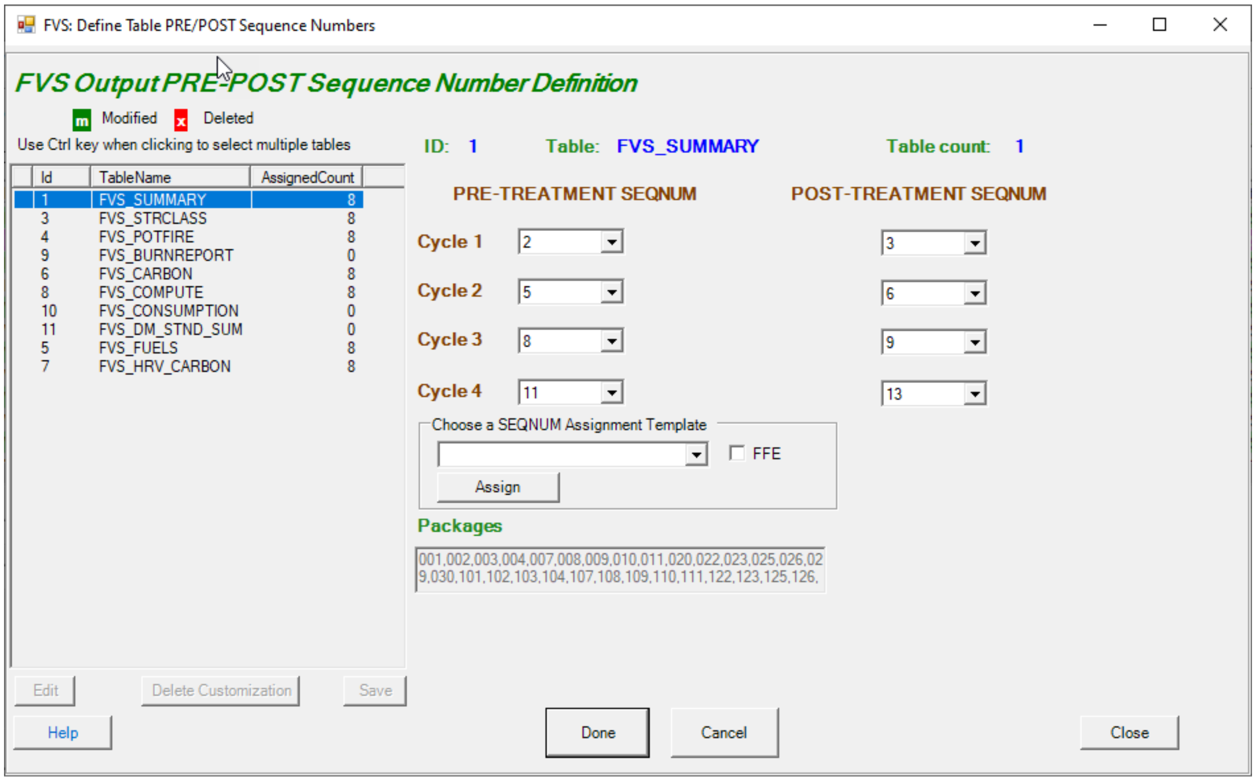
🟩 Modified 🟥 Deleted

Use Ctrl key when clicking to select multiple tables

ID: NA Table: NA Table count: 0

Id	TableName	AssignedCount
1	FVS_SUMMARY	8
3	FVS_STRCLASS	8
4	FVS_POTFIRE	8
9	FVS_BURNREPORT	0
6	FVS_CARBON	8
8	FVS_COMPUTE	8
10	FVS_CONSUMPTION	0
11	FVS_DM_STND_SUM	0
5	FVS_FUELS	8
7	FVS_HRV_CARBON	8

The fact that sequence numbers are assigned (i.e., a non-zero count) does not necessarily make them the right sequence numbers and at some point before proceeding to load data, every table should be checked. To view the current assignments, select a table name and press “edit”. The assignments populate and can be edited, if needed. In this assignment for Summary:


FVS: Define Table PRE/POST Sequence Numbers

FVS Output PRE-POST Sequence Number Definition

🟩 Modified 🟥 Deleted

Use Ctrl key when clicking to select multiple tables

Id	TableName	AssignedCount
1	FVS_SUMMARY	8
3	FVS_STRCLASS	8
4	FVS_POTFIRE	8
9	FVS_BURNREPORT	0
6	FVS_CARBON	8
8	FVS_COMPUTE	8
10	FVS_CONSUMPTION	0
11	FVS_DM_STND_SUM	0
5	FVS_FUELS	8
7	FVS_HRV_CARBON	8

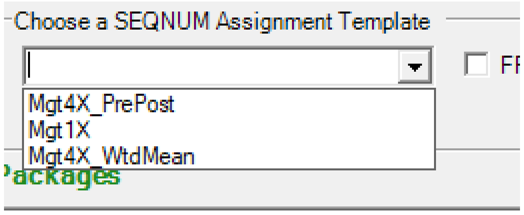
ID: 1 Table: FVS_SUMMARY Table count: 1

	PRE-TREATMENT SEQNUM	POST-TREATMENT SEQNUM
Cycle 1	2	3
Cycle 2	5	6
Cycle 3	8	9
Cycle 4	11	13

Choose a SEQNUM Assignment Template: ☐ FFE

Packages: 001,002,003,004,007,008,009,010,011,020,022,023,025,026,02 9,030,101,102,103,104,107,108,109,110,111,122,123,125,126,

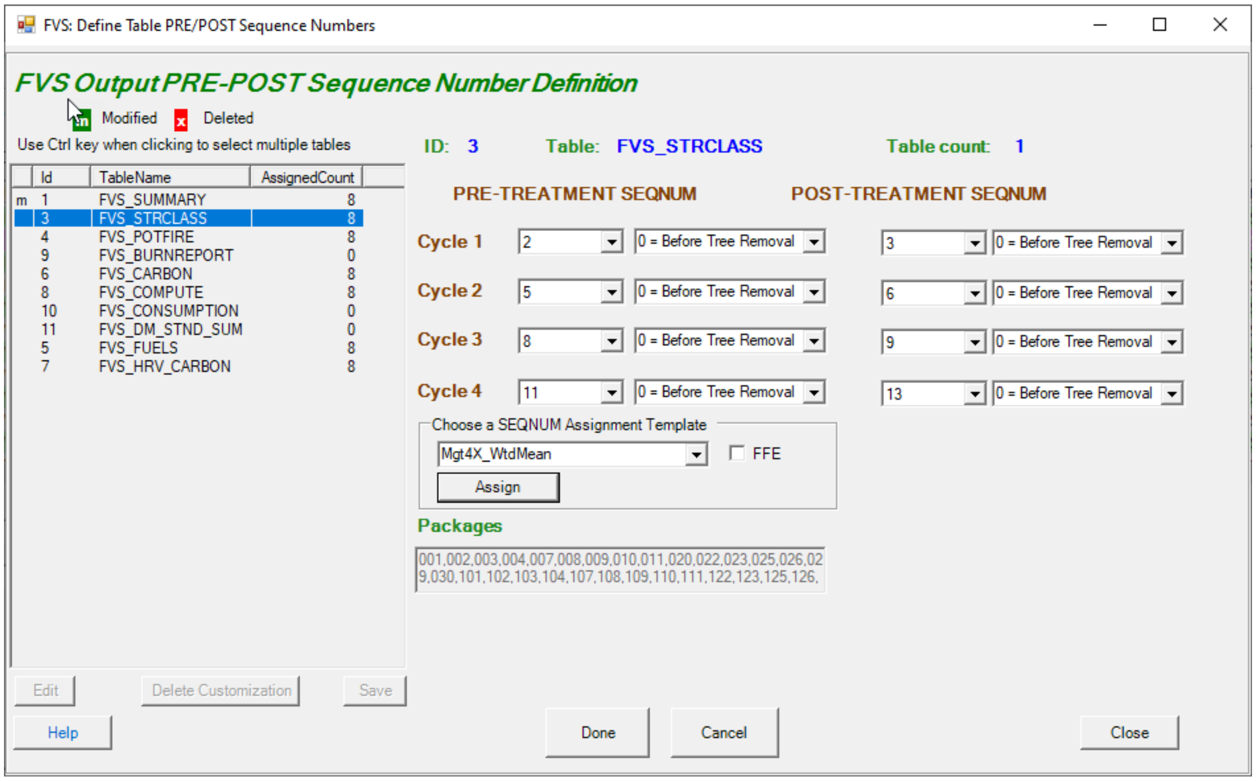
The cycle 1 Pre(treatment) value of 2 and the cycle 1 post of 3 tell us that for each stand, for each package, BioSum will take the 2nd output row in the summary table as the Pre value and the 3rd as the Post value, where table output rows are sorted by year within stand and package. This will be the case whether YEAR holds values like 2020, 2001, 2022, 2030, 2031, 2032, etc. as they would if the plot was sampled in 2020 and if InvYear 1 had not been specified in the FVS simulation or values like 1, 2, 3, 11, 12, 13, etc. if InvYear was set to 1 (the recommend option!). This assignment reflects the recommended practice of modeling a grow-only year before the first treatment can occur, via a 1 1 8 1 1 8 1 1 8 etc. pattern of projections which enables the specification of a true pre and post treatment year for each cycle for both FFE timed tables and non-FFE ones (Summary is an example of the latter).
These assignments can be made manually, but it is easier and more consistent to use the templates available in the template drop down.



The assignments displayed are patterned on the Mgt4X_WtdMean template (management at up to 4 cycles with a weighted mean style analysis which compares a treatment trajectory to a base trajectory such as grow-only). Selecting that template and choosing Assign will set the exact same assignments seen in this figure. The Max4_PrePost is a very similar template differing only in what is assigned for Cycle 4 Post—Sequence number 12, which pulls the record for immediately following treatment in Cycle 4 rather than a later value that allows for a weighted mean that extends further into the 4th decade. The Mgt1X template is designed for a BioSum simulation which treats only at cycle 1, then follows the outcome for 3 more cycles. The same assignments make sense in that case whether doing a pre-post or a weighted mean style analysis.

To end assignment for a table, click Done. If changes were made, an “m” for modified will appear on that table’s line in the table grid. The choices made will not take effect in the FVSOUT configuration until a Save is executed (with a button click). Closing without saving abandons the changes.

STRCLASS, another non-FFE table, has essentially the same assignments as SUMMARY except that one has the option of selecting the before or after tree removal cases (there are two lines in that output table for every simulated year). In almost all cases, you will want the before tree removal cases.



FVS: Define Table PRE/POST Sequence Numbers

FVS Output PRE-POST Sequence Number Definition

Use Ctrl key when clicking to select multiple tables

	Id	TableName	AssignedCount
m	1	FVS_SUMMARY	8
	3	FVS_STRCLASS	8
	4	FVS_POTFIRE	8
	9	FVS_BURNREPORT	0
	6	FVS_CARBON	8
	8	FVS_COMPUTE	8
	10	FVS_CONSUMPTION	0
	11	FVS_DM_STND_SUM	0
	5	FVS_FUELS	8
	7	FVS_HRV_CARBON	8

ID: 3 Table: FVS_STRCLASS Table count: 1

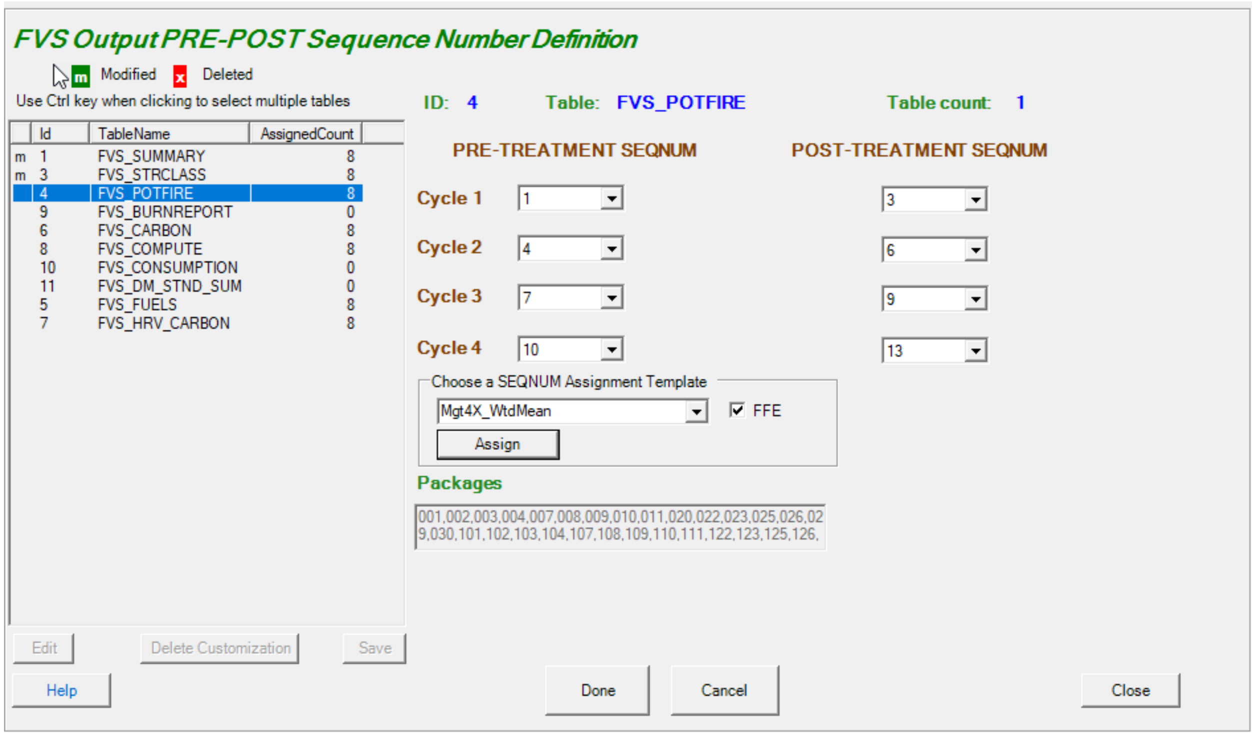
	PRE-TREATMENT SEQNUM		POST-TREATMENT SEQNUM	
Cycle 1	2	0 = Before Tree Removal	3	0 = Before Tree Removal
Cycle 2	5	0 = Before Tree Removal	6	0 = Before Tree Removal
Cycle 3	8	0 = Before Tree Removal	9	0 = Before Tree Removal
Cycle 4	11	0 = Before Tree Removal	13	0 = Before Tree Removal

Choose a SEQNUM Assignment Template: Mgt4X_WtdMean ☐ FFE

Packages: 001,002,003,004,007,008,009,010,011,020,022,023,025,026,029,030,101,102,103,104,107,108,109,110,111,122,123,125,126,

COMPUTE is a non-FFE table that must match the sequence numbers entered for the SUMMARY Table. If the sequence numbers between the two tables do not match, cost accounting for packages and treatments that include supplemental harvest costs (Additional CPA) will not work properly. Ensure that the FFE box is not checked when setting the COMPUTE table’s sequence numbers.

POTFIRE is an FFE table and has subtle differences in the assignments relative to the non-FFE tables owing to it displaying only post-harvest values in each record. FFE style tables need to have the FFE box checked in order for the Assign button to make the right assignments (e.g., for the CARBON and FUELS tables). BioSum knows that POTFIRE is an FFE table so will only assign it an FFE sequence (that relies on grow only years to have the pre-treatment values at each cycle) regardless of whether or not you check the FFE box for this table.



FVS Output PRE-POST Sequence Number Definition

Modified | Deleted

Use Ctrl key when clicking to select multiple tables

	Id	TableName	AssignedCount
m	1	FVS_SUMMARY	8
m	3	FVS_STRCLASS	8
	4	FVS_POTFIRE	8
	9	FVS_BURNREPORT	0
	6	FVS_CARBON	8
	8	FVS_COMPUTE	8
	10	FVS_CONSUMPTION	0
	11	FVS_DM_STND_SUM	0
	5	FVS_FUELS	8
	7	FVS_HRV_CARBON	8

ID: 4 Table: FVS_POTFIRE Table count: 1

	PRE-TREATMENT SEQNUM	POST-TREATMENT SEQNUM
Cycle 1	1	3
Cycle 2	4	6
Cycle 3	7	9
Cycle 4	10	13

Choose a SEQNUM Assignment Template: Mgt4X_WtdMean [✓ FFE]

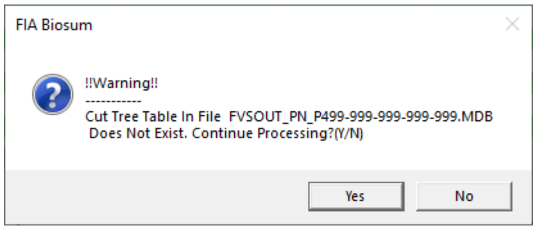
Packages: 001,002,003,004,007,008,009,010,011,020,022,023,025,026,029,030,101,102,103,104,107,108,109,110,111,122,123,125,126,

The Packages list is display only and not changeable. As of BioSum version 5.10.0, all packages must be assigned the same sequence number pattern—as we've concluded that any other more flexible arrangement only guarantees more confusion than value. If after assigning sequence numbers to a table, you decide not to load that table's data, simply select the table and choose "Delete Customization".
After all customizations are complete (or at any time in the process), click "Save" to save your work. Selecting "Close" will return you to the main Join and Append dialog.

Steps 2 and 4 in the Join and Append Dialog are audit steps that assess data readiness and perform QA chores on the package-variants with checked boxes.

Select Step 2 <Pre-processing Audit Check> and "Execute Step" to run an audit on the output databases. During this audit each table in the FVS output is analyzed and PRE and POST year is assigned to each cycle. The analysis is written to several tables with the table name format of audit_pre_post_rx_year_fvsoutputtablename. Additionally, the data from the FVS_summary table is analyzed to determine if the year in the table is represented in the other tables.

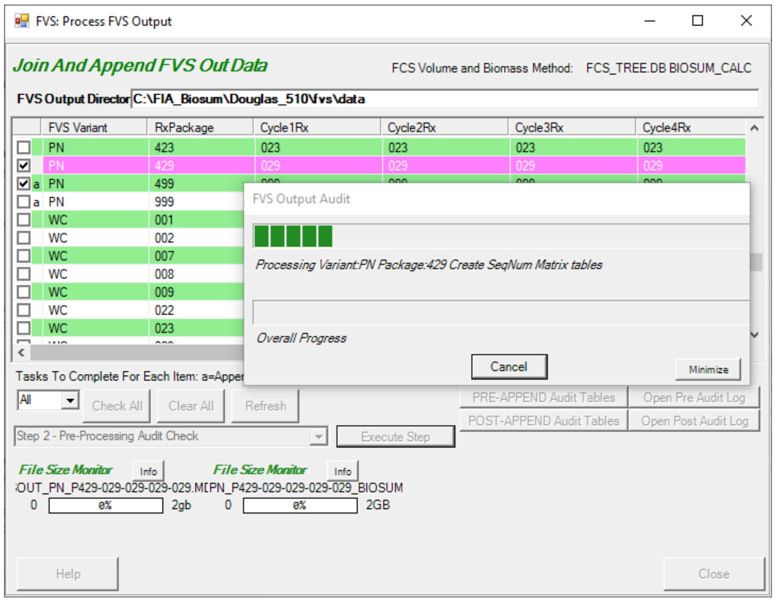
If BioSum detects an error or inconsistency in the data, a pop-up window will appear. If there are errors not significant enough to affect processing, the audit will pass with a warning. It is up to the user to determine if warnings require action. More significant errors will return a failed audit result. The user must address the specified error before continuing. Packages that don't contain mechanical thinning (such as a grow-only simulation) and therefore do not contain a CUTLIST, will receive a message asking if you want to continue without a cutlist. Select <Yes> to continue. Packages without a CUTLIST will technically fail the audit, but if other issues aren't present in the audit results log, the audit is functionally successful.


FIA Biosum

!!Warning!!

Cut Tree Table In File FVSOUT_PN_P499-999-999-999-999.MDB
 Does Not Exist. Continue Processing?(Y/N)

The audit operation builds and analyzes temporary databases that live within your windows\user\[username]\appdata\temp folder. The sizes of these databases as the audit proceeds are monitored with progress bars as a diagnostic in case a glitch occurs owing to exceed the MS Access maximum database size allowed (2 GB).


FVS: Process FVS Output

Join And Append FVS Out Data

FCS Volume and Biomass Method: FCS_TREE.DB BIOSUM_CALC

FVS Output Director C:\FIA_Biosum\Douglas_510\fvs\data

		FVS Variant	RxPackage	Cycle1Rx	Cycle2Rx	Cycle3Rx	Cycle4Rx
☐		PN	423	023	023	023	023
☑		PN	429	029	029	029	029
☑	a	PN	499				
☐	a	PN	999				
☐		WC	001				
☐		WC	002				
☐		WC	007				
☐		WC	008				
☐		WC	009				
☐		WC	022				
☐		WC	023				

FVS Output Audit

Processing Variant:PN Package:429 Create SeqNum Matrix tables

Overall Progress

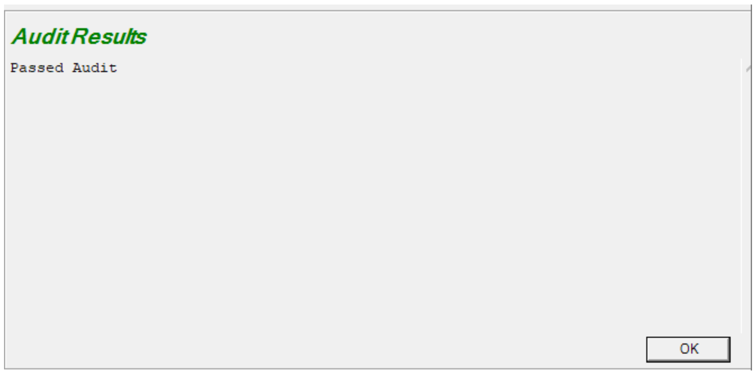
Tasks To Complete For Each Item: a=Apper

Step 2 - Pre-Processing Audit Check

File Size Monitor
OUT_PN_P429-029-029-029-029.M
0 0% 2gb

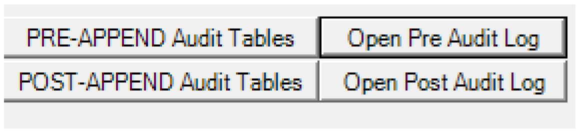
File Size Monitor
PN_P429-029-029-029-029_BIOSUM
0 0% 2GB

When the audit concludes, a dialog appears returning an audit status.

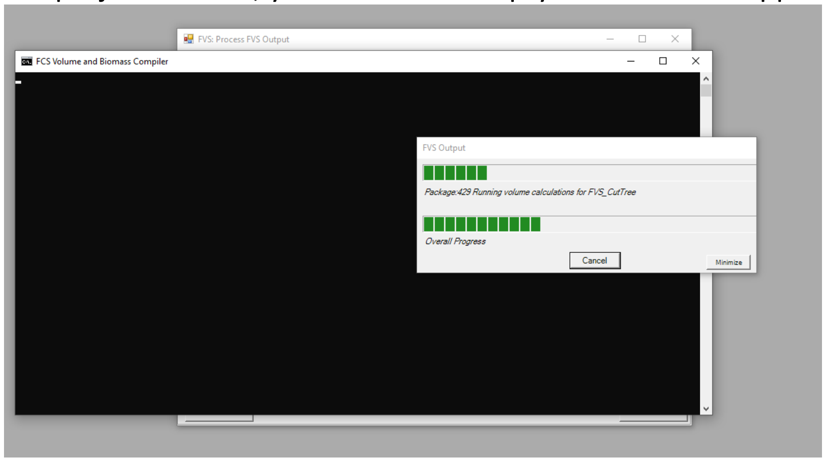

Audit Results

```
Passed Audit
```

Detailed information about any issues that arose can be viewed by summoning the pre-append audit log and/or pre-append audit tables



Step 3 loads the FVS output data into BioSum PrePost databases (in the project's fvs\db folder) and the cut tree list in the FVSOUT_TREE_LIST.DB (in the project's fvs\data folder). Execute Step 3 (<Append FVS Output Data>). Pre and post record values for all FVS output tables (except the atrList, treeList, and cutList tables) are appended to tables in the project's fvs\db folder. For example, pretreatment SUMMARY records are written to PRE_FVS_SUMMARY table and post treatment records to POST_FVS_SUMMARY table based on the sequence number assignments selected in the previous steps. The BioSum Optimizer module then uses these tables to define treatment effectiveness.
Biomass and volume of the trees specified as harvested in the FVS CutList table are calculated using regional and species specific formulas. This process runs within Java. BioSum updates the fvs\data\FVSOUT_TREE_LIST.DB database with the calculated volumes and biomass.   Calculated volumes and biomass are written, for example, to the VOLCFNET, VOLCFGRS, VOLCSGRS, DRYBIOM, and DRYBIOT columns in the CUTLIST table. These values are used to calculate harvest value and costs in BioSum Processor.
A warning dialog for packages with no harvested trees can be cleared—this is typically displayed when a package is grow-only (so no cut trees will be written to FVSOUT_TREE_LIST.DB) but such packages nonetheless must be loaded into BioSum (to populate the PrePost databases). When the loader calculates volume and biomass of the projected trees, you will see an empty black window appear for a short time.



Package:429 Running volume calculations for FVS_CutTree

Overall Progress

Finally, once Step 3 is complete, run step 4 (<Post-Processing Audit Check>) to ensure all necessary data are present and formatted correctly. If errors exist, evaluate and rectify before moving on to the Processor module.

It is essential to complete the post-load audit to ensure data integrity, step 4. Audit results will post to the Audit Results dialog and be written to the log file that can be accessed via the "Open Post Audit Log" button. It is common for there to be warnings about species changes, almost always due to the mapping of species that must be specified in the Tree Species table to address the fact that some trees in an FVS variant will not be directly supported as the species that they are.

After successful loading and a clean bill of health from pre- and post-append audits for every variant package combination, one can proceed to the Processor module. There are two optional tasks that can be completed here at any time—they source data from the FVSOut.DB and do not depend on a successful append operation. They also act on all package combinations represented in the FVSOut database (ignoring the checkboxes).

The ”Create FVSOut_BioSum.DB” task writes an SQLite database, sourced from FVSOut.DB, that formats all data as data types that are fully compatible with MS Access. This can be very helpful if needing to work with FVSOut data from MS Access, as the native data types in that database (such as LONG TEXT, which translates to Memo in Access) are nearly impossible to work with (e.g., sort, filter, etc.). As this is not a small file, it only makes sense to create it if you believe you will use it. Moreover, it can be created at any time, as it is not used in any BioSum module.

The “Write FVS_InForest to FVSOUT_TREE_LIST.DB” task copies all data from the TREELIST in FVSOut.DB into the FVSOUT_TREE_LIST.DB (in the project’s fvs\data folder). This can be useful if needing tree level biomass or carbon data, for example, at all time points in the simulation for analysis beyond what the BioSum framework manages directly. As this is not a small file, it only makes sense to create it if you believe you will use it. Moreover, it can be created at any time, as it is not used in any BioSum module.

Additional columns in the data grid (which can be scrolled to see) provide status information that is sometimes useful for diagnosing issues.

Advanced Topic: Creating FVS Input Databases using BioSum

Aside from the many potentially fruitful applications of a full BioSum analysis, this software can also be useful when there is a need to create a variant-specific FVS input database from data formatted consistent with FIADB but not downloaded from the Datamart. This generates an Access .ACCDB file (not an SQLite file) that can be read by either FVS Online or the deprecated FVS Suppose. This use case is outside of the BioSum scope documented in this User Guide, but workflow that has successfully accomplished this task for FIADB formatted spatially intensified off-grid plots that are not part of the FIA stratification, entailed the following steps:

1. Start with a DATAMART originated FIADB SQLite database, copying the table structures for PLOT, COND, TREE, SITETREE, and SEEDLING into a new SQLite database. This can be accomplished by copying just the data structures or copying the tables, then deleting all records in these tables to create a template.
2. Copy the table structures for the four POP_ tables (EVAL, ESTN_UNIT, STRATUM and PLOT_STRATUM_ASSGN from an existing BioSum project into your SQLite template).
3. Populate the PLOT, COND, TREE, SITETREE, and SEEDLING tables in the template via queries from your custom FIADB style data source.
4. Populate the POP tables to enable the data to load into BioSum so that an FVSIn.accdb can be created. The simplest POP table quartet that will enable a BioSum load won’t necessarily generate valid area expansions for each plot, but if that is not important to

the application for which the FVS input file is required, then that is not a problem. You'll need a single record in each of three of the tables (ESTN_UNIT, EVAL and STRATUM) – that is to say one estimation unit, one evaluation, and one stratum, and as many records in PLOT_STRATUM_ASSGN as exist in the PLOT table (referencing the one record in each of the other 3 POP tables).

5. When creating the FVSIn.accdb (which will appear in the variant folder, along with an empty FVSIn.DB), choose for the input database that contains FVS_STAND_INIT and FVS_TREE_INIT any FIADB database that does NOT contain records for your plots in those tables (e.g., could even be from another state). BioSum will then use its old workflow to populate the FVSIn.accdb without relying on FIADB INIT tables.

If contemplating this use case and in need of the most current guidance on this use of BioSum, contact BioSum support.

Literature Cited

Crookston, Nicholas L.; Gammel, Dennis L.; Rebain, Stephanie; Robinson, Donald; Keyser, Chad E.; Dahl, Christopher A.; David, Lance R.; Shettles, Michael A. 2003 (revised November 2022). Users Guide to the Database Extension of the Forest Vegetation Simulator Version 2.0. Internal Rep. Fort Collins, CO: U. S. Department of Agriculture, Forest Service, Forest Management Service Center. 60p

Jain, T.B., J.S. Fried and S. Loreno. 2020. Simulating the effectiveness of improvement cuts and commercial thinning to enhance fire resistance in west coast dry mixed conifer forests. Forest Science 66(2):157-177.

Rebain, Stephanie A. comp. 2010 (revised February 1, 2022). The Fire and Fuels Extension to the Forest Vegetation Simulator: Updated Model Documentation. Internal Rep. Fort Collins, CO: U. S. Department of Agriculture, Forest Service, Forest Management Service Center. 409p.

Dixon, Gary E. comp. 2002. Essential FVS: A user's guide to the Forest Vegetation Simulator. Internal Rep. Fort Collins, CO: U. S. Department of Agriculture, Forest Service, Forest Management Service Center. 226p. (Revised: August 24, 2022)

Van Dyck, Michael G; Smith-Mateja, Erin E. (Editors). 2000. Keyword Reference Guide for the Forest Vegetation Simulator. Fort Collins, CO: U.S. Department of Agriculture, Forest Service, Forest Management Service Center. 118p. (Revised: December, 2022)

Vandendriesche, Donald A. 2009. An Empirical Approach for Estimating Natural Regeneration for the Forest Vegetation Simulator. U.S. Department of Agriculture, Forest Service, Rocky Mountain Research Station. Ogden, UT. Proceedings: National Silviculture Workshop; June 15-19, 2009. Boise, ID.

Vandendriesche, Donald. 2010. FVS out of the box-assembly required. In Jain, Theresa B.; Graham, Russell T.; and Sandquist, Jonathan, tech. eds. 2010. *Integrated management of carbon sequestration and biomass utilization opportunities in a changing climate:*

Proceedings of the 2009 National Silviculture Workshop; 2009 June 15-18; Boise, ID. Proceedings RMRS-P-61. Fort Collins, CO: U.S. Department of Agriculture, Forest Service, Rocky Mountain Research Station. 351 p.
http://www.fs.fed.us/rm/pubs/rmrs_p061/rmrs_p061_289_306.pdf

Chapter 5. Processor

Overview

BioSum's Processor module handles two major tasks: 1) preparing data summaries for input to OpCost (a forest operations cost simulator developed specifically for use with BioSum) and 2) summing, for each modeled silvicultural sequence, the volume, biomass, and value of wood collected from each stand at each simulation cycle where harvest occurs.

By processing FVS generated "cut lists" associated with harvest events occurring as part of a silvicultural sequence, Processor calculates a suite of cost-relevant metrics from the tree records contained in the cut list tables (generated during the FVS simulation and subsequent FVS Output Load operation in BioSum and stored in the project/fvs/data/variant_name/BioSumCalc directories) and attributes of the plot and condition (stored in project/db/master) to which the stand is linked. Cut list metrics include, for each of three user-defined harvested tree size classes (chip, small log and large log), trees per acre (tpa), tpa-weighted average volume (cubic feet) per tree, tpa and volume weighted average wood density (lbs/ft^3) per tree, percent of volume in hardwood species, percent of harvest volume in that size class that will be chipped, and, for small and large log trees, quadratic mean diameter.

Processor also calculates trees per acre and average volume per tree for brush-cut size (sub chip tree-sized) trees in the cut list and the fraction of the live tree basal area that is harvested under the prescription. For each combination of stand, silvicultural sequence (package) and FVS cycle, these summary metrics are passed as input to OpCost along, with the assumed harvest system and operation size (area in acres) and stand-level parameters such as slope, yarding distance and time required for harvest equipment to "move-in" and be ready for forest operations.

Based on these inputs, OpCost predicts harvest costs per acre, with chipping and move-in costs broken out separately, which is useful for some analyses. All inputs to and outputs from OpCost are accessible via Access databases within the OpCost directory of a BioSum project. Processor transfers appropriate results to a harvest_costs table in the processor scenario database (in the project/processor/scenario_name directory).

Processor also calculates per acre volume, weight, and value of merchantable and energy wood removed, by FVS cycle, as a result of implementing a silvicultural sequence. These are binned by user-defined diameter classes and species groups and stored in the processor scenario database's tree_vol_val_by_species_diam_groups table.

Module Components

The button that activates the Processor module can be found in the module-selector on the left side of the FIA BioSum Manager desktop (it's labeled "Processor"). Selecting it opens a 2-button panel that accesses these tasks: (1) **Tree Species—audit the tree species table, and (2) Start BioSum Processor's interactive parameter specification workflow.** The first 1 must complete successfully before initiating the BioSum Processor.

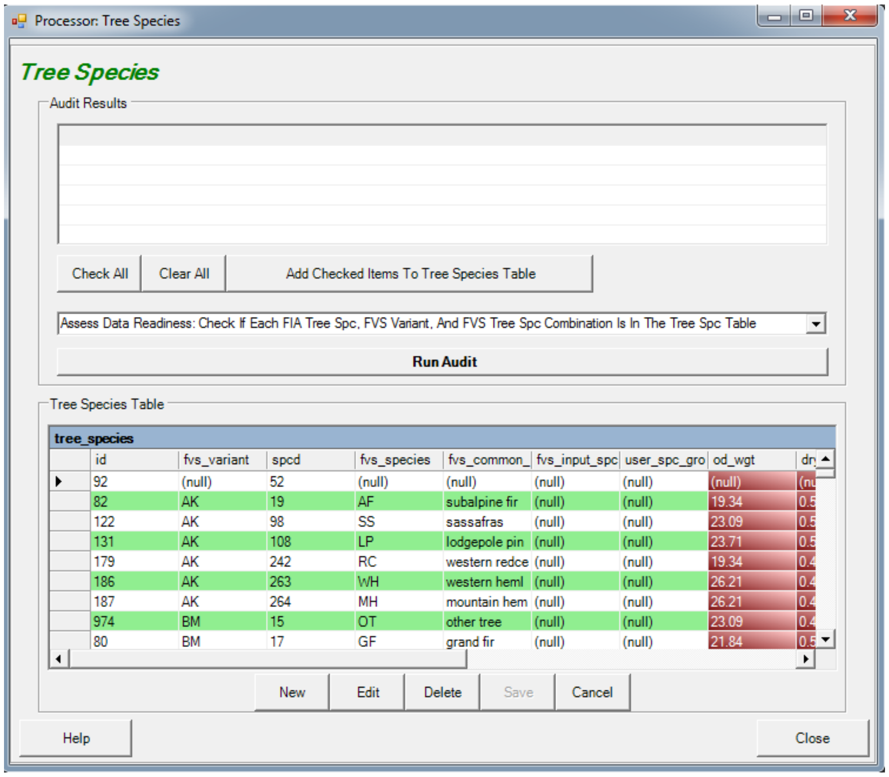
Tree Species

This button initiates a dialog for conducting audits of cut list species attributes and for managing the tree_species table stored in project\db\ref_master. As with the Tree Species task in the FVS module's work flow, the purpose of this table is to provide a systematic method for assigning FVS species codes for every FIA species code that occurs in the cut list tree data generated from FVS. These assignments can vary by FVS variant, so the table contains multiple records for most species.

Audits should be run the first time FVS cut list output is used, to assess data readiness for subsequent processes. The audit will identify any combinations of FIA tree species, FVS variant, and FVS tree species found in the FVS output that are absent from the *tree_species* table. This could happen, for example, due to regeneration provisions coded into the FVS KCP files if tree regeneration is modeled for species that do not occur in the tree_species table for that variant. To assure successful completion of the rest of the Processor module, users are advised to ensure a clean audit.

1. Click the **<Tree Species**> button in the Processor module selector panel to open the **Tree Species** window. The data displayed in the *tree_species* table in the bottom half of this window derives from a work table, generated by filtering the /db/ref_master_tree_species table based on the FIA species codes present in the /fvs/data/FVSOUT_TREE_LIST.db database file. Note that ALL records for species with matches in the FVS_CutTree table will enter this work table, regardless of the variant associated with the record in the cut list.

This table shows every combination of variant and FIA species code (spcd) and the FVS species to which it is "mapped" (fvs_species and fvs_input_spcd), and the scientific nomenclature (genus and species). Processor uses the FIA species code to assign oven dry weight (wood density) and dry to green ratio parameters from a reference table, as both are needed to estimate green weight of merchantable and energy wood for calculating costs of delivering wood from the forest to facilities.


Processor: Tree Species

Tree Species

Audit Results

Assess Data Readiness: Check If Each FIA Tree Spc, FVS Variant, And FVS Tree Spc Combination Is In The Tree Spc Table

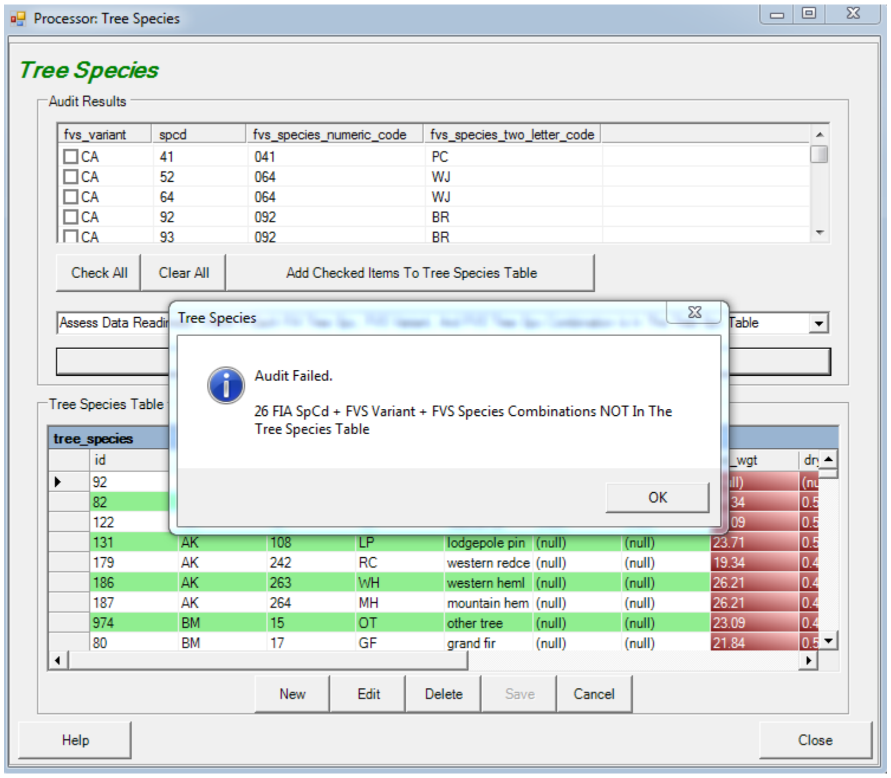
Tree Species Table

tree_species

id	fvs_variant	spcd	fvs_species	fvs_common_	fvs_input_spc	user_spc_gro	od_wgt	dr
92	(null)	52	(null)	(null)	(null)	(null)	(null)	(nu
82	AK	19	AF	subalpine fir	(null)	(null)	19.34	0.5
122	AK	98	SS	sassafras	(null)	(null)	23.09	0.5
131	AK	108	LP	lodgepole pin	(null)	(null)	23.71	0.5
179	AK	242	RC	western redce	(null)	(null)	19.34	0.4
186	AK	263	WH	western heml	(null)	(null)	26.21	0.4
187	AK	264	MH	mountain hem	(null)	(null)	26.21	0.4
974	BM	15	OT	other tree	(null)	(null)	23.09	0.4
80	BM	17	GF	grand fir	(null)	(null)	21.84	0.5

Figure 5.1 –Tree Species window, showing a work table derived from the *tree_species table in the lower half.*

2. In the drop down list box above the tree_species table, make sure the audit displayed is **"Check If Each FIA Tree Spc, FVS Variant, And FVS Tree Spc Combination Is In The Tree Spc Table"**), then click **<Run Audit>**. Within the FVS module completed earlier in the BioSum workflow, this audit was performed to check if each FIA tree species and FVS variant combination in project_name/db/master_tree existed in the *tree_species* table. Within Processor, this audit checks, using FVS_CutTree tables as the source, rather than the master.TREE table, to be sure that every combination of FIA tree species (as determined by the tree record in MASTER.TREE) and FVS tree species (as specified in the FVS_CutTree output) for a tree has a tree_species record for the FVS variant where that tree exists. It also checks new trees "created" during the FVS simulation based on keywords used to represent regeneration. The audit will display a message dialog reporting the number of variant/tree species combinations that are not represented in the table based on a review of FVS_CutTree trees.



Processor: Tree Species

Tree Species

Audit Results

fvs_variant	spcd	fvs_species_numeric_code	fvs_species_two_letter_code
☐ CA	41	041	PC
☐ CA	52	064	WJ
☐ CA	64	064	WJ
☐ CA	92	092	BR
☐ CA	93	092	BR

Tree Species

Audit Failed.

26 FIA SpCd + FVS Variant + FVS Species Combinations NOT In The Tree Species Table

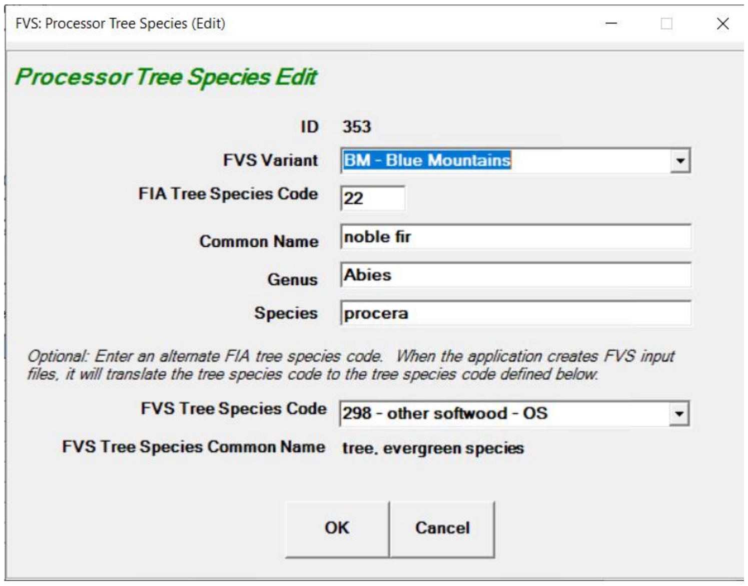
Tree Species Table

tree_species

id							_wgt	dr
92							ll)	(nu
82							34	0.5
122							09	0.5
131	AK	108	LP	lodgepole pin	(null)	(null)	23.71	0.5
179	AK	242	RC	western redce	(null)	(null)	19.34	0.4
186	AK	263	WH	western heml	(null)	(null)	26.21	0.4
187	AK	264	MH	mountain hem	(null)	(null)	26.21	0.4
974	BM	15	OT	other tree	(null)	(null)	23.09	0.4
80	BM	17	GF	grand fir	(null)	(null)	21.84	0.5

Figure 5.2 – Audit Failed message. Records in the Audit Results window need to be added to the tree_species table.

3. If this audit fails, results will be displayed for cases where combinations are missing from the *tree_species* table. These records can be selected (via checkboxes) for addition to the *tree_species* table individually, or en masse (by clicking **<Check All>**), followed by **<Add Checked Items To Tree Species Table>** button. Be sure to **<Save>** changes after adding records. These additions to the work table will automatically propagate through to also update *ref_master.tree_species*.
   Important Note: Due to a glitch in the audit, the audit may report a failure even if all species are in the Tree Species table. If there are no checkboxes in upper grid to check, this error message may be ignored.



FVS: Processor Tree Species (Edit)

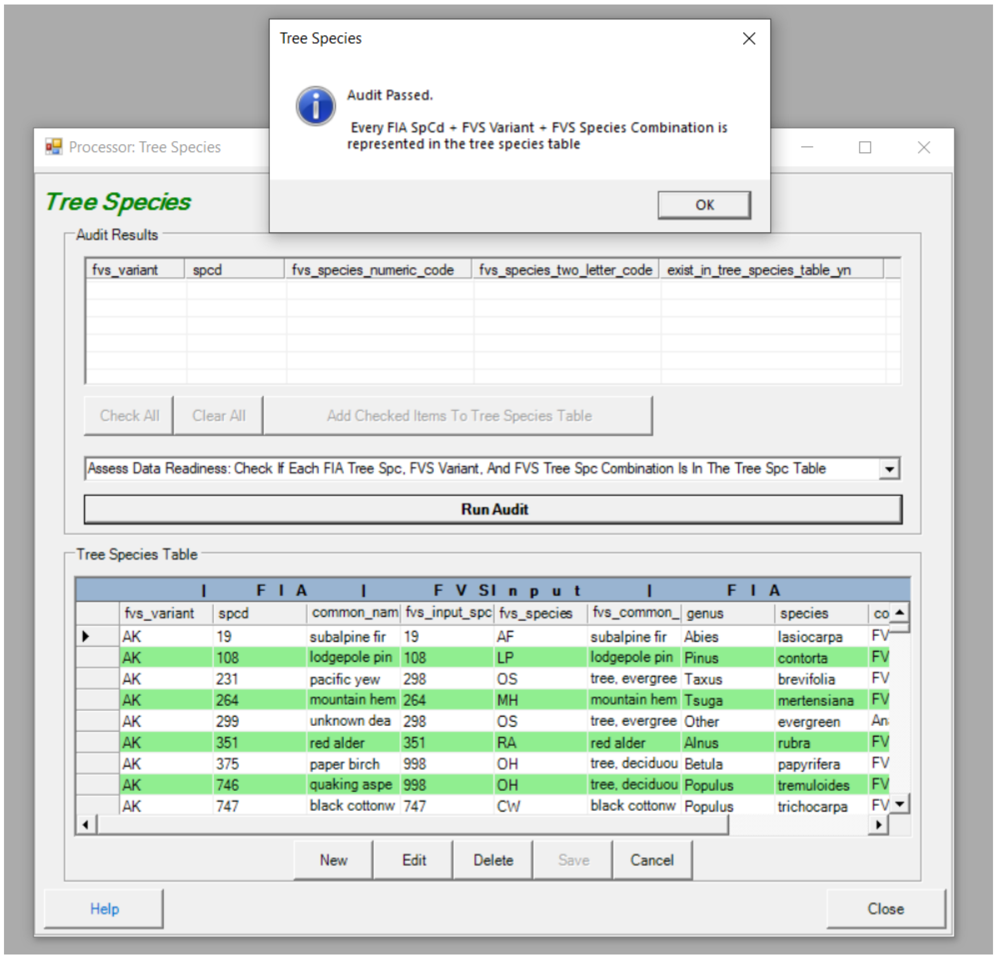
Processor Tree Species Edit

ID	353
FVS Variant	BM - Blue Mountains
FIA Tree Species Code	22
Common Name	noble fir
Genus	Abies
Species	procera

Optional: Enter an alternate FIA tree species code. When the application creates FVS input files, it will translate the tree species code to the tree species code defined below.

FVS Tree Species Code	298 - other softwood - OS
FVS Tree Species Common Name	tree, evergreen species

Figure 5.3 – Tree species edit window for viewing cross-walks between FIA and FVS tree species codes; these can also be edited in the tree species table part of the Tree Species dialogue.



Tree Species

Audit Passed.

Every FIA SpCd + FVS Variant + FVS Species Combination is represented in the tree species table

Processor: Tree Species

Tree Species

Audit Results

fvs_variant	spcd	fvs_species_numeric_code	fvs_species_two_letter_code	exist_in_tree_species_table_yn

Assess Data Readiness: Check If Each FIA Tree Spc, FVS Variant, And FVS Tree Spc Combination Is In The Tree Spc Table

Tree Species Table

| FIA			| FVS Input			| FIA			
fvs_variant	spcd	common_nam	fvs_input_spc	fvs_species	fvs_common_	genus	species	co
AK	19	subalpine fir	19	AF	subalpine fir	Abies	lasiocarpa	FV
AK	108	lodgepole pin	108	LP	lodgepole pin	Pinus	contorta	FV
AK	231	pacific yew	298	OS	tree, evergree	Taxus	brevifolia	FV
AK	264	mountain hem	264	MH	mountain hem	Tsuga	mertensiana	FV
AK	299	unknown dea	298	OS	tree, evergree	Other	evergreen	An
AK	351	red alder	351	RA	red alder	Alnus	rubra	FV
AK	375	paper birch	998	OH	tree, deciduou	Betula	papyrifera	FV
AK	746	quaking aspe	998	OH	tree, deciduou	Populus	tremuloides	FV
AK	747	black cottonw	747	CW	black cottonw	Populus	trichocarpa	FV

Figure 5.4 – the "Audit Passed" dialog box signifies a successful tree species audit.

4. After the audit has passed without error, **<Save>** changes and **<Close>** the Tree Species window.

OpCost

Introduction: OpCost is an R script that estimates treatment costs (dollars per acre) for each combination of stand, silvicutural sequence and treatment year as a function of these attributes, calculated in BioSum from project data that draws on plot, condition and tree attributes as well as user input:

- Percent slope
- Yarding distance
- FVS simulation year (used when applying discounting)
- Stand elevation
- Harvest system
- Chip trees per acre
- Chip trees Merchantable as a percent of Total Volume
- Chip trees average volume per tree (ft3)
- Chip trees average wood density (lbs./ft3)
- Chip trees hardwood percent
- Small log trees per acre
- Small log trees Merchantable as a percent of Total Volume
- Small log trees Chip Percent with Category 1 and 3 harvest systems
- Small log trees Chip Percent with Category 2 and 4 harvest systems
- Small log trees Chip Percent with Category 5 harvest systems
- Small log trees average volume per tree (ft3)
- Small log trees average wood density (lbs./ft3)
- Small log trees hardwood percent
- Large log trees per acre
- Large log trees Merchantable as a percent of Total Volume
- Large log trees Chip Percent with Category 1, 3 and 4 harvest systems
- Large log trees Chip Percent with Category 2 harvest systems
- Large log trees Chip Percent with Category 5 harvest systems
- Large log trees average volume per tree (ft3)
- Large log trees average wood density (lbs./ft3)
- Large log trees hardwood percent
- BrushCut (cut but not taken to landing) TPA
- BrushCut average volume per tree (ft3)
- Move-in hours (to bring each piece of logging equipment to the work site)
- Harvest area (acres) on which the silvicultural sequence is implemented
- Basal area fraction (of all live tree basal area) that is harvested
- Quadratic mean diameter of small log trees (inches)

- Quadratic mean diameter of large log trees (inches)

OpCost relies on published harvest cost models developed from empirical studies of harvest systems conducted in a variety of geographic regions with different topographies and silvicultural systems. The data from these studies, some of which date back decades, include information about the kinds and number of machines typically used for a given system, and the models developed from these data typically predict, for each machine used in a logging system, the productive machine hours (PMH) required to log, forward or yard, load and, if applicable, chip the harvested wood. Machine time requirements estimated from these models are multiplied by typical operating costs per PMH to arrive at treatment cost, and move-in times are multiplied by the hourly rental rate per machine and divided by the assumed area treated to calculate move-in costs per acre for getting all of the equipment associated with a harvest system to a treatment site and ready to begin production. The sum of treatment and move-in cost and an optionally specified set of supplemental per acre costs (for example, to account for pile and burning of activity and surface fuels) yields overall harvest cost per acre for a stand-sequence-year combination.

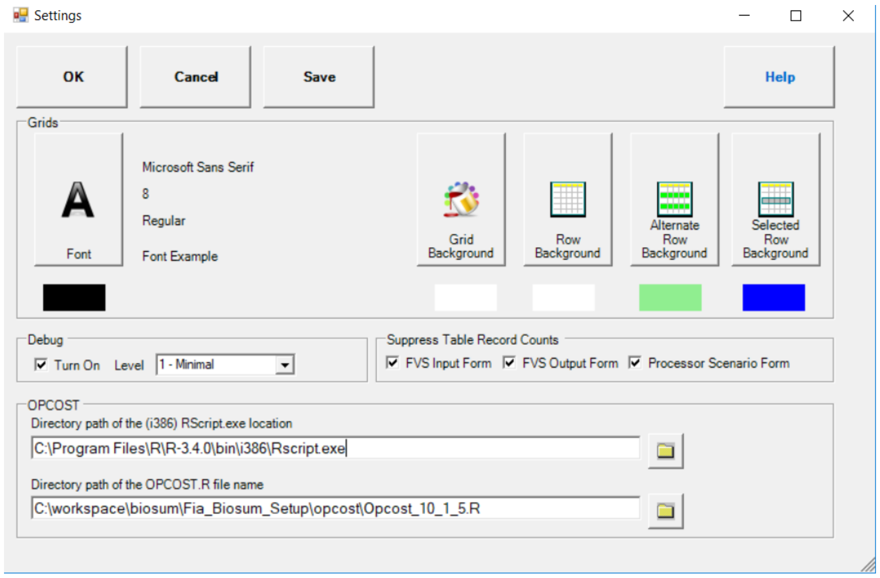
As of December 2018, OpCost encodes 24 models (equations), each of which uses some of the above information to predict the logging machine time required for a particular piece of machinery such as a mechanized harvester, tethered forwarder, stroke boom delimber or chainsaw that is used in one or more logging systems. OpCost and BioSum currently recognize and can model machine time requirements for 11 different harvest systems, each of which consist of multiple machines working in a coordinated fashion to move wood from standing tree to logs or chips on a truck. For some machines, multiple equations are included, and in these cases, an average of the predicted time required is used in calculating the machine cost. Many of the equations have limiting conditions that govern the range of input values for which valid predictions can be obtained. Some of these limits, such as a maximum average tree diameter or maximum slope or yarding distance, relate to the range of data from which the model was estimated; in other cases, the limit is necessary to avoid predicting, for example, a negative machine time. Owing to these limits, equations sometimes drop out of the running for some machines. For a few stands, all available equations may be ruled out for a given machine type—most commonly this happens with yarders on stands with extreme yarding distance (of over a mile). Because costs cannot be estimated for implementing management on these stands, these stands are dropped from a BioSum analysis. Any stands dropped for this reason will appear in the OpCost_errors table in the OpCost_input database at the end of a Processor operation. The OpCost_errors table may be useful for debugging, or for choosing a different harvest system to model for stands that end up in this table. These stands will not appear in the Opcost_output table or the harvest_cost table in the processor scenario and will not be processed further in a BioSum analysis unless rerun successfully with different inputs or a different harvest system. It is very important to check the OpCost_errors table to understand why some stands are dropped from the simulation and to satisfy yourself that this is an appropriate outcome, for example, because of excessive slope or yarding distance.

The equations included in OpCost are best thought of as a “starter set”. They may or may not be the best equations for a particular treatment in a particular place or forest type. For example, some were developed from small diameter conifer stands while others were developed for large diameter hardwoods. These equations are a vetted subset of those selected by the research collaborator at the University of Idaho responsible for the first version of OpCost (Bell et al. 2017), that replaced the Fuel Reduction Cost Simulator (FRCS, Fight et al. 2006) used in earlier versions of BioSum. As an Excel macro based tool, FRCS was challenging to document and modify as well as to integrate with the rest of BioSum. Some of the equations used in FRCS may not accurately reflect current technologies and harvest systems, and the OpCost effort was intended to pull in new equations and harvest systems, where possible. However, relatively few of the newer equations are based on studies conducted in the western U.S. Some users may wish to review the equations and the literature on which they are based. Thanks to OpCost’s modular design which relegates nearly all equations, units, limiting criteria and parameters to tables in an Access database, opcost_ref, the equations and parameters can be easily inspected, supplemented, adjusted and replaced as needed.

OpCost Settings

OpCost settings are accessible from the **<Settings>** menu at the top of any BioSum screen. The OpCost settings are at the bottom of the settings dialog and include controls for entering or confirming the full path (including directories) to both the binary file that implements the Rscript interpreter and to the OpCost R script (a plain text .R file).

1. Open the OPCOST settings dialog by clicking the **<Settings>** menu at the top of the BioSum window.



Microsoft Sans Serif
8
Regular
Font Example

Debug: ☑ Turn On Level 1 - Minimal

Suppress Table Record Counts: ☑ FVS Input Form ☑ FVS Output Form ☑ Processor Scenario Form

Directory path of the (i386) RScript.exe location
C:\Program Files\R\R-3.4.0\bin\i386\Rscript.exe

Directory path of the OPCOST.R file name
C:\workspace\biosum\Fia_Biosum_Setup\opcost\Opcost_10_1_5.R

Figure 5.5 – BioSum Settings dialog.

2. The first entry field will typically already be populated with the path to the Rscript executable file in the R folder on the C:\ drive, but it is worth checking to ensure that the location is correct. If using the 64-bit version of BioSum, the x64/Rscript.exe should be selected. Or when using the 32-bit version of BioSum, the i386/Rscript.exe should be selected. OpCost was written and tested in version 4.4.0 of R, and is not guaranteed to work in newer versions.
3. The second field should point to the OpCost R script. Select the folder icon next to **Directory path of the OPCOST.R file name** and navigate to the R script found here: C:\Program Files (x86)\FIA PNW Portland Forestry Sciences Lab\FIA Biosum *version*\OpCost. Select the OpCost_*version*.R file you wish to use, and click **<Open>**.
4. Click **<Save>** to save the settings specified in the dialogue. If **<OK>** is selected, the settings will be used during the current BioSum session and then saved when BioSum is closed.

Start BioSum Processor

This task consists of several steps through which Processor scenarios are defined and executed to produce intermediate outputs that serve as inputs to the **Core Analysis** module. Calculations depend on links among tables in project_name/db/master.mdb, project_name/db/ref_master.mdb, project_name/fvs/data/ /BiosumCalc/variant_code_Package_ID_TREE_CUTLIST.mdb, and a number of scenario-specific tables within project_name/processor/db/scenario_processor_rule_definitions.mdb. Tables that are used as inputs include:

- The *plot* and *cond* tables in master contain information on the stands and conditions loaded in the current project.
- The *variant_tree_cutlist* tables contain biomass and volume estimates for harvested trees calculated using the FIADB Oracle Schema.
- The *tree_species* table is a reference table used to calculate the weight of harvested trees and convert FIA species codes to those used in FVS.
- The *tree_species_groups* and *diameter_groups* tables contain user-specified species and diameter group parameters.

Output from Processor, which serves as essential input to Treatment Optimizer, is stored at project_name/processor/Scenario_name/db/Scenario_results.mdb and includes:

- The *harvest_costs* table, which stores cost per acre information generated by OpCost, and
- The *tree_vol_val_by_species_diam_groups* table, which stores key volume and value information by species and diameter groups.

1. To start the BioSum Processor, click the **<Start BioSum Processor>** button.
2. The **Processor Scenario** window will open. The first time BioSum Processor is started, a default scenario will be created titled **scenario1.** If scenario1 has already been defined and you wish to create a new scenario, click **<New>.** Enter a scenario name and description, then **<OK>, to begin work with a new scenario, or <Cancel> to return to the processor scenario selection dialog**

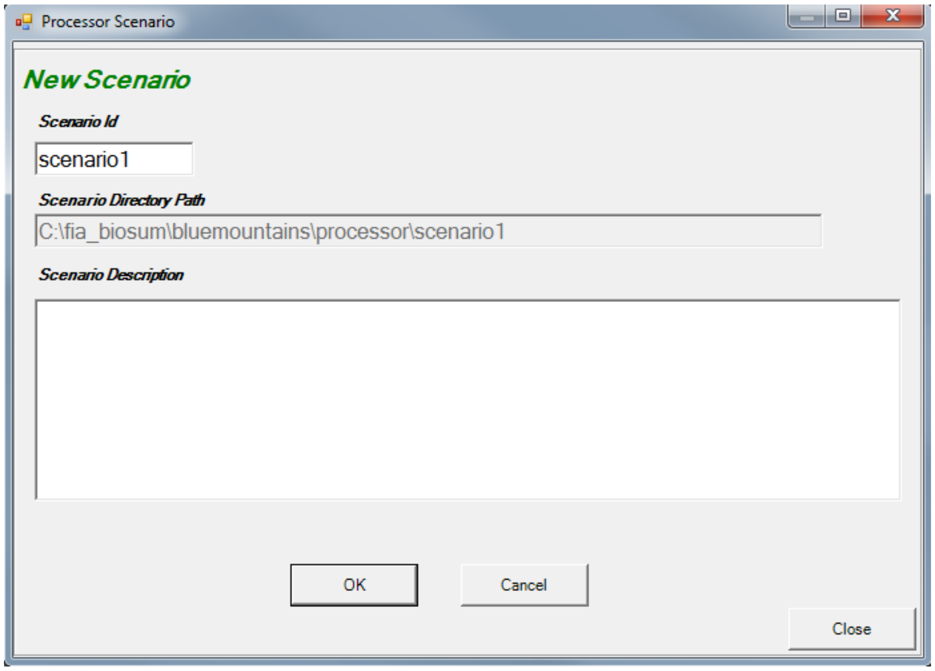
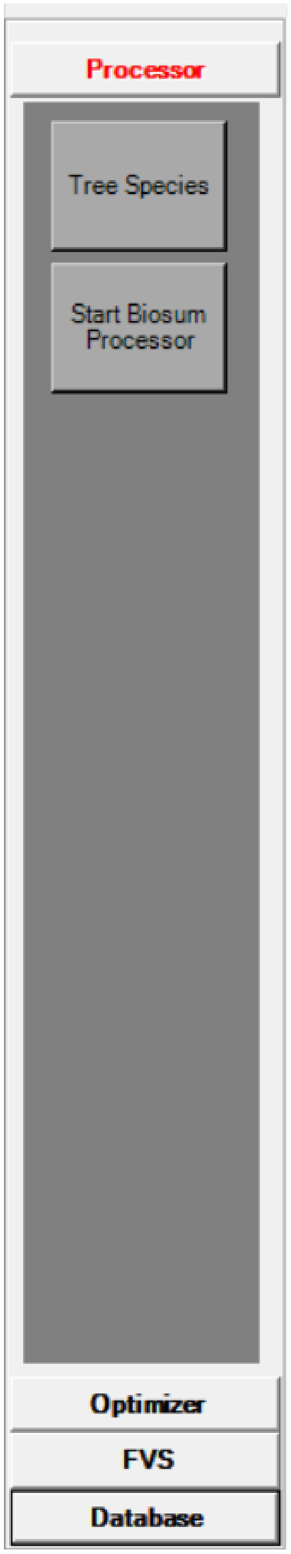
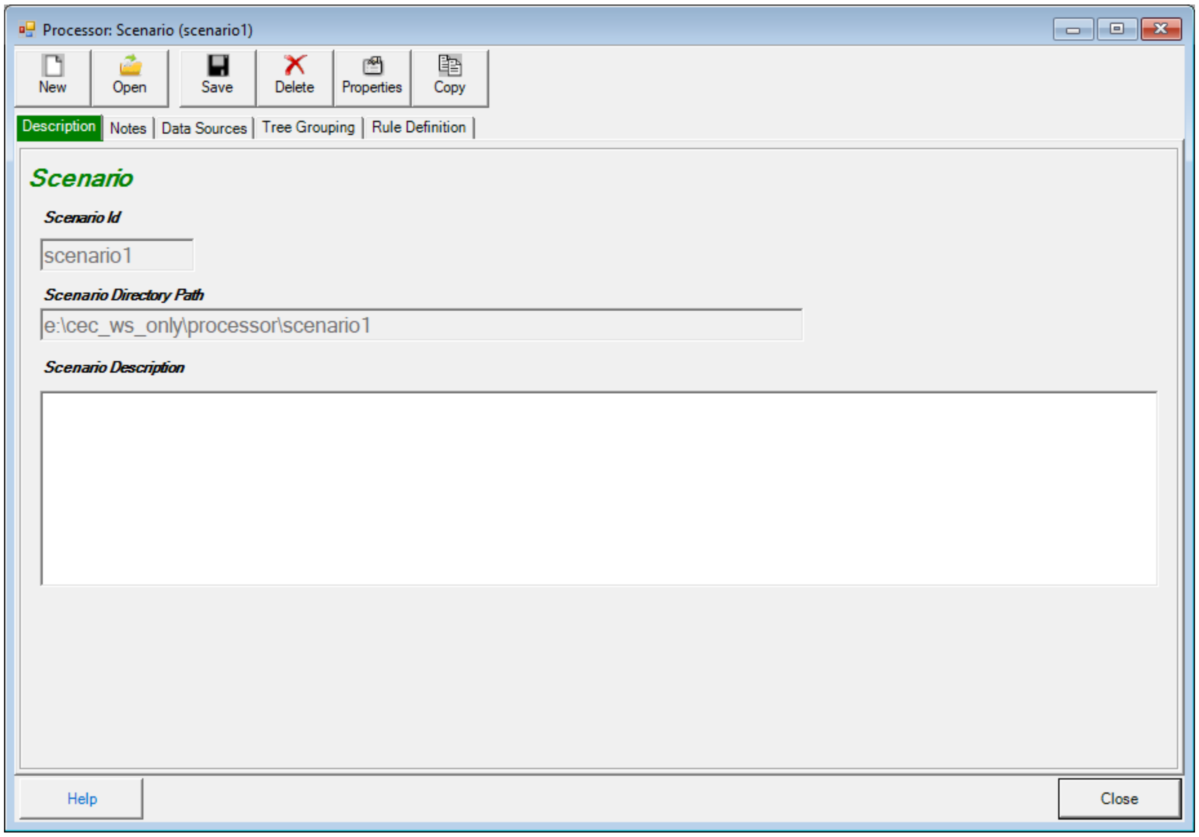


Figure 5.6 – New processor scenario dialog.



3. If not initiating a new processor scenario, select the scenario that you wish to define or refine in the **Scenario List** and then click **<OK>.**



Scenario

Scenario Id

scenario1

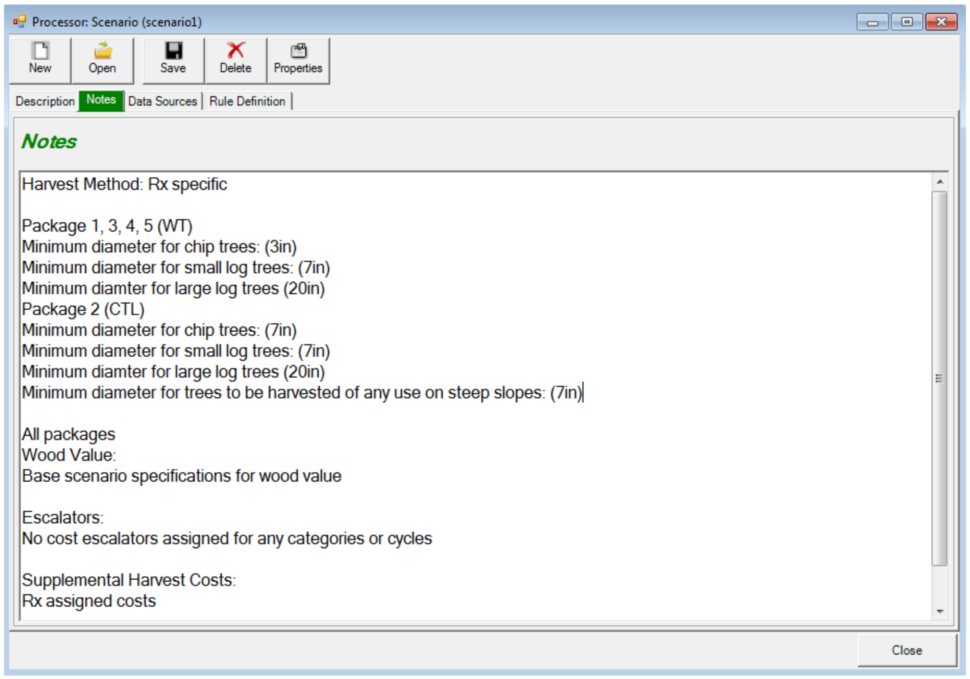
Scenario Directory Path

e:\cec_ws_only\processor\scenario1

Scenario Description

Figure 5.7- Processor scenario selection dialog.

4. The Processor Scenario window has five top-level tabs: Description, Notes, Data Sources, Tree Grouping, and Rule Definition and 6 action buttons (new, open, save, delete, properties and copy).
   a. The function of most of the action buttons is straightforward, but the copy function may be less intuitive. To copy an existing scenario, first create a New scenario. With the new scenario open, select copy, wait a while for choices to appear, then choose a scenario to copy into the currently open scenario. A dialog will appear asking for confirmation of the desire to copy FROM scenario A to scenario B—if the dialog looks right, choose Yes. A Done/OK dialog will display when copying is complete.
   b. The **Description** tab provides a **Scenario Description** text box to enter or revise brief descriptive information about the processor scenario.
   c. The **Notes** tab accesses a text box into which extensive notes may be recorded about the processor scenario for future reference.



Harvest Method: Rx specific

Package 1, 3, 4, 5 (WT)
Minimum diameter for chip trees: (3in)
Minimum diameter for small log trees: (7in)
Minimum diamter for large log trees (20in)
Package 2 (CTL)
Minimum diameter for chip trees: (7in)
Minimum diameter for small log trees: (7in)
Minimum diamter for large log trees (20in)
Minimum diameter for trees to be harvested of any use on steep slopes: (7in)

All packages
Wood Value:
Base scenario specifications for wood value

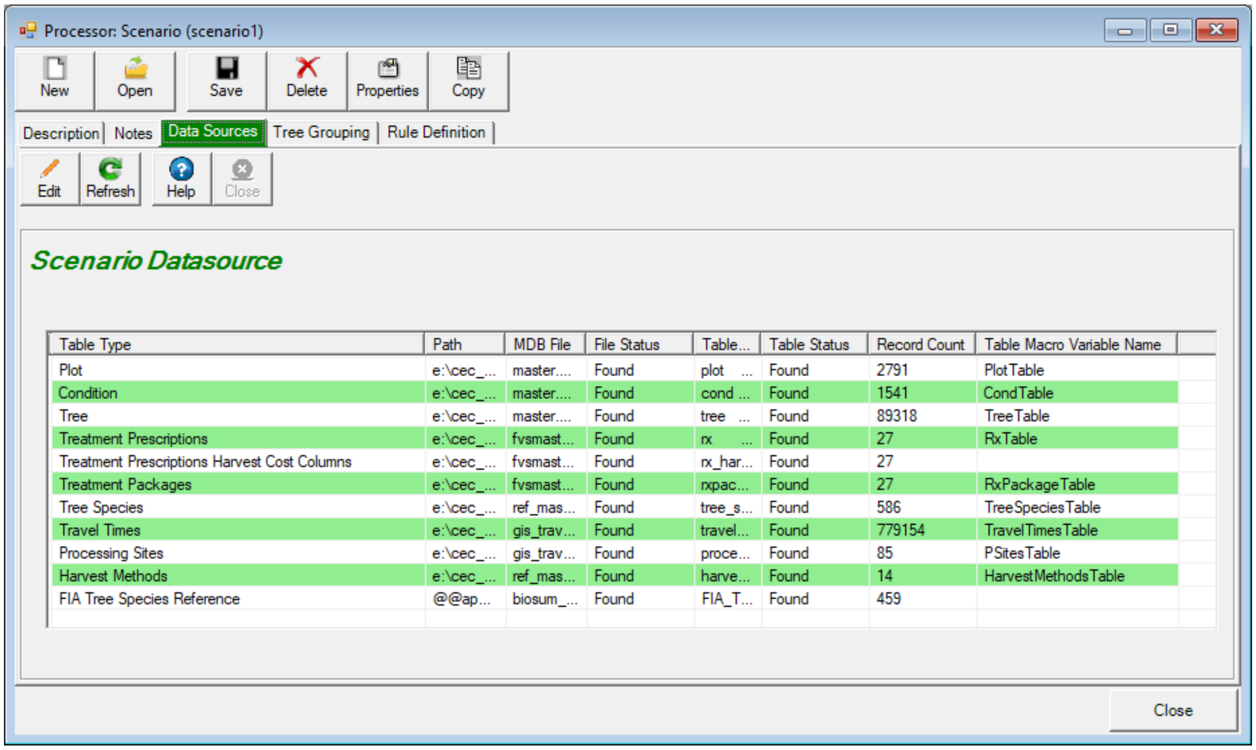
Escalators:
No cost escalators assigned for any categories or cycles

Supplemental Harvest Costs:
Rx assigned costs

Figure 5.8 – Notes tab of Scenario window.

d. The **Data Sources** tab displays pointers (file paths, file names, table names, status, records counts, etc.) to all types of source data used in the Processor scenario. Any of these data sources can be copied to another table or database and the pointer updated to point to the copy instead of the original data source. These copies can then be updated or customized to allow consideration of different situations. For example, if it was desired to customize the diameter groups for a particular processor scenario (and the core analysis scenarios that link to that processor scenario), it is easy to copy the tree_diam_groups table to another location (in the same database file or a different one) and edit the entries in the copy. To edit the pointer to the data source, and to make copies of data sources, select a table type and click **<Edit>.** The **Edit Data Source** window will appear. Here you can move, copy, or rename existing Access database files and tables, and reset links for any table type.

   Caution: Making changes to data sources is an advanced capability, not to be undertaken lightly or by BioSum beginners, and has the potential to be confusing or produce unintended consequences.



Processor: Scenario (scenario1)

New | Open | Save | Delete | Properties | Copy

Description | Notes | Data Sources | Tree Grouping | Rule Definition

Edit | Refresh | Help | Close

Scenario Datasource

Table Type	Path	MDB File	File Status	Table...	Table Status	Record Count	Table Macro Variable Name
Plot	e:\cec_...	master....	Found	plot ...	Found	2791	PlotTable
Condition	e:\cec_...	master....	Found	cond ...	Found	1541	CondTable
Tree	e:\cec_...	master....	Found	tree ...	Found	89318	TreeTable
Treatment Prescriptions	e:\cec_...	fvsmast...	Found	rx ...	Found	27	RxTable
Treatment Prescriptions Harvest Cost Columns	e:\cec_...	fvsmast...	Found	rx_har...	Found	27	
Treatment Packages	e:\cec_...	fvsmast...	Found	rxpac...	Found	27	RxPackageTable
Tree Species	e:\cec_...	ref_mas...	Found	tree_s...	Found	586	TreeSpeciesTable
Travel Times	e:\cec_...	gis_trav...	Found	travel...	Found	779154	TravelTimesTable
Processing Sites	e:\cec_...	gis_trav...	Found	proce...	Found	85	PSitesTable
Harvest Methods	e:\cec_...	ref_mas...	Found	harve...	Found	14	HarvestMethodsTable
FIA Tree Species Reference	@@ap...	biosum_...	Found	FIA_T...	Found	459	

Close

Figure 5.9 – Data Sources tab of the Processor Scenario window.

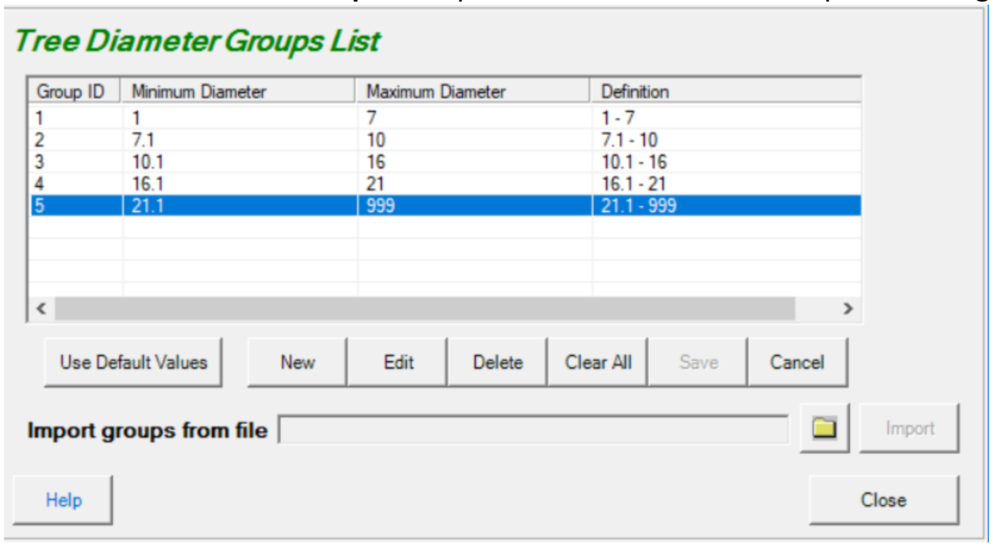
e. The **Tree Grouping** tab provides access to two grouping definitions, accessed by buttons: (1) Tree Diameter Groups and (2) Tree Species Groups—assign species groups. The data entered in these dialogs determines how BioSum summarizes volume, biomass and value by size class (diameter group) and species group. When parameterizing the Processor scenario via Rule definitions, further along in the Processor work flow, you'll have the opportunity to assign merchantable values for harvested trees for each combination of diameter and species group specified here as well as energy wood value per unit weight.

Note that altering these diameter or species group specifications later, after having run Processor (and this is NOT advised), generates a need to re-run the processor scenario and all treatment optimizer scenarios that depend on it; otherwise, the database will be corrupted in that entries in the tree_vol_val_by_species_diam_groups table, and in the treatment optimizer results that are linked to them, will be inconsistent with the most recently entered limits.

Tree Diameter Groups

The form accessed from this button is used to specify diameter groups by which harvested tree volume and value will be summarized. Alternatively, a button on the form allows loading of pre-specified diameter group specifications from a text file.

Click **<Tree Diameter Groups>** to open the Tree Diameter Groups List dialog.


Tree Diameter Groups List

Group ID	Minimum Diameter	Maximum Diameter	Definition
1	1	7	1 - 7
2	7.1	10	7.1 - 10
3	10.1	16	10.1 - 16
4	16.1	21	16.1 - 21
5	21.1	999	21.1 - 999

Figure 5.10 - Tree Diameter Groups entry form.

To enter diameter groups via this form:

i. To add a new diameter group, click **<New>**. Enter minimum and maximum diameter values in the **Tree Diameter Groups** edit window. Values are entered and displayed in inches. An unlimited number of non-overlapping diameter groups can be specified; they must be specified in ascending order. The maximum diameter for the last diameter group entered should be at least as large as the largest tree that might be cut by any prescription in any stand; 999 is a good choice to ensure that all trees will be accounted for in volume, biomass and value summaries. **Important! It is incumbent on the analyst to define diameter groups that comport with brush-cut, chip, small and large tree diameter thresholds limits specified during the work flow defined by the Start BioSum Processor task**. For example, if trees less than 5" diameter are to be directed to the energy wood bin, there should be a break in the diameter group list at 5". Or, if in some packages, trees less than 6" are to be directed to energy wood bin, but in other packages the threshold is 5", diameter class limits should be defined to demarcate one or more groups <5", a group between 5 and 6" and one or more groups >6".

ii. To make changes to an existing diameter group definition, select the group from the list and click **<Edit>**.

iii. To load, or revert to, the default diameter group specification click <**Use Default Values**>; these are listed in Table 5.1.

Table 5.1 – Default diameter group values.

Group ID	Minimum Diameter	Maximum Diameter
1	1	7
2	7.1	10
3	10.1	16
4	16.1	21
5	21.1	999

iv. To delete a diameter group, select that group to highlight it and click **<Delete>**. To delete all diameter classes click **<Clear All>**.

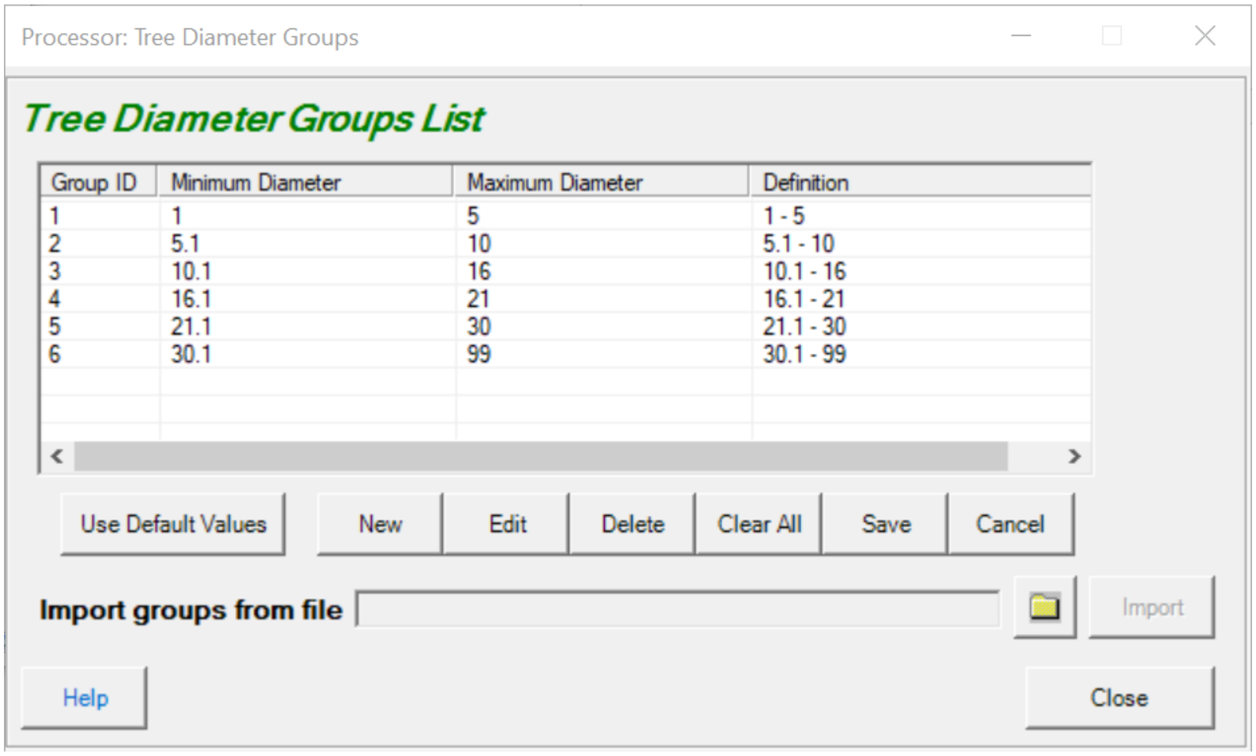
To import diameter class specifications from a text file:

When a large number of diameter classes are needed, specification via the **Tree Diameter Groups** editor may be cumbersome and efficiency may be gained by importing them from a text file. This workflow consists of 4 steps:

i. Using notepad (or a text editor of your choice) specify the maximum diameter for each of the tree species groups. The diameter values should be in ascending order with one value on each line. Biosum tree diameter groups are precise to the first decimal place. Values with more decimal places will be rounded to the closest tenth. For example, the contents of a diameter class specification text file suitable for import might be the following six lines:

```
5
10
16
21
30
99
```

which would generate six diameter classes:


Processor: Tree Diameter Groups

Tree Diameter Groups List

Group ID	Minimum Diameter	Maximum Diameter	Definition
1	1	5	1 - 5
2	5.1	10	5.1 - 10
3	10.1	16	10.1 - 16
4	16.1	21	16.1 - 21
5	21.1	30	21.1 - 30
6	30.1	99	30.1 - 99

Figure 5.11

ii. After you have created and saved the text file to your computer, click the folder button that appears above the close button near the bottom of the screen to specify the files location.

iii. When the **Import groups from file** text box contains the path to your selected file, the <**Import>** button will become enabled. Click this button to begin the import process. As the warning message indicates, the imported groups will overwrite the existing contents of the **Tree Diameter Groups List**.

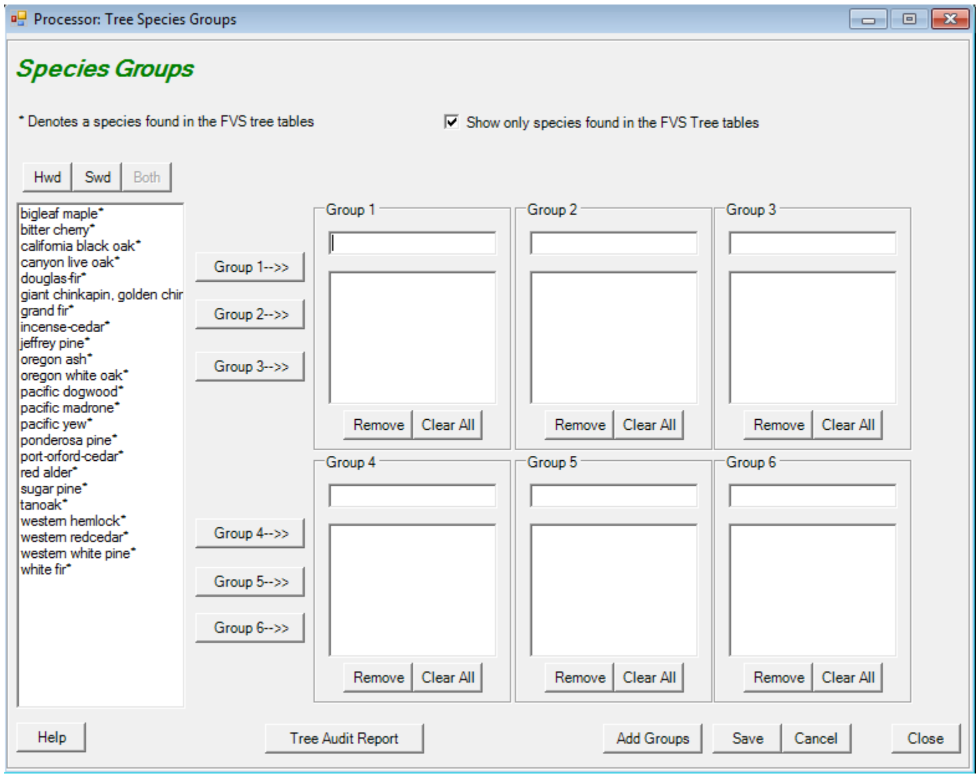
iv. Biosum will determine the Group ID, Minimum Diameter, and Definition values based on the contents of the text file. The default minimum diameter for Group ID 1 is 1. This can be changed after the import process by editing tree species group 1. The minimum diameter for the remaining groups is the maximum diameter for the preceding group + 0.1. BioSum does not allow gaps between tree diameter classes.

Whichever way diameter class information was specified (entering the form or importing from text file), if satisfied with the values in the Tree Diameter Groups List, click **<Save>** to save the values. Alternatively, click **<Cancel>** and close the **Tree Diameter Groups** window to revert to the previous diameter class specification (if any).

Tree Species Groups

BioSum summarizes volume, biomass and value by size class (diameter group) and species group. When parameterizing Processor towards the end of this module, merchantable values for harvested trees will need to be assigned for each combination of diameter and species group. Before that can happen, species groups must be assigned.

i. Click the **<Tree Species Groups>** button to display the **Species Groups** window.



* Denotes a species found in the FVS tree tables

[✓] Show only species found in the FVS Tree tables

bigleaf maple*
bitter cherry*
california black oak*
canyon live oak*
douglas-fir*
giant chinkapin, golden chir
grand fir*
incense-cedar*
jeffrey pine*
oregon ash*
oregon white oak*
pacific dogwood*
pacific madrone*
pacific yew*
ponderosa pine*
port-orford-cedar*
red alder*
sugar pine*
tanoak*
western hemlock*
western redcedar*
western white pine*
white fir*

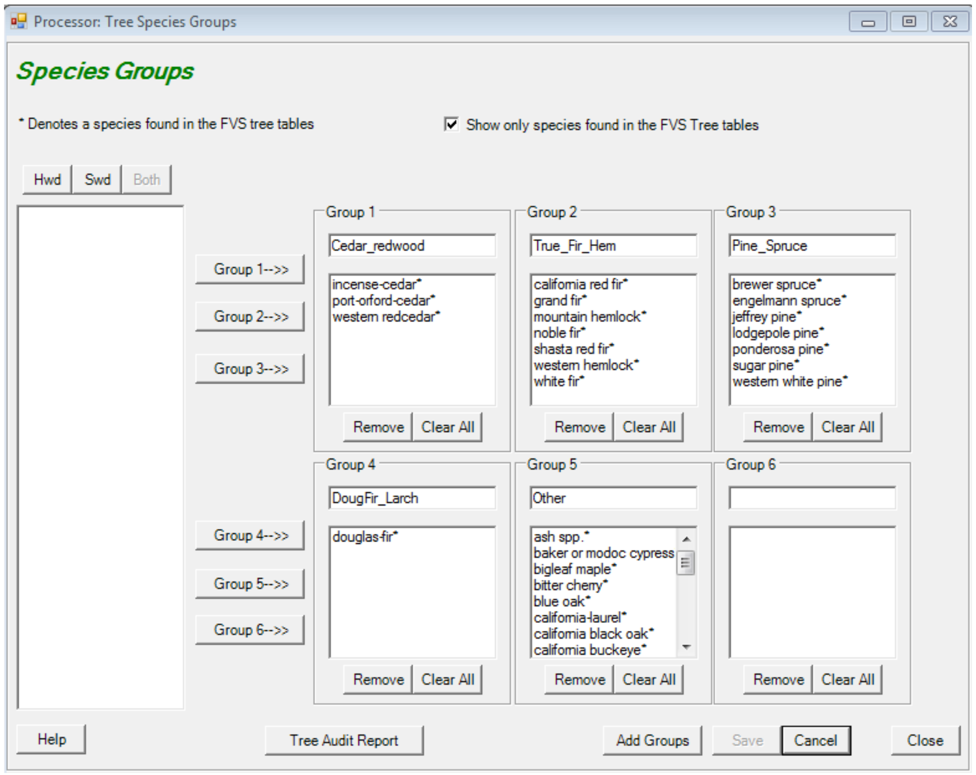
Figure 5.12 – Tree species groups window.

ii. Click the check box at the top of the window, next to **“Show only species found in the FVS Tree Tables”** to limit the display of tree species common names to only those present in the project dataset.

iii. Select a species from the list on the left and click any of the **<Group>** buttons to insert the species into that group. Note that species are removed from the list on the left as they are entered into groups. Enter a group name in the text box above each group species listing; blank names are not allowed. The **<Remove>** button will remove the selected species from the group and the **<Clear All>** button will remove all species from the group, returning them to the full list of unallocated species at the left edge of the species groups form. Additional groups can be added by clicking the **<Add Groups>** button at the bottom of the form.

iv. The **<Tree Audit Report>** button pulls up the Tree Species audit window, should the analyst need to edit tree species records or rerun an audit while using this form.

v. When finished assigning species to groups, the list of unallocated species will be empty. Click **<Save>** and then **<Close>** to complete this task.


Processor: Tree Species Groups

Species Groups

* Denotes a species found in the FVS tree tables

☑ Show only species found in the FVS Tree tables

Group 1: Cedar_redwood	Group 2: True_Fir_Hem	Group 3: Pine_Spruce
incense-cedar*, port-orford-cedar*, western redcedar*	california red fir*, grand fir*, mountain hemlock*, noble fir*, shasta red fir*, western hemlock*, white fir*	brewer spruce*, engelmann spruce*, jeffrey pine*, lodgepole pine*, ponderosa pine*, sugar pine*, western white pine*
Group 4: DougFir_Larch	Group 5: Other	Group 6:
douglas-fir*	ash spp.*, baker or modoc cypress, bigleaf maple*, bitter cherry*, blue oak*, california-laurel*, california black oak*, california buckeye*	

Figure 5.13 - All species have been grouped and group names have been added.

f. The **Rule Definitions** tab is a trigger that, after a moderate delay, the length of which depends on the size of the project and can be as long as a few minutes, displays six, second-tier tabs for entering processor scenario parameters and selecting which packages and stands to process: **Harvest Method**, **Move-In Cost**, **Wood Value**, **Escalators**, **Supplemental Harvest Costs**, and **Run**. Each of these second tier tabs access parameter settings that control some aspect of the Processor Scenario. If a scenario, say TimberPrice1 is copied into a new scenario name, say TimberPrice2, all of these parameter settings will be copied along with it. Then, if interested, for example, in the effect on treatment feasibility of higher or lower prices for merchantable and/or energy wood, it's easy to modify wood values stored in the TimberPrice2 scenario, re-run BioSum, and see how the effect plays out.

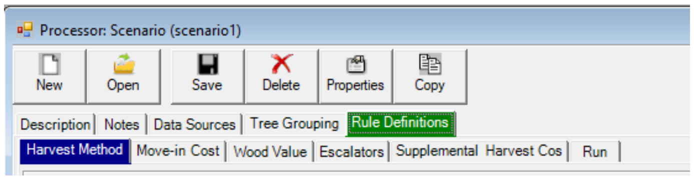
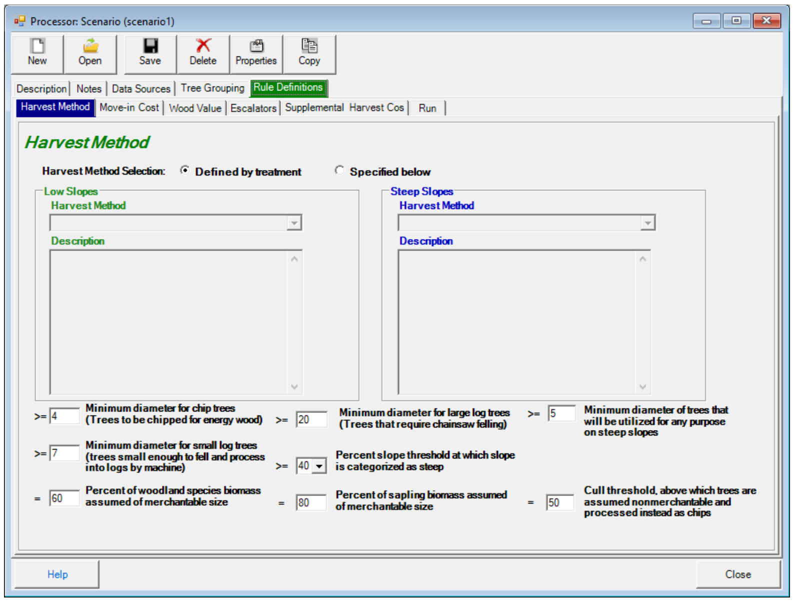


Figure 5.14 – Harvest Method is the first second-tier tab, nested beneath the Rule Definitions tab of the Processor Scenario window.

i. **Harvest Method:** Every stand-treatment combination must have an associated harvest method for OpCost to simulate costs when the information is passed to OpCost by BioSum Processor. Harvest methods can be assigned within the BioSum FVS module when prescriptions (Rx) are specified, or here, via the Harvest Method fields that, when populated, apply to all treatments in the scenario (Figure 5.19).
   1. By default, BioSum Processor will use the harvest methods that were defined for each treatment during treatment creation (FVS module). The <**Defined by treatment**> radio button will be toggled as selected.
   2. To instead define two standard harvest methods (one for steep, one for gentle slopes) that will be applied for all treatments in the processor session, select the radio button <**Specified below**>. Select **a Low Slope Harvest Method** and a **Steep Slope Harvest Method** for the scenario using the drop-down menus. A description of each method is provided in the **Description** text box. A current list and description of the harvest methods supported by BioSum and OpCost can also be found in **Appendix A** of this User Guide.
   3. If different than the displayed default values, three tree size parameters can be defined by revising the entries in their respective entry fields: 1) **Minimum diameter for chip trees**—trees smaller than this may be harvested as part of the prescription but will not be collected to the landing for utilization; 2) **Minimum diameter for small log trees**—these are trees that can be harvested and processed by machine (if harvested under a mechanized harvest system) and which have boles potentially suitable for utilization as merchantable logs (not chips); 3) **Minimum diameter for large log trees**—trees larger than this cannot be harvested and processed by mechanized equipment; rather they must be manually felled, bucked and yarded or skidded to the landing. VERY IMPORTANT: As of May 2023, the equations in OpCost that estimate mechanized harvesting cost will fail if the quadratic mean diameter of small log trees is 21 inches or greater. While trees larger than this threshold may, in fact, be harvested using a mechanized harvester, currently available equations are not capable of estimating the machine time required for harvesting such trees and they should be handled instead as large log trees (which are modeled in OpCost as manually felled). Stands with a QMD for small log trees of 21 inches or larger will be dropped from analysis until new logging cost equations are developed and incorporated into OpCost. Note that “chip trees”, as used here, refers to size class of trees from a harvest cost calculation point of view—these trees may be chipped and utilized as energy wood or not chipped and utilized as merchantable wood, or not utilized at all, depending on other settings selected by the user in the Processor module. Also, note that chipped wood (to be used for bioenergy, for

example) may include either the boles or entirety of trees in the chip size class (depending on harvest system) and also the tops and limbs of merchantable sized trees (that may be costed as small or large log trees) and the entirety of small and large log trees of noncommercial species.

4. Define the **Percent slope threshold at which slope is categorized as steep.** Stands on slopes greater than or equal to this threshold will be modeled as harvested using a harvest method appropriate for steep slopes, whether that method is determined at prescription level or for the processor scenario.
5. The **Percent of woodland species biomass assumed of merchantable size** parameter determines how BioSum and OpCost account for harvesting costs for these species under different harvest methods. This assumption is needed because FIA does not have separate estimates of volume or biomass for merchantable and non-merchantable parts of these trees. Available equations estimate total volume and biomass.
6. The parameter **Percent of sapling biomass assumed of merchantable size** is needed because trees smaller than 5 inches dbh have no calculated bole volume, only total volume, in FIA databases. Whether all or parts of these small trees are utilized as merchantable logs, energy wood or not at all, the percent of their volume in boles is needed to accurately estimate treatment costs, and this parameter provides the basis for this.
7. The **Minimum Diameter of Trees to be harvested for All Uses ON STEEP SLOPES,** tells OpCost which trees need to be accounted as being transported to the landing, versus cut and left.
8. The **Cull threshold, above which trees are assumed nonmerchantable and processed instead as chips**, enables customization of how much of a tree must be rated as cull before BioSum assumes that it has no merchantable value and reverts to utilization as energy wood.



Harvest Method

Harvest Method Selection: [✓ Defined by treatment] Specified below

Low Slopes
Harvest Method:
Description:

Steep Slopes
Harvest Method:
Description:

>= 4 Minimum diameter for chip trees (Trees to be chipped for energy wood)	>= 20 Minimum diameter for large log trees (Trees that require chainsaw felling)	>= 5 Minimum diameter of trees that will be utilized for any purpose on steep slopes
>= 7 Minimum diameter for small log trees (trees small enough to fell and process into logs by machine)	>= 40 Percent slope threshold at which slope is categorized as steep	
= 60 Percent of woodland species biomass assumed of merchantable size	= 80 Percent of sapling biomass assumed of merchantable size	= 50 Cull threshold, above which trees are assumed nonmerchantable and processed instead as chips

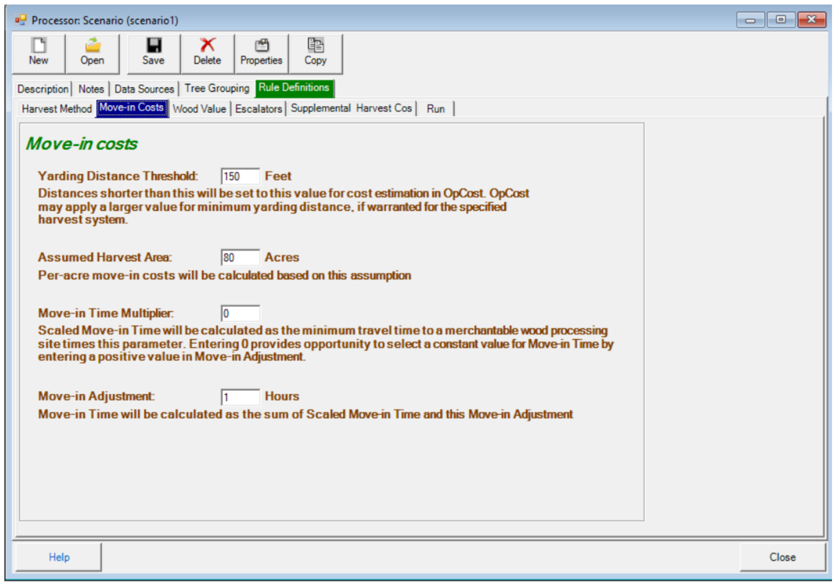
Figure 5.15 – The harvest method tab parameters will need to be specified by the user when choosing to override the prescription specific harvest system choices with a single pair of systems for all treatments (one for low and one for steep slopes).

ii. **Move-in Costs:** This form contains three parameters that can be used to account for the costs of moving logging equipment to a forest operation and one, **yarding distance threshold**, that does not relate to move-in costs, but does not readily fit within any other dialog. To estimate move-in costs, OpCost must be supplied with a rough average of the time required to move forest operations equipment (e.g., feller-bunchers, forwarders, chippers) from a previous location (either a home-base or another forest operation) to the location of the FIA plot where the stand exists. It must also know the number of pieces of equipment that will need to be moved in—a statistic that OpCost derives from the harvest system chosen for implementation—and, to obtain per acre move-in costs, the area of a forest operation over which the move-in costs can be distributed.

1. The yarding distance threshold overrides the yarding distance calculated between the FIA plot and the nearest point on the road network to address issues that arise when short yarding distances are passed to the operations cost equations within OpCost. For example, if the calculated distance for a particular stand, based on FIA plot location and road position, is 3 feet, a threshold value of 100 feet will modify that 3 to 100 when that stand's data is passed to OpCost. For some of the equations used in OpCost to estimate yarding or forwarding costs for some harvest systems, providing small values for average yarding distance produces anomalous results because the values are outside the range of data that were used to fit the equations. Setting the threshold to 150 feet or greater will likely avoid these anomalies, though for some harvest systems (e.g., helicopter-

based ones), Processor and/or OpCost may internally override even this threshold with a higher threshold value.

2. Assumed Harvest Area, in acres, is used to convert move-in costs for the stand to a per acre cost that can be summed with the operations and supplemental harvest costs per acre to obtain total per acre costs. The parameter value should reflect the typical size of forest operations that implement the kinds of treatments being modeled.
3. Given the uncertainty concerning exactly where equipment might be based, one option for obtaining a rough approximation of move-in time (which OpCost uses to obtain move-in cost) is the simplistic assumption that equipment "lives" where merchantable wood processing facilities exist (i.e., in at least a semi-developed area), and specifically, at the nearest facility for processing merchantable wood. To model that assumption, the analyst would enter a "1" for the **Move-in Time Multiplier**. Alternatively, any non-negative value other than 1 can be selected to represent at time that is a fraction or multiple of the time to the closest wood processing facility. If desiring a move-in cost estimate that depends only on travel time to the nearest wood processing facility, ensure that the **Move-In Adjustment** is set to zero.
4. Another option is to assume a fixed move-in time for all stands that is independent of where stands are relative to wood processing facilities. To do so, set **Move-in Time Multiplier** to zero (effectively cancelling out the use of travel time in the calculation) and set the fixed **Move-in Adjustment** time (in decimal hours) to a non-negative value.
5. The third option is to combine approaches—e.g., a value of 0.5 for **Move-in Multiplier** and 0.75 for **Move-in Adjustment** would calculate move-in time as the sum of half the travel time to the nearest merchantable wood processing facility and 45 minutes.



Move-in costs

Yarding Distance Threshold: 150 Feet

Distances shorter than this will be set to this value for cost estimation in OpCost. OpCost may apply a larger value for minimum yarding distance, if warranted for the specified harvest system.

Assumed Harvest Area: 80 Acres

Per-acre move-in costs will be calculated based on this assumption

Move-in Time Multiplier: 0

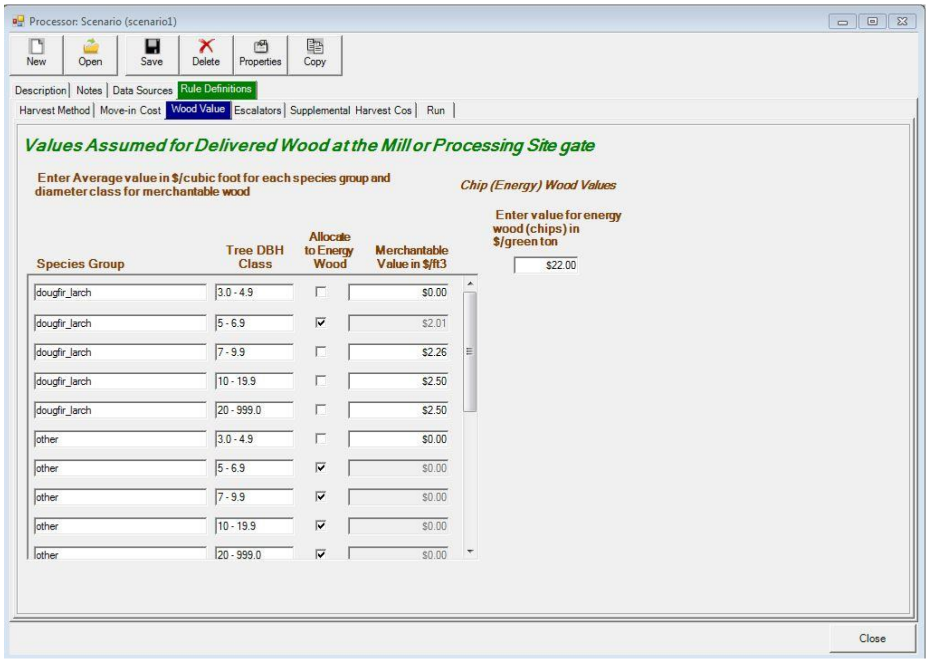
Scaled Move-in Time will be calculated as the minimum travel time to a merchantable wood processing site times this parameter. Entering 0 provides opportunity to select a constant value for Move-in Time by entering a positive value in Move-in Adjustment.

Move-in Adjustment: 1 Hours

Move-in Time will be calculated as the sum of Scaled Move-in Time and this Move-in Adjustment

Figure 5.16: The Move-in Costs form contains three parameters that can be used to account for the costs of moving logging equipment to a forest operation and one that does not.

iii. **Wood Value:** This form provides for entering/revising the merchantable value in dollars per cubic foot, at the mill gate, for each combination of species and diameter group represented in the Processor scenario, and a value in dollars per green ton, at the delivery point (e.g., a biomass-based energy generating facility), for chipped residues (the material chipped at a treatment site and loaded onto a chip truck). The latter should be entered in the field labeled "Energy value for energy wood (chips) in $/green ton". To allocate a species/diameter group combination to energy wood, use the **Allocate to Energy Wood** checkbox. Note: if silvicultural sequences are designed to have different thresholds, or wood allocation choices, it is possible to select parameters, and run processor for one set of sequences with one set of wood value parameters, then run another set of sequences with a modified set of parameters.



Values Assumed for Delivered Wood at the Mill or Processing Site gate

Enter Average value in $/cubic foot for each species group and diameter class for merchantable wood

Species Group	Tree DBH Class	Allocate to Energy Wood	Merchantable Value in $/ft3
dougfir_larch	3.0 - 4.9	☐	$0.00
dougfir_larch	5 - 6.9	✓	$2.01
dougfir_larch	7 - 9.9	☐	$2.26
dougfir_larch	10 - 19.9	☐	$2.50
dougfir_larch	20 - 999.0	☐	$2.50
other	3.0 - 4.9	☐	$0.00
other	5 - 6.9	✓	$0.00
other	7 - 9.9	✓	$0.00
other	10 - 19.9	✓	$0.00
other	20 - 999.0	✓	$0.00

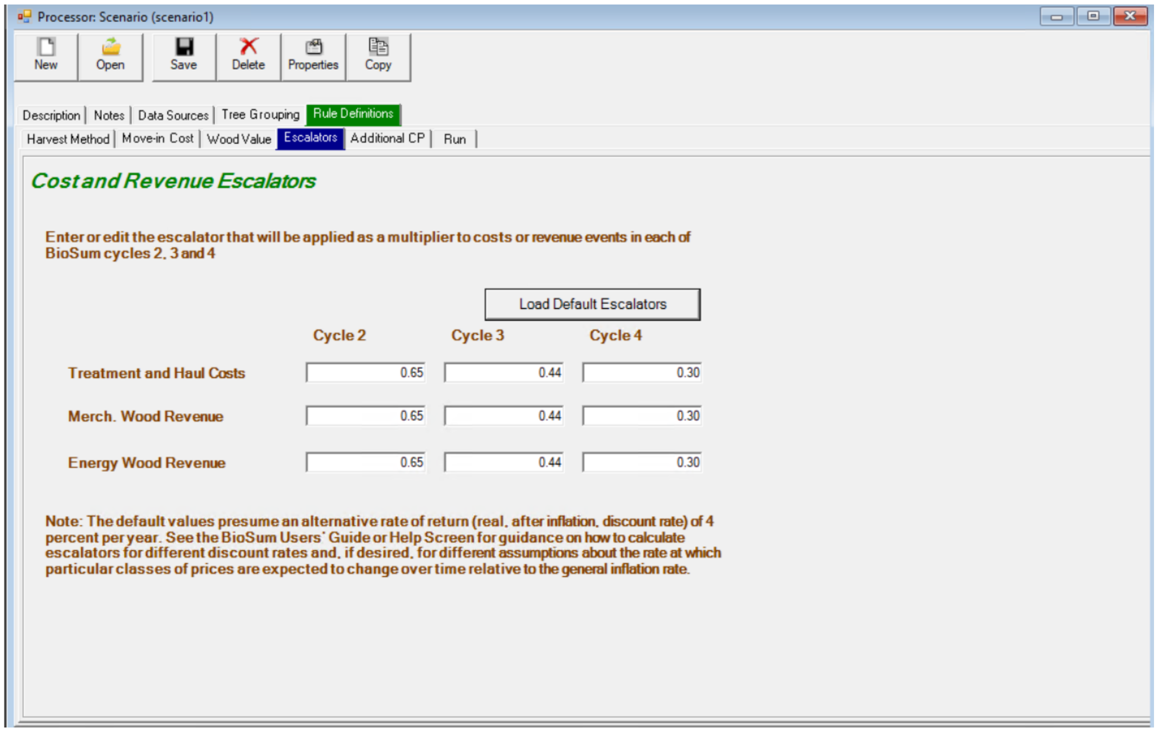
Chip (Energy) Wood Values

Enter value for energy wood (chips) in $/green ton: $22.00

Figure 5.17 – Wood Value tab, specify values assumed for delivered wood by diameter class and species group.

iv. **Escalators:** The escalators tab accesses a dialog for specifying multipliers that can be applied to costs incurred and revenues received at BioSum cycles 2, 3 and 4 to account for 1) a discount rate (alternative rate of return) and/or 2) anticipated differential appreciation in prices or costs among merchantable wood, bioenergy feedstocks (chips) and factor costs associated with implementing silvicultural prescriptions and hauling harvested wood. Either or both of these simulation elements can be accounted for, but if it is to be both, then the multipliers supplied by the analyst need to account for both elements. If not intending to account for either of these elements (rarely an appropriate choice, given that some positive discount rate almost always applies), all multipliers would be set to one, as they are in a new project or processor scenario. Note that there is no reason to specify a multiplier for BioSum cycle 1 because, given that it is assumed that costs and revenues occur at the beginning of cycle 1 (i.e., year 1, or close to it), any discounting or inflation between the vantage of the time point when the simulation is conducted and the realization of those costs and revenues isn't significant.

Escalators that encode a default annual discount rate of 4 percent for cycles 2-4 and assume no differential in the inflation rates for prices of wood and bioenergy and treatment and haul costs can be summoned with the "Load Default Escalators" button to replace any currently specified escalators. These will reflect the nominal cycle length (of 5 or 10 years) selected during the Rx Packages definitions. Only one cycle length should be selected for a given BioSum project when specifying Rx Packages. If more than one cycle length was specified, default escalators will be assigned based on the longer of the cycle lengths defined (i.e., 10 years).



Cost and Revenue Escalators

Enter or edit the escalator that will be applied as a multiplier to costs or revenue events in each of BioSum cycles 2, 3 and 4

	Cycle 2	Cycle 3	Cycle 4
Treatment and Haul Costs	0.65	0.44	0.30
Merch. Wood Revenue	0.65	0.44	0.30
Energy Wood Revenue	0.65	0.44	0.30

Note: The default values presume an alternative rate of return (real, after inflation, discount rate) of 4 percent per year. See the BioSum Users' Guide or Help Screen for guidance on how to calculate escalators for different discount rates and, if desired, for different assumptions about the rate at which particular classes of prices are expected to change over time relative to the general inflation rate.

Figure 5.18 – Cost and revenue escalators tab showing default escalators representing a 4 percent discount rate.

Case 1: Discount rate—this is the most typical use of price and cost escalators in a BioSum analysis. The intent of this use case is to account for the alternative rate of return on capital that should be assumed in an investment analysis. This is especially relevant to an analysis of management on private lands, though it may also be relevant on public lands. The discount rate accounts for the delivery of, say, $1000 per acre in net revenue eleven years from now, at the beginning of the 2nd BioSum cycle (i.e., 2034), being worth less to a forest owner than receiving that same revenue today (in 2023). The present value (PV), that is to say, the value today, of $1000 received in 2034 is given by applying to the future value (FV) the formula:

$$PV = FV(1/(1+i)^{t})$$

where *t* is the number of years separating today and the future value (11 years in this case) and *i* is the interest rate, a.k.a. alternative rate of return. For this example, if an alternative rate of return of 4% is considered to be the best, deflated (that is to say, adjusted for inflation) return available today for capital invested in an asset with equivalent risk to that observed in forestry, PV is $1000(1/(1.04)^{11})$ = 1000 * 0.650 = \$650. The present value of \$1000 received at the beginning of the 4th BioSum cycle (say 31 years from now) would be \$300.

Discount rates affect how streams of costs and revenues associated with multiple, potentially desirable silvicultural sequences that could be applied in a stand are evaluated. When a silvicultural sequence must exceed a feasibility threshold (for example, present net revenue > 0) to be considered potentially optimal and/or when an economic criterion determines which sequence is best or is used to break ties, then these evaluations can be critical. With a 4 percent discount rate, a sequence that imposed costs of \$400 per acre at year 1 (say for precommercial thinning) and returned \$1000 per are in net revenue (sales of wood minus harvest and haul costs) in year 31 would have a negative present net revenue because the \$1000 discounted to the present would be only \$300 per acre of revenue, which is less than the \$400 per acre expended on precommercial thinning at year 1. It might still be a desirable and viable option if the economic threshold specified in a scenario called for losing less than or equal to \$100 per acre.

Of course, if the inflation rate (represented by _r_) is not zero, then net revenue in today's dollars would be greater at year 31 in nominal currency for that year. Take the example of a prospective (at year 31) \$5000 per acre in sales of merchantable and energy wood and harvest and haul costs that sum to \$4000, netting to +1000 using today's prices and costs. If the expected rate of inflation is 8 percent, as it was in 2022, the net revenue at year 31 in nominal 2054 dollars would be $1000*(1+r)t = 1000(1.08^{31})$= \$10,868. However, future nominal dollars are of no great interest because financial analysis requires that all costs and revenues be "moved" to the same point in time for comparison, and typically, the present time is selected for this comparison point. Thus, that future net revenue of \$10,868 in nominal 2054 currency would need to be discounted by dividing by (1.08^{31}), bringing us right back to \$1000 in today's dollars. This is why in most circumstances, it is not necessary to account for the inflation rate *r* at all (but it is necessary to account for the discount rate, *i*).

The values to enter in the escalators registers on this screen should be calculated as $(1/(1+i)^{t})$ for each value of t (e.g., 11, 21 and 31, if those are the first years of BioSum cycles 2, 3 and 4). In our example of a 4 percent real discount rate (net of inflation), these would compute as 0.6496, 0.4388 and 0.2965 for years 11, 21 and 31 (Cycles 2-4). For each column (BioSum cycle), enter the one multiplier associated with that cycle for every row—Operating costs, Merch and Energy wood revenue so that all costs and revenues are consistently discounted. The values will differ among rows only for scenarios in which inflation is assumed to behave differently among these three cost and revenue components (see next section).

Case 2: Differential appreciation—This use of the escalator fields is designed to enable simulation of scenarios in which users anticipate differentials in price and cost appreciation for some prices and costs relative to others. For example, consider this scenario: the crushing urgency of combatting climate change leads to dramatic and sustained subsidies, tax breaks and other policies that promote using wood as a construction material in lieu of concrete and steel because it avoids fossil C emissions and sequesters carbon for a very long time. It is easy to see how such demand-stimulating policies could result in the value of merchantable wood rising faster than inflation (and faster than both bioenergy prices and harvest and haul costs). Suppose a general inflation rate (applicable to energy wood revenue and costs of production) of 8 percent and a merchantable wood price inflation rate of 10 percent. In the simple, though unrealistic, case of needing to account for this differential but not needing to account for an alternative rate of return (i.e., a discount rate set at zero), we use the differential between the wood price inflation and general inflation, 2 percent, as the basis of computing the escalator for wood products prices. Because wood products prices are expected to enjoy a premium over and above general inflation, we would set multipliers as $(1+r)^t$, which in this case would be $(1.02)^{11} = 1.243$ for cost and revenue amounts associated with cycle 2, $(1.02)^{21} = 1.516$ for cycle 3 and $(1.02)^{31} = 1.848$ for cycle 4, if we assume that this differential persists over the entire simulation. If, instead, we believe that this differential will be a temporary ten-year "bump" that doesn't begin until ten years from now, then we would set the escalator to 1 for cycle 2, and 1.243 for cycles 3 and 4. Note that though wood product price appreciation reverts to the general inflation rate for the 4th cycle, it was elevated for 10 years and thus stays ahead of general inflation going forward – hence the continuation of the 1.243 escalator in cycle 4. All other escalators (for bioenergy and costs) remain set at 1 for all cycles because they track general inflation. Note that we don't use 8 and 10 to calculate escalators for all prices and costs because the 8 percent general inflation is already accounted—if we set the escalator for wood prices based on 10 percent, we would then have to discount by 8 percent inflation to bring it back to present value terms—which is the same as the net 2 percent used in this example. If we believe that policies of promotion will also lead bioenergy prices to rise faster than inflation, but somewhat less so than for wood (say 9 percent), we could calculate bioenergy escalators as a net 1 percent/year escalator (1.01^t).

Case 3: Discount rate *and* differential appreciation—As indicated earlier, escalators would almost never be based on differential rates alone because discount rates are an incontrovertible reality of valuation decisions. Economic theory and empirical evidence have confirmed that a dollar in the future is almost never worth as much as a dollar today, whether as a cost or a revenue, and whether it is a public agency or a private entity making those judgements. Different entities may have different rates of return, perhaps owing to different assumptions or risk profiles, but accounting for such differences is beyond the scope of what BioSum supports within the model and would have to be implemented exogenously if desired.

Within BioSum, the discount rate and differential appreciation can be accounted for simultaneously by the escalator settings. Consider this scenario: Discount rate of i = 4 percent, and an expectation that wood and bioenergy prices will track general inflation rates but treatment and haul costs will rise 1 percent faster than general inflation owing to their dependence on fossil fuel inputs, for the duration of the simulation (r = 1 percent). Both scenario aspects can be represented in the escalators by combining the interest rates, as appropriate for each economic element. For wood and bioenergy prices, the escalators (multipliers) would be calculated, if $i > r$ (discount rate greater than positive differential above inflation for the relevant cost or price), as $(1/(1+(i\text{-}r))^t)$, where t takes on the year in the cycle at which costs or prices occurs. In the very unlikely event that $r > i$, the multiplier would be calculated as $(1+(r\text{-}i))^t$ and the general case (that works regardless of relative magnitude of i and r) is $(1+(r))^t /(1+i)^t$

Escalator calculation tool—An Excel workbook is available at http://biosum.info/downloads/processor/Escalators.xlsx in which to enter assumptions about discount rate and, if warranted, differential appreciation rates per price and cost category, per cycle. The values calculated on that sheet can be transcribed into the Cost and Revenue Escalators dialog. A button on this dialog allows you to choose the default settings (4 percent discount rate, and no differential appreciation rates), which will then be applied, based also on the current cycle length specified in the prescription definition section of BioSum's FVS module.

v. **Additional CPA:** Recall that within the FVS module, additional cost per acre *flags* (not the actual costs) can be specified when prescriptions (Rxs) are defined. When coding the FVS simulation of packages containing prescriptions with additional CPAs, these flags were to have been set to be populated with 1's and 0's each cycle depending on whether the cost of the additional activity is to be incurred. ***This*** tab provides an interface for actually populating the BioSum cost variables associated with these cost components with analyst-assigned values.

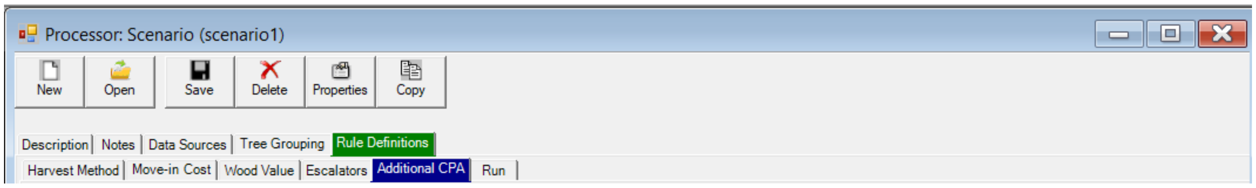
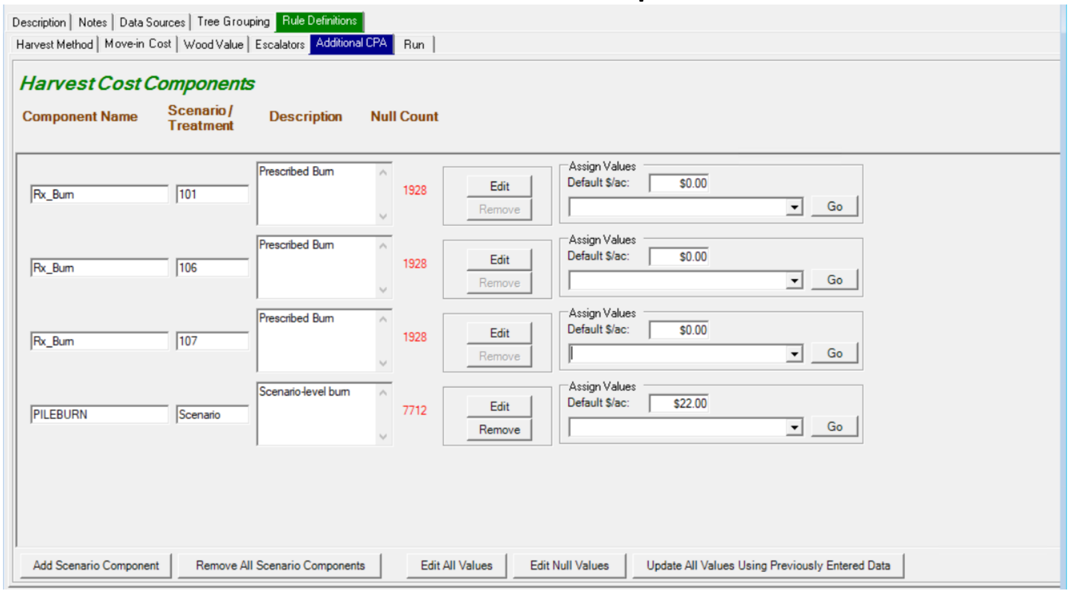


Figure 5.19 –Additional CPA tab—assign costs to previously defined cost components specific to each treatment and/or define and populate scenario harvest cost components that apply to all packages modeled in the processor scenario.

1. To assign a value to an Rx Harvest Cost Component that was created at the time the Rx was defined, find that component in the **Component Name** column (you may need to scroll if there are several). Optionally, the description for that Cost Component can be modified in the **Description** text box. The cost per acre MUST be defined in the **Assign Values: Default cost/ac** section to assess a cost. For Rx Harvest Cost Components, this is the final required step.

2. To add a new cost component to be applied to ALL treatments in this processor scenario, click **<Add Scenario Component>.** When the Harvest Cost window appears, select a cost component from the drop-down list, or enter a new cost component, then click **<OK>.** Scenario components can be edited or removed by clicking the **<Edit>** or **<Remove>** buttons found next to the **Component Name** section.

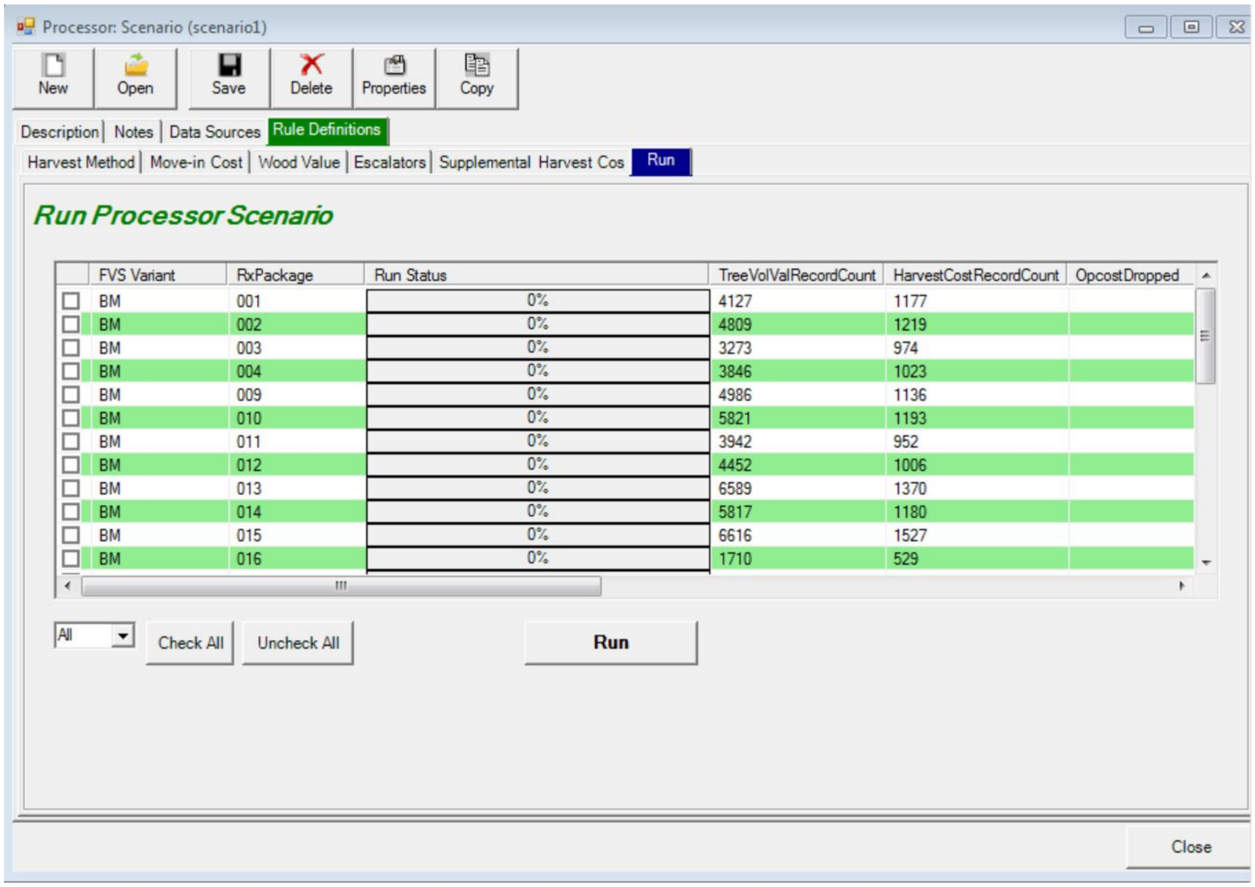


Component Name	Scenario / Treatment	Description	Null Count	Assign Values Default $/ac:
Rx_Burn	101	Prescribed Burn	1928	$0.00
Rx_Burn	106	Prescribed Burn	1928	$0.00
Rx_Burn	107	Prescribed Burn	1928	$0.00
PILEBURN	Scenario	Scenario-level burn	7712	$22.00

Figure 5.20 –Harvest Cost Components— Define and populate Scenario harvest cost components that should be applied to all packages harvested and modeled in the processor scenario.

3. Define the default dollar amount of the scenario component in the **Assign Values: Default cost/ac** section to assess a cost.
4. Use the pick list next to the <**Go**> button to select an option for assigning scenario component costs at the stand level. The CPA can also be reviewed/edited by using MS Access to examine the scenario_additional_harvest_costs table. This table can be found in the \processor\db\scenario_processor_rule_definitions.mdb and contains a column for each harvest cost component.

vi. Run: Once all the scenario rule definitions are complete, a processor simulation can begin. Note: Individual RxPackages within a single project may contain different Processor parameters by running a subset of rxpackages. For example, it may be the case that there are different minimum diameter thresholds in the rxpackages that use a Whole Tree harvest system, versus those using a Cut to Length method. In this case, users can input the Whole Tree parameters, run those packages through Processor, change the inputs to reflect a Cut to Length system, and then run the remaining packages.



Run Processor Scenario

	FVS Variant	RxPackage	Run Status	TreeVolValRecordCount	HarvestCostRecordCount	OpcostDropped
☐	BM	001	0%	4127	1177	
☐	BM	002	0%	4809	1219	
☐	BM	003	0%	3273	974	
☐	BM	004	0%	3846	1023	
☐	BM	009	0%	4986	1136	
☐	BM	010	0%	5821	1193	
☐	BM	011	0%	3942	952	
☐	BM	012	0%	4452	1006	
☐	BM	013	0%	6589	1370	
☐	BM	014	0%	5817	1180	
☐	BM	015	0%	6616	1527	
☐	BM	016	0%	1710	529	

Figure 5.21 – The Run tab dialog allows selection of the variant/Rxpackage combinations to be processed and provides a “Run” button to initiate processing.

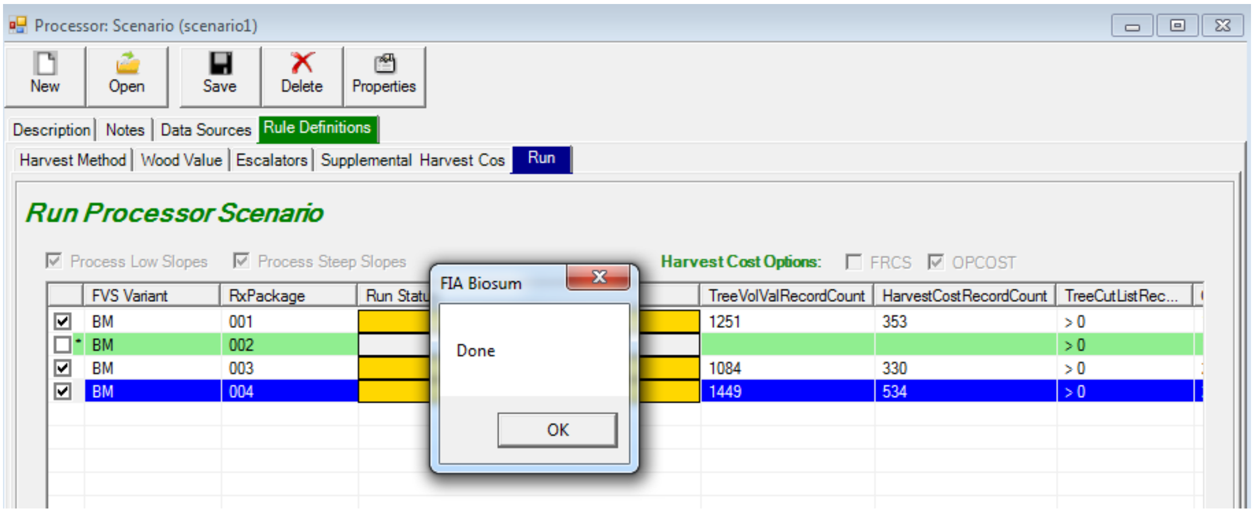
1. Check the box next to any variant/rxpackage combinations to be included in the processor simulation. Silvicultural sequences (RxPackages) that do not involve mechanical thinning, for example a grow-only sequence, will not appear on the list, and cannot be run through Processor because there are no harvesting costs or wood volume or biomass to calculate. Stands can be processed one at a time, by package, by variant, or all stands in one run.

2. Click **<Run>** to start Processor. Depending on computer speed and data size, this can take from several minutes to over an hour.
3. When complete, a "Done" OK dialog will display (it may display underneath other windows on the desktop). Click OK to confirm this. All checked stand package combinations should have a 100% bar displayed for run status. If an error is displayed (usually at the beginning of the run attempt) indicating that there were "no trees in the cut list", this may indicate a number of potential problems—to find out which one, check the log file written to the user/temp folder. Sometimes, this error generates due to an incomplete GIS workflow (with yarding distance not yet specified in master.plot).
4. the data generated by Processor will be stored in processor/[scenario_name]/scenario_results.mdb in two tables:
    - tree_vol_val_by_species_diam_groups, which contains, for each stand, silvicultural sequence, and simulation cycle, the volume, mass and value of collected chip (non-merchantable) and merchantable wood and the volume and mass of cut and scattered (brush-cut) wood, per acre, and placeholder—whether a no cost record is inserted for cycle 1 due to no activity in that cycle (necessitated by BioSum's internal architecture);
    - harvest_costs, which contains, for each stand, silvicultural sequence and simulation cycle, (1) the complete on-site cost in dollars per acre for the sequence *incurred in that cycle*, inclusive of harvest, brush-cutting, chipping and move-in costs, as well as any supplemental costs such as surface fuel abatement via fire or mastication, but *excluding* transportation from forest to processing facilities, (2) the harvest cost in dollars per acre, inclusive of harvest, brush-cutting, chipping and move-in costs, (3) chipping costs broken out separately, (4) equipment move-in costs broken out separately and some housekeeping flags such as (5) placeholder—whether a no cost record is inserted for cycle 1 due to no activity in that cycle (necessitated by BioSum's internal architecture), (6) override—whether harvest specifications used as input to OpCost had to be modified for

parameters to be within valid ranges for the equations embedded in that model, and (7) any warning messages generated during OpCost operation. Other columns are deprecated (obsolete) as of OpCost 10.

- Intermediate data used to produce the harvest_costs table can be found in the OpCost folder, which contains one database per variant-silvicultural sequence combination, named by OpCost version used to calculate harvest costs, variant, sequence, prescriptions, and date and time stamp (so that harvest cost results from sequences processed earlier are preserved in the event these sequences are re-processed). These databases, which can be useful for debugging/quality assurance and/or for evaluating alternative harvest systems worth considering, contain 2 or 3 tables each: an opcost_input table, with each row containing cut list metrics calculated by BioSum for a single stand and year of entry, an opcost_output table, containing estimated harvest costs and, if OpCost generated errors, then an opcost_errors table, containing the input data for stand-year combinations that did not successfully process in OpCost.

5. If additional variant/sequence combinations need to be processed, they can be selected in this form, and processing re-initiated. Any of the processor tabs can also be accessed, to inspect or edit inputs. When done with processor operations, click **<Close>** to return to BioSum Manager**.**



Done

	FVS Variant	RxPackage	Run Statu	TreeVolValRecordCount	HarvestCostRecordCount	TreeCutListRec...
✓	BM	001		1251	353	> 0
☐	BM	002				> 0
✓	BM	003		1084	330	> 0
✓	BM	004		1449	534	> 0

Figure 5.22 – Processor scenario successfully run with three variant/rxPackage combination.

Running OpCost in Stand-alone mode (Advanced Topic)

The OpCost R script was designed to be called by BioSum as a command window process spawned from BioSum that repeatedly runs R, once for each combination of variant and silvicultural sequence, supplying this script and the appropriate input/output database file name as parameters. However, as of version 10.1.5, OpCost can also be run in stand-alone mode from within the R or Rstudio environment by loading the OpCost script and editing to specify file names, with full paths for:

5. the reference database containing harvest system and harvest machine parameters,
6. the database containing OpCost inputs,
7. the database to which OpCost outputs should be written (can be the same as #2),
8. the folder to be used for writing graphics output, if that option is enabled.

Stand-alone OpCost has many potential uses, including, for example:

- Quality Assurance- are the cost estimates reasonable?
- Experimenting with the cost implications of different harvest systems;
- Building graphical summaries of the distribution of machine times, by equation and tree size class;
- Generating harvest cost estimates for purposes unrelated to analysis within the full BioSum framework;
- Building a "testbed" input file to compare and contrast the effects of tree size, tree count, slope and yarding distance on operations cost under different harvest systems.

Open the OpCost R file (which may follow the naming convention OpCost_10_1_5.R or just 10_1_5.R) in R or Rstudio; then edit the code block containing file and path information—this can be found between lines 37 and 49 – to point to the opcost_input database, output database, opcost_ref database and graphic output folder. Make sure that all of these files and folders already exist on your computer (if writing output to a different database than the one that contains the OpCost input, you will need to not only specify a database name but also create an empty Access .accdb file). If using the graphics system and wanting to customize that output, it may be desirable to tweak graph features such as axis limits and graphics file type to save to.

As OpCost is distributed with BioSum, the graphics generating code section is commented out (with a "#" in position 1 of every line) because we do **not** want to run that code when running BioSum (it would add greatly to the time and disk space usage, and in batch mode, would overwrite the most recently generated graphics files for each variant/silvicultural sequence, in turn).

To include graphics output when running OpCost in standalone mode, just select/highlight with mouse (in Rstudio) lines 587-697. Then press ctrl-shfit-C and to toggle the comment character (#) off so that this code will run. This enables the graphics module that generates box and whisker plots for all harvest machines for the analyst specified harvest system for all modeled stands for each entry as well as for all harvest machines across all harvest systems that OpCost supports. To disable graphics generation, select the same lines and press ctrl-shfit-C again and it will restore the # character at the beginning of each line, turning all these lines into unprocessed comments.

Customizing Harvest Cost Estimation (Advanced topic)

As described earlier, OpCost is a cost-estimation framework that includes a “starter set” of parameters primarily describing 11 harvest systems by the logging machines that make up a system, quantifying the expected productive machine hours required by each machine in a harvest system to complete the work on one acre, and calculating the cost of completing that work on one acre. Most users will not need or want to modify OpCost. However, some will. This section provides an overview of how OpCost works and how such customization can be undertaken.

The redesign of OpCost in 2018 that led to a code structure that moved virtually all equations, parameters and assumptions about the machines that comprise a harvest system out of the OpCost R code and into Access (or .CSV) data tables. This exposes virtually all assumptions and calculations into an easily readable form where they can be inspected, verified and revised if needed, and provides access to the cost per machine hour parameters so that local or more current values can be entered if desired. It also opens the potential to add additional machine equations, machine types and/or harvest systems to OpCost by editing the contents of tables in the opcost_ref.accdb file that BioSum’s installation places in “C:\Program Files (x86)\FIA PNW Portland Forestry Sciences Lab\FIA Biosum 5.8.8\db\”.

OpCost’s four parameter tables describe harvest systems as to equipment types used (opcost_harvestsystem_ref); identify particular equations to associate with each of those equipment types (opcost_harvestequation_ref); contain the algebraic representation of those equations, the units of the calculated results, a conversion equation for translating the results to productive machine hours (PMH) per acre, and a limit statement specifying any boundary conditions that govern application of the equation— for example, if the data on which the equation was based were collected only for harvests of trees less than a specified diameter (opcost_equation_ref); and applicable costs per machine hour (opcost_cost_ref). It is critical to remember that all data, including names of variables, machines and harvest systems, must be absolutely consistent with regard to spelling, capitalization and punctuation including spaces (if any) in opcost_ref and in the OpCost input database tables as R is a case-sensitive language.

The last of these, opcost_ref, is the most straightforward to modify (Figure XX). The DefaultCPH value of $210.10 for the FellerBuncher in this example, is the cost per machine hour that will be applied to machine time estimated by OpCost. The other, state-specific values for CPH were considered applicable for those states circa 2016 and can be copied over to the DefaultCPH column for a more localized estimate. These state specific columns are not processed by OpCost so can be used to warehouse values for any state or region; moreover, their column names can be edited as needed (e.g., to CaliforniaCPH) since they are only used to store data that the analyst will ultimately need to copy over to the DefaultCPH column for the data to take effect in an OpCost simulation. The MoveInCostMultiplier is an algebraic expression, applied to DefaultCPH for the machine, using variables defined in OpCost, to represent the time required for the machine to be unloaded and setup at the operations site. The MoveInCostLowBoy is also an algebraic expression for the cost of transporting logging equipment to the site where treatments will be applied. For this FellerBuncher, a move-in cost will be calculated as 0.6533 hrs times the unit cost of 210.10 dollars/hr divided by the assumed area, in acres, treated at one location (yielding $/ac) **plus** the move-in time (hrs) times 100 $/hr divided by assumed area, in acres, giving a second term in $/ac. If the equipment to perform the operations are transported from an hour away (Move_In_Hours=1) and a harvest area of 80 acres is assumed, then the move-in costs for the feller-buncher calculated by OpCost will be ((210.10 * 0.6533) + 100)/80 = $2.96/ac. The 0.6533 hours is approximately 39 minutes. For some equipment, such as a skyline system that needs to be positioned, anchored and tested before operations can begin, it may be appropriate to specify a longer setup time for calculating this MoveInCostMultiplier.

Machine	FellerBuncher
DefaultCPH	210.1
IdahoCPH	210.1
WashingtonCPH	190.1
OregonCPH	175.1
MoveInCostMultiplier	0.6533/Harvest_area_assumed_acres
MoveInCostLowBoy	(Move_In_Hours*100)/Harvest_area_assumed_acres

Figure XX. Data from one row of opcost_cost_ref

Modification of other tables should not be undertaken casually, or without expectation of allowing ample time for empirical testing and quality assurance of the changes made. While some analysts will no-doubt imagine other modifications that we have not yet considered, this documentation covers the three that we view as the most likely modifications analysts might attempt:

1. Modifying (or adding or removing) equations for particular machine types
2. Adding a new machine type
3. Adding a new harvest system

Modifying equations

FRCS contained a comparatively large number of equations per machine type. Sometimes the equations differed as to the scale of the machine (e.g., different equations for small or large skidders), but usually, the equations were for the same general kind of machine operated in different contexts (e.g., of slope and yarding distance) and/or tree harvesting burden (e.g., small vs large diameter trees, primarily hardwoods vs primarily conifers, high or low trees per acre harvested, etc.). By estimating with different equations and calculating the mean estimated time required for that machine's work on an acre, it was thought that a more robust estimate would result. However, in reality, there is likely one equation that best represents a particular situation (though which equation has that property is often unknown), and in essence diluting the most valid results of that one best equation by averaging them in with the results from several other equations that are less well fit to the situation, may not lead to greater accuracy.

The current version of OpCost has far fewer equations per machine than did FRCS, in most cases.

that we envision at this time, and the ones that we will address via this documentation, are

opcost_harvestsystem_ref

opcost_harvestsystem_ref		
HarvestingSystem	**Activity**	**Machine**
Ground-Based CTL	Chipper	Chipper
Ground-Based CTL	Forwarder	Forwarder
Ground-Based CTL	MechBrushCut	Harvester_BC
Ground-Based CTL	Harvester	Harvester
Ground-Based CTL	Chainsaw	Chainsaw

opcost_harvestequation_ref									
HM#	**Tree Size**	**Method**	**BC**	**Cut**	**Move**	**Chip**	**Load**	**Extra**	**Comment**
2	BC	Ground-Based CTL	BC_01H						
2	CT	Ground-Based CTL		HAR_04C, HAR_05C, HAR_08C	FOR_06	CHIP_02			
2	SL	Ground-Based CTL		HAR_04S, HAR_05S, HAR_08S	FOR_06	CHIP_01			
2	LL	Ground-Based CTL		SAW_03L	FOR_06	CHIP_01			

opcost_equation_ref

EquationID	Name	Machine	Size	Equation	Units
HAR_04C	jirousek	Harvester	Chip	60.711 * ((twitchVol_ct)^0.6545)	m^3/PMH
HAR_04S	jirousek	Harvester	Small	60.711 * ((twitchVol_sl)^0.6545)	m^3/PMH
HAR_05C	karha	Harvester	Chip	0.288 + (0.1004 * ((twitchVol_ct)/0.001)) + (-0.00008*((twitchVol_ct)/0.001)^2)	m^3/PMH
HAR_05S	karha	Harvester	Small	0.288 + (0.1004 * ((twitchVol_sl)/0.001)) + (-0.00008*((twitchVol_sl)/0.001)^2)	m^3/PMH

HAR_08C	klepac	Harvester	Chip	24.796 + 0.331419 * dbh_ct ^ 2	sec/tree
HAR_08S	klepac	Harvester	Small	24.796 + 0.331419 * QMD_SL ^ 2	sec/tree

EquationID	HoursPerAcreConversion	LimitStatement
HAR_04C	totalVol_ct/result	twitchVol_ct<1.1
HAR_04S	totalVol_sl/result	twitchVol_sl<1.1
HAR_05C	totalVol_ct/result	twitchVol_ct<1.25
HAR_05S	totalVol_sl/result	twitchVol_sl<1.25
HAR_08C	(Chip.tree.per.acre * result)/60/60	dbh_ct<21
HAR_08S	(Small.log.trees.per.acre * result)/60/60	QMD_SL<21

EquationID	OldTimePerAcreConversions	OldEqu
HAR_04C	ifelse(twitchVolM(m)<1.4, (totalVolM(m)/jirousekHar(m)), NA)	60.711*(twitchVolM(m)^0.6545)
HAR_04S	ifelse(twitchVolM(m)<1.4, (totalVolM(m)/jirousekHar(m)), NA)	60.711*(twitchVolM(m)^0.6545)
HAR_05C	ifelse(twitchVolM(m)<.0, (totalVol(m)*0.0283168)/karhaHar(m), NA)	0.288+(0.1004*(m$"Percent.Slope"/0.001))+(-0.00008*(tw
HAR_05S	ifelse(twitchVolM(m)<.0, (totalVol(m)*0.0283168)/karhaHar(m), NA)	0.288+(0.1004*(m$"Percent.Slope"/0.001))+(-0.00008*(tw
HAR_08C	((klepacHar(m)*treesRemoved(m))/60)/60	24.796+0.331419*(dbh(m)^2)
HAR_08S	((klepacHar(m)*treesRemoved(m))/60)/60	24.796+0.331419*(dbh(m)^2)

Outline
Designed to be controlled by ref tables in opcost_ref.accdb
Explain what is in each of the 4 tables opcost_harvestsytem_ref, opcost_equation_ref, opcost_harvestequation_ref and opcost_cost_ref and how they are used.
Explain how to add a harvest system
Explain how to drop an existing harvest equation from a system

Explain how to add a harvest equation and add it to a system

Literature Cited

Fight, R.D., B.R. Hartsough and P. Noordijk. 2006. Users guide for FRCS: fuel reduction cost simulator software. Gen. Tech. Rep. PNW-GTR-668. Portland, OR: U.S. Department of Agriculture, Forest Service, Pacific Northwest Research Station. 23 p

Chapter 6. Treatment Optimizer

Overview

The Treatment Optimizer module builds on databases generated by successfully completing all the preceding modules (Database, FVS, Processor), enabling analysis and exploration of one or more optimization scenarios. A complete scenario contains assumptions about the land base to be considered, processing sites to which merchantable wood and residues can be delivered, per hour costs for wood transportation and, by reference to a processor scenario, simulation cycle specific per acre costs, and wood production and revenues associated with each modeled treatment. It also contains decision rules, crafted by the analyst while engaging in treatment optimizer, that determine which treatments are effective for each stand at each time, and when more than one are effective, which is preferred. Economic considerations can be integrated here, for example to filter out stand-prescription combinations deemed too costly, and to include cost in the consideration of what a preferred treatment looks like.

Module Components

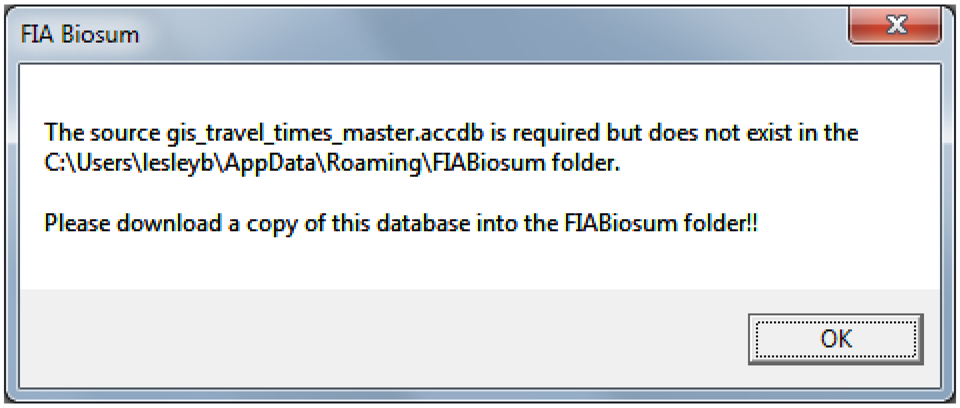
The button that activates the Treatment Optimizer module can be found in the module-selector on the left-side of the FIA BioSum Manager desktop (it is labeled “Treatment Optimizer”).

Selecting it opens a 3-button panel that accesses three tasks: (1) **Define Calculated Variables**, (2) **Load GIS Data**, (3) **Optimization Scenario**, and (4) **Export to SQLITE**. Unless Calculated Variables are used, the current version of Treatment Optimizer supports analysis for treatment effects, production, costs and revenues associated with cycle 1 only.

Load GIS Data

Treatment Optimizer calculates haul costs as an element of an optimization scenario. The processing_site and travel_time tables in the \gis\db\gis_travel_times.accdb must be populated prior to calculating haul costs. If the project plots are in one of eight western states (CA, CO, ID, MT, OR, UT, WA, and WY), BioSum provides a master database and automated loading process that may be used to populate these tables.

Before loading the gis_travel_times.accdb, the gis_travel_times_master.accdb needs to be in your AppData directory. The path to this folder varies by computer. The easiest way to locate the correct folder is to click the <**Load GIS Data**> button. If BioSum can't find gis_travel_times_master.accdb, the error message provides the path that it needs to be copied to:

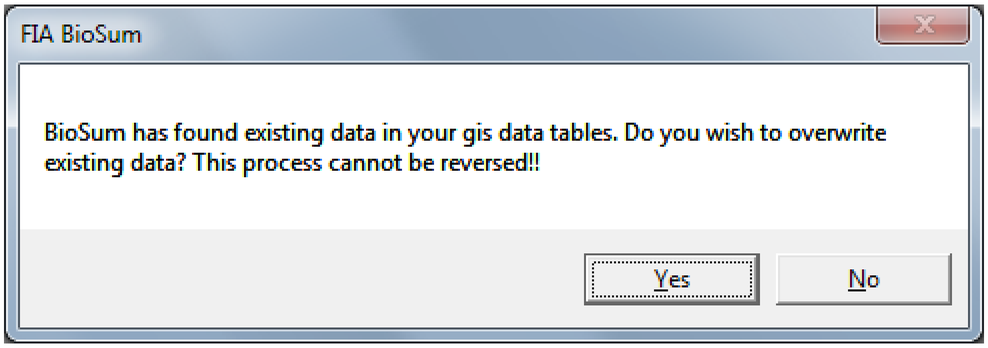


The source gis_travel_times_master.accdb is required but does not exist in the C:\Users\lesleyb\AppData\Roaming\FIABiosum folder.

Please download a copy of this database into the FIABiosum folder!!

gis_travel_times_master.accdb may be in your C:\FIA\setup folder as it is distributed as part of the BioSum full installation package. If you can't find the database there or want to make sure you have the most current copy, the database can also be downloaded from http://biosum.info. Note that the database is large (approximately 500,000 kb).

When the gis_travel_times_master.accdb is in place, click the <**Load GIS Data**> button to start the load process. If BioSum detects existing data in the processing_site or travel_time tables, it will ask for confirmation that the data should be reloaded:



BioSum has found existing data in your gis data tables. Do you wish to overwrite existing data? This process cannot be reversed!!

If you allow existing data to be overwritten, BioSum provides the option of backing up the existing data before proceeding.

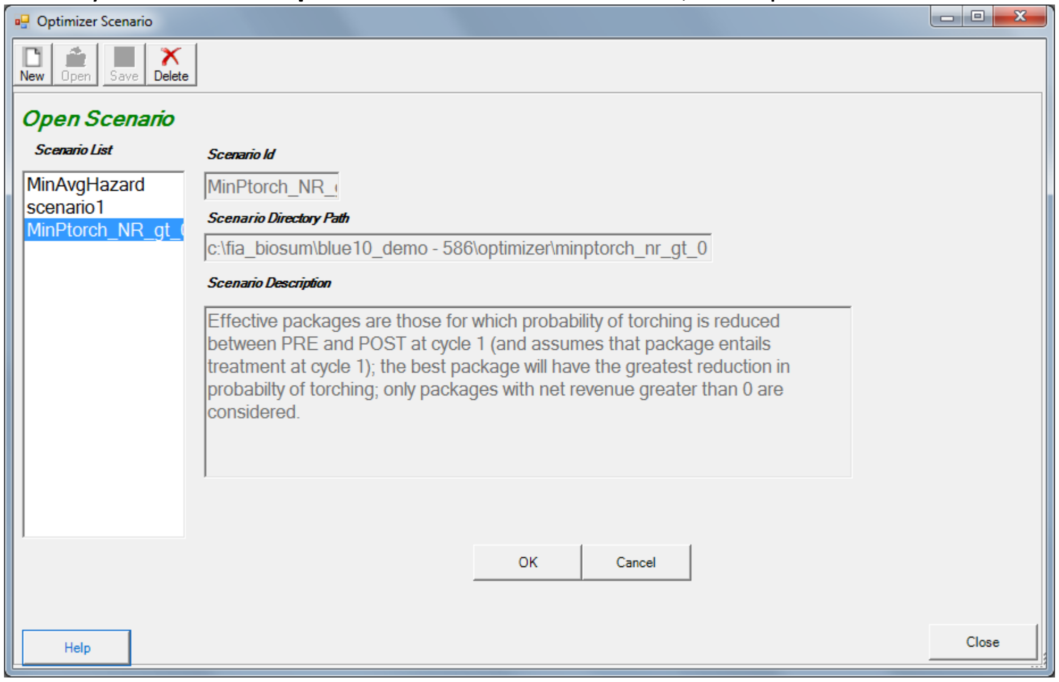
The automated load process may take a few minutes depending on the number of plots in the project. BioSum starts by selecting records from the master travel_time table associated with all plots in the project. Then BioSum creates processing_site records for every processing_site listed in the project travel_time table. The processing_site table contains some fields that aren't displayed in the BioSum interface and may serve as a resource, if supplementary information about processing_sites is helpful.

A progress indicator will appear in the middle of the BioSum window for the duration of the process. Detailed process steps will appear in the lower right-hand corner of the BioSum Window. When the load has successfully completed, BioSum will show this message:

"If you updated existing GIS data, verify the selected sites in Treatment Optimizer. GIS data successfully loaded!"

Optimization Scenario

When you click the <**Optimization Scenario**> button, the Open Scenario screen will appear:



MinAvgHazard
scenario1
MinPtorch_NR_gt_(

MinPtorch_NR_(

c:\fia_biosum\blue10_demo - 586\optimizer\minptorch_nr_gt_0

Effective packages are those for which probability of torching is reduced between PRE and POST at cycle 1 (and assumes that package entails treatment at cycle 1); the best package will have the greatest reduction in probabilty of torching; only packages with net revenue greater than 0 are considered.

Figure 6.5: Open Scenario window

The Open Scenario form has the following fields:

1. **Scenario List**: This is the list of all the optimization scenarios available in the project. When a scenario in this list is selected (highlighted), the other fields on the screen will update to reflect information about the selected scenario.
2. **Scenario Id**: The optimization scenario id currently selected from the list
3. **Scenario Directory Path**: The computer file system path to the scenario currently selected from the list
4. **Scenario Description**: An optional free form description of the scenario currently selected from the list

There are two buttons enabled on the toolbar in the upper left corner of this window. They are:

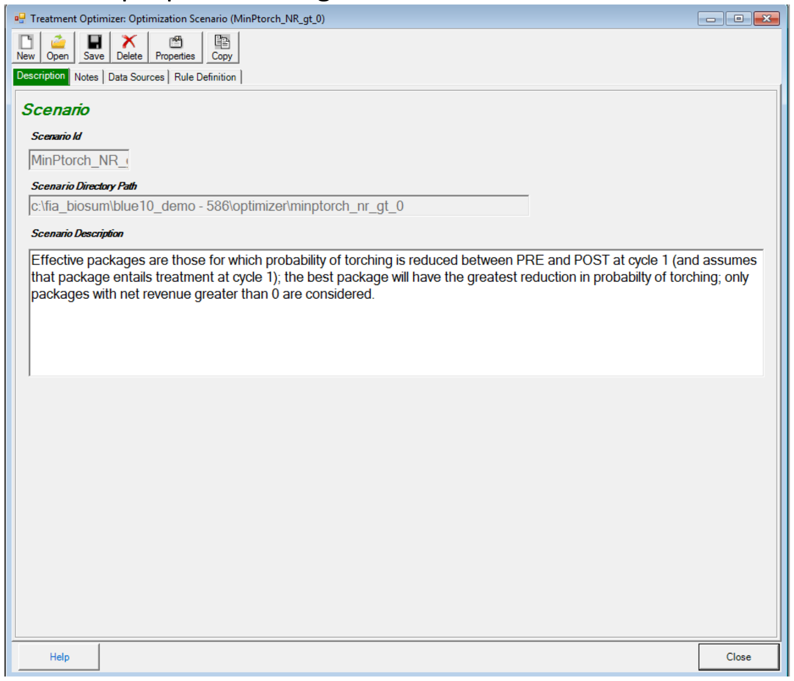
1. **<New>**: Use this button to create a new optimization scenario. The **New Scenario** window that will open has places for you to enter the **Scenario Id** and **Description**. The scenario id is required and the description is optional, but strongly recommended. Although the suggested scenario id will start with scenario…, it is recommended to choose a more descriptive scenario id so that it will be useful when trying to identify and select a particular scenario in the future.
2. **<Delete>**: Use this button to delete the selected scenario. All configuration settings and databases associated with the selected scenario will be deleted. **Caution**: this action cannot be undone.

There are two buttons at the base of this window. They are:

1. **<OK>**: This button opens the selected optimization scenario and displays the settings it contains, ready for viewing or editing.
2. **<Cancel>**: Closes the Open Scenario window

Treatment Optimizer Scenario properties

After clicking the <**OK**> button to open an optimization scenario, you can review and update the scenario properties using a combination of buttons and tabs that are available on this form.



Scenario

Scenario Id

MinPtorch_NR_

Scenario Directory Path

c:\fia_biosum\blue10_demo - 586\optimizer\minptorch_nr_gt_0

Scenario Description

Effective packages are those for which probability of torching is reduced between PRE and POST at cycle 1 (and assumes that package entails treatment at cycle 1); the best package will have the greatest reduction in probabilty of torching; only packages with net revenue greater than 0 are considered.

Figure 6.6: First page (Description tab) of optimization scenario properties

The toolbar in the upper left corner of this form has 6 enabled buttons:

1. **<New>**: Opens the new scenario screen to create a new optimization scenario, but see note for <**Open**> button below, as to advisability. It is almost always better to close open scenarios before initializing a new one.
2. **<Open>**: Allows you to select and open an additional scenario. Having multiple optimization scenarios open at the same time can be confusing and is not advised except when there is a desire to inspect and compare two different scenarios, screen by screen.
3. **<Save>**: Saves the configuration settings for the optimization scenario that is open. This button saves all of the settings for the entire scenario regardless of which tab is currently active.
4. **<Delete>**: Deletes the scenario that is open in this window and returns to the Open Scenario screen. All configuration settings and databases associated with the deleted scenario will be removed. **Caution:** this action cannot be undone

5. **<Properties>**: Displays the configuration settings for the current scenario in plain text format, allowing for the review of all configuration settings on a single page. This button is also useful for documenting a scenario, as you can copy and paste the settings into an external text document that can be saved and used as a log or audit trail.
6. **<Copy>**: Copies the settings from a selected scenario to the current scenario. **Caution**: copying a scenario overwrites any existing scenario settings and this action cannot be undone.

The Scenario Description can be updated on this form, if desired. Clicking the **<Close>** button closes the optimization scenario properties window. FIA Biosum will prompt you to save unsaved changes, if any, when this button is clicked.
There are three other tabs on this screen, in addition to **Description: Notes, Data Sources** and **Rule Definitions**.

1. **Notes**: This space accommodates detailed notes about a scenario. Use this tab if the Description field is not large enough to accommodate your descriptive text, or if there is a need to otherwise supplement the information contained in the description.
2. **Data Sources**: This tab displays pointers (file paths, file names, table names, status, records counts, etc.) to all types of source data used in the optimization scenario. Any of these data sources can be copied to another table or database and the pointer updated to point to the copy instead of the original data source. These copies can then be updated or customized to allow further customization of scenarios. For example, an alternate version of travel times could be generated, reflecting the construction of a new road, and could allow the analysis to account for faster travel times from stands to processing facilities.

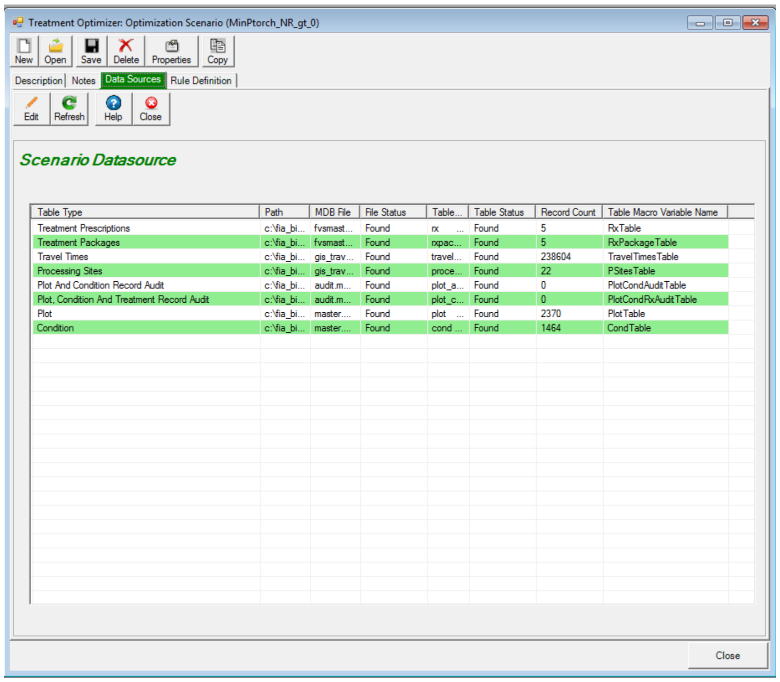


Table Type	Path	MDB File	File Status	Table...	Table Status	Record Count	Table Macro Variable Name
Treatment Prescriptions	c:\fia_bi...	fvsmast...	Found	rx ...	Found	5	RxTable
Treatment Packages	c:\fia_bi...	fvsmast...	Found	rxpac...	Found	5	RxPackageTable
Travel Times	c:\fia_bi...	gis_trav...	Found	travel...	Found	238604	TravelTimesTable
Processing Sites	c:\fia_bi...	gis_trav...	Found	proce...	Found	22	PSitesTable
Plot And Condition Record Audit	c:\fia_bi...	audit.m...	Found	plot_a...	Found	0	PlotCondAuditTable
Plot, Condition And Treatment Record Audit	c:\fia_bi...	audit.m...	Found	plot_c...	Found	0	PlotCondRxAuditTable
Plot	c:\fia_bi...	master....	Found	plot ...	Found	2370	PlotTable
Condition	c:\fia_bi...	master....	Found	cond ...	Found	1464	CondTable

Figure 6.7: The Data Sources tab lists all of the optimization scenario data sources

To edit the pointer to the data source, and to make copies of data sources, select a table type and click **<Edit>**. The **Data Source Edit** window will appear. Here you can move, copy, or rename existing Access DB files and tables, and reset links for any table type.

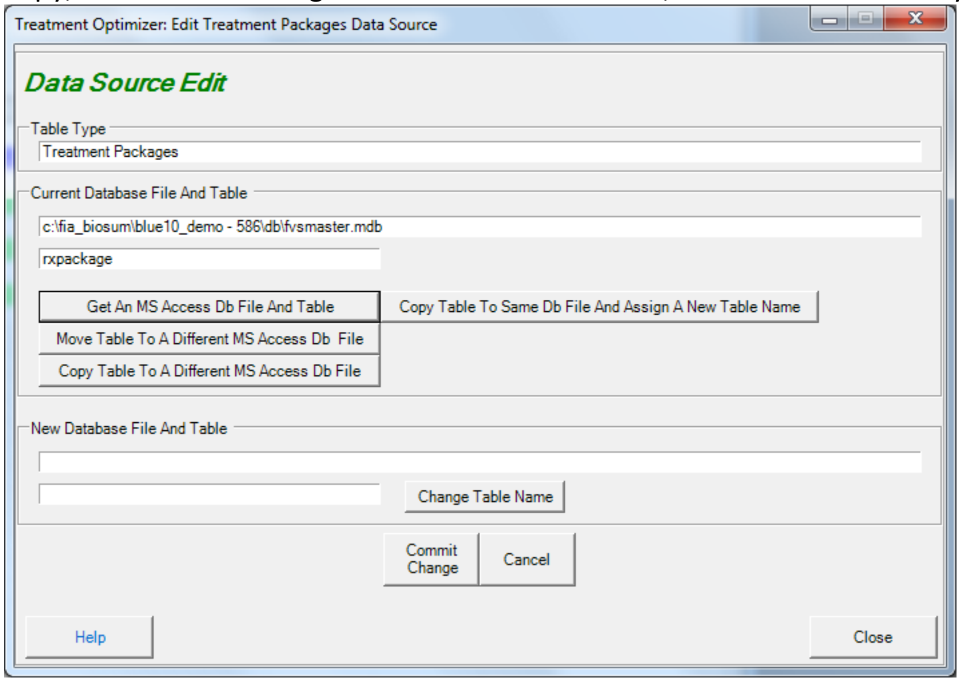

Treatment Optimizer: Edit Treatment Packages Data Source
Data Source Edit
Get An MS Access Db File And Table
Copy Table To Same Db File And Assign A New Table Name
Move Table To A Different MS Access Db File
Copy Table To A Different MS Access Db File
Change Table Name
Commit Change
Cancel
Help
Close

Table Type: Treatment Packages

Current Database File And Table: c:\fia_biosum\blue10_demo - 586\db\fvsmaster.mdb; rxpackage

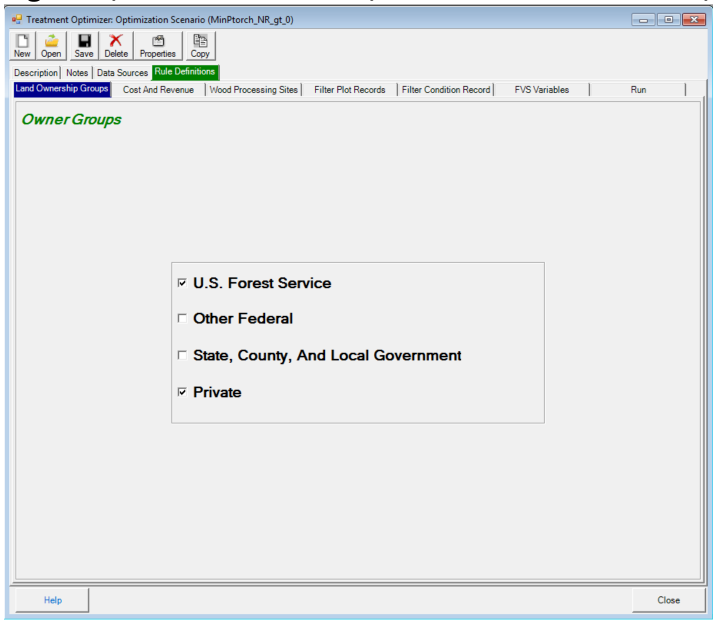
New Database File And Table:

Figure 6.8: The Data Source Edit window

Caution: Making changes to data sources is an advanced capability, not to be undertaken lightly or by FIA Biosum beginners, and has the potential to be confusing or produce unintended consequences.

Rule Definitions

Most of the properties related to a case scenario are configured from the Rule Definitions tab. This tab has 7 child tabs. They are: **Land Ownership Groups**, **Cost and Revenue, Wood Processing Sites, Filter Plot Records, Filter Condition Records, FVS Variables** and **Run**.



- [✓] U.S. Forest Service
- [ ] Other Federal
- [ ] State, County, And Local Government
- [✓] Private

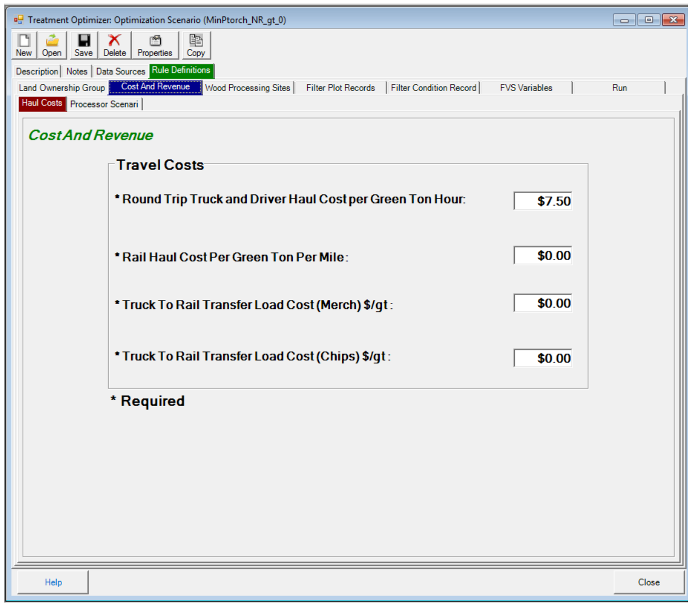
Figure 6.9: The first child tab on the Rule Definitions tab is Land Ownership Groups

1. **Land Ownership Groups**: Allows you to filter your scenario input plots according to landowner group (for example, USFS, private). A checked box tells FIA Biosum to include stands from the corresponding land owner group. One, some, or all owner group boxes can be checked depending on which ownerships are assumed to be managing with the treatments specified in this project.

   It is possible to query final outputs to determine how much of the area treated falls belongs to each owner group, and how much woody biomass, by size class, was generated by each owner group, but these kinds of answers are inextricably linked to the land base considered at the time of the analysis. For example, a bioenergy facility at a given site may only be economically feasible (in terms of longevity of feedstock supply) if all treatable lands are treated. By applying the landownership constraints prior to running the analysis, it is easy to determine scenarios that will make it more obvious when there is not enough supply to warrant a bioenergy facility (e.g., if only Private was checked).
2. **Cost and Revenue**: This tab has 2 child tabs. They are: **Haul Costs** and **Processor Scenario**. The fields on the **Haul Costs** tab relate to costs of hauling woody material (e.g., merchantable wood and woody biomass destined for bioenergy feedstock) from the forest to processing sites. They are:
   a. **Round Trip Truck and Driver Haul Cost per Green Ton Hour:** should be entered in dollars and is used, in combination with the program-calculated round-trip

travel times between plots and processing sites, to calculate the costs of hauling woody material (chips or logs) to processing sites via the road network.

b. **Rail Haul Cost per Green Ton Mile**: should contain the per-mile cost, in dollars, of moving one ton of material one mile along a rail network once it is already on the rail network.
c. **Truck to Rail Transfer Load Cost (Merch) $/gt**: specifies the costs of inter-modal transition from truck to rail for merchantable wood. If no rail is specified in the scenario, this field can be set to 0.
d. **Truck to Rail Transfer Load Cost (Chip) $/gt**: specifies the costs of inter-modal transition from truck to rail for chip wood. If no rail is specified in the scenario, this field can be set to 0.



Cost And Revenue

Travel Costs

* Round Trip Truck and Driver Haul Cost per Green Ton Hour:	$7.50
* Rail Haul Cost Per Green Ton Per Mile:	$0.00
* Truck To Rail Transfer Load Cost (Merch) $/gt:	$0.00
* Truck To Rail Transfer Load Cost (Chips) $/gt:	$0.00

* Required

Figure 6.10: Haul Costs child tab on the Costs and Revenue tab

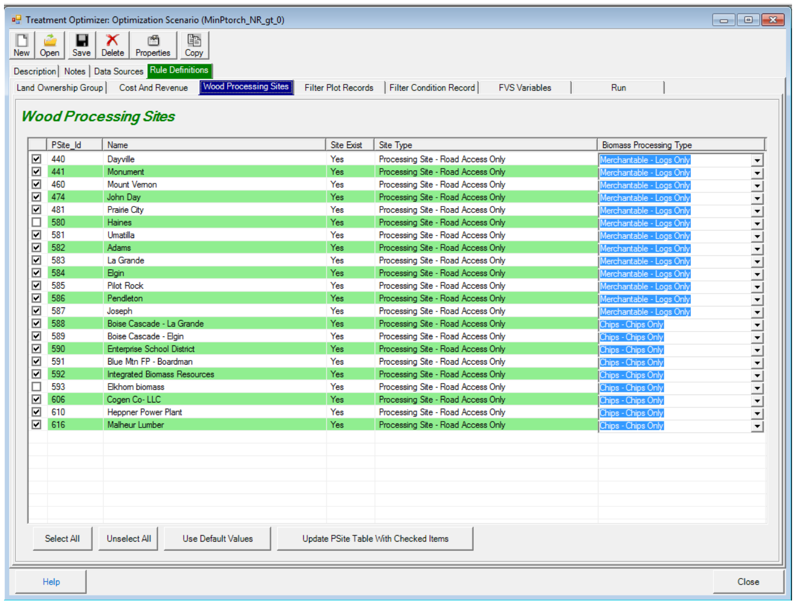
e. Each optimization scenario MUST be associated with a processor scenario. Use the **Processor Scenario** tab to select the processor scenario that should be used as the source for harvested wood volumes, weights, values, and onsite harvest costs.

3. **Wood Processing Sites**: Displays the names and types of processing sites (psites) that are available for consideration in the scenario. Use the checkbox in the left-hand column to include/exclude a psite from the scenario.

   The **Biomass Processing Type** column on the right-hand side of the table allows you to specify if the psite can process: **Merchantable – Logs Only**, **Chips – Chips Only** or **Both – Logs and Chips**.

There are 4 buttons below the **Wood Processing Sites** table that may be used to manipulate data on the table. They are:

a. <**Select All**>: Checks the boxes for all the psites on the current table for inclusion in the current scenario
b. <**Unselect All**>: Unchecks the boxes for all the psites on the current table to exclude them from the current scenario
c. <**Use Default Values**>: There is a master psite configuration table called 'processing_site' associated with each FIA Biosum project. This table is located in \gis\db\gis_travel_times.mdb. Use this button if you have made changes to the **Biomass Processing Type** column for the current scenario but wish to revert to the default psite settings from the project master psite table.
d. <**Update PSite Table With Checked Items**>: Use this button to update the **Biomass Processing Type** field on the project master psite table from the current Wood Processing sites table.

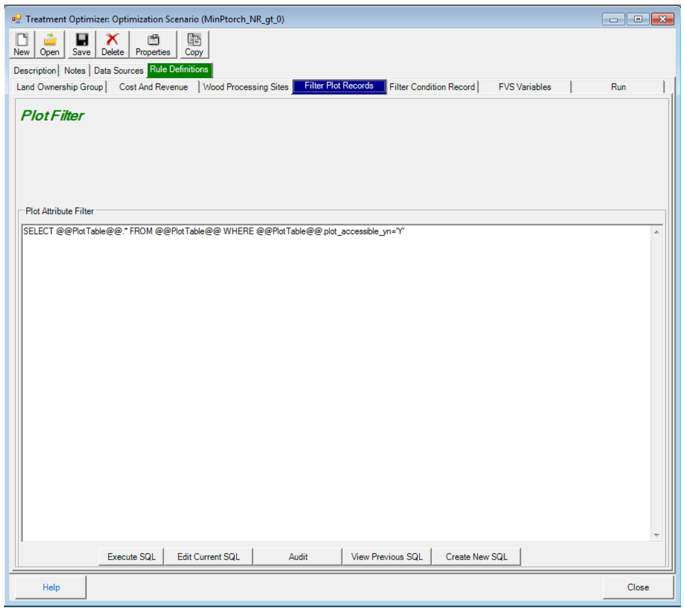


Wood Processing Sites

	PSite_Id	Name	Site Exist	Site Type	Biomass Processing Type
☑	440	Dayville	Yes	Processing Site - Road Access Only	Merchantable - Logs Only
☑	441	Monument	Yes	Processing Site - Road Access Only	Merchantable - Logs Only
☑	460	Mount Vernon	Yes	Processing Site - Road Access Only	Merchantable - Logs Only
☑	474	John Day	Yes	Processing Site - Road Access Only	Merchantable - Logs Only
☑	481	Prairie City	Yes	Processing Site - Road Access Only	Merchantable - Logs Only
☐	580	Haines	Yes	Processing Site - Road Access Only	Merchantable - Logs Only
☑	581	Umatilla	Yes	Processing Site - Road Access Only	Merchantable - Logs Only
☑	582	Adams	Yes	Processing Site - Road Access Only	Merchantable - Logs Only
☑	583	La Grande	Yes	Processing Site - Road Access Only	Merchantable - Logs Only
☑	584	Elgin	Yes	Processing Site - Road Access Only	Merchantable - Logs Only
☑	585	Pilot Rock	Yes	Processing Site - Road Access Only	Merchantable - Logs Only
☑	586	Pendleton	Yes	Processing Site - Road Access Only	Merchantable - Logs Only
☑	587	Joseph	Yes	Processing Site - Road Access Only	Merchantable - Logs Only
☑	588	Boise Cascade - La Grande	Yes	Processing Site - Road Access Only	Chips - Chips Only
☑	589	Boise Cascade - Elgin	Yes	Processing Site - Road Access Only	Chips - Chips Only
☑	590	Enterprise School District	Yes	Processing Site - Road Access Only	Chips - Chips Only
☑	591	Blue Mtn FP - Boardman	Yes	Processing Site - Road Access Only	Chips - Chips Only
☑	592	Integrated Biomass Resources	Yes	Processing Site - Road Access Only	Chips - Chips Only
☐	593	Elkhorn biomass	Yes	Processing Site - Road Access Only	Chips - Chips Only
☑	606	Cogen Co- LLC	Yes	Processing Site - Road Access Only	Chips - Chips Only
☑	610	Heppner Power Plant	Yes	Processing Site - Road Access Only	Chips - Chips Only
☑	616	Malheur Lumber	Yes	Processing Site - Road Access Only	Chips - Chips Only

Figure 6.11: The Wood Processing Sites tab displays details of the psites in a scenario

4. **Filter Plot Records**: Allows you to build a SQL filter to include/exclude plots from the optimization analysis. The plot table structure includes the fields *gis_roadless_yn* and *gis_protected_area_yn*. The values in these two fields are obtained during the GIS process. If it is determined that the plot is in a roadless area then the flag *gis_roadless_yn* is set to 'Y'. Likewise, if a plot is in a protected area then the variable *gis_protected_area_yn* is set to 'Y'. When an optimization scenario is calculated, the field *plot_accessible_yn* value is set by evaluating the values in *gis_roadless_yn* and *gis_protected_area_yn* fields. By default an optimization scenario includes an attribute filter to only include plots where *plot_accessible_yn* = 'Y'.

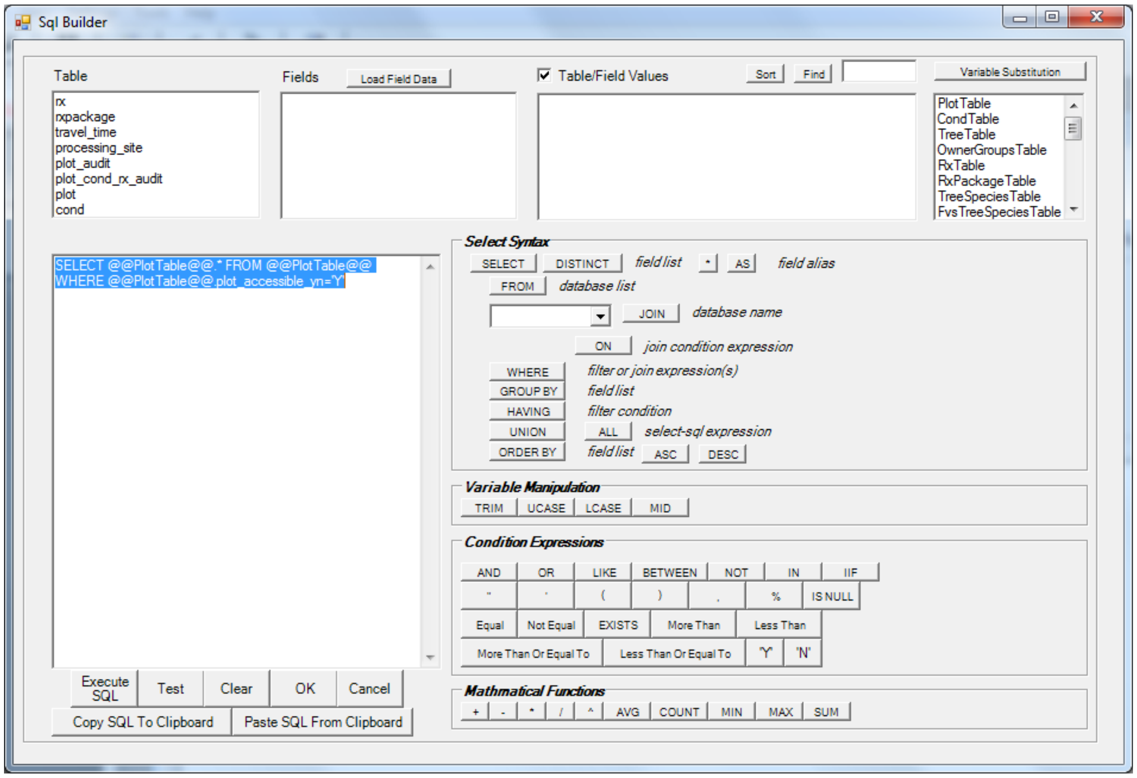


```
SELECT @@PlotTable@@.* FROM @@PlotTable@@ WHERE @@PlotTable@@.plot_accessible_yn='Y'
```

Figure 6.12: Plot attribute filter tab

The **Filter Plot Records** tab has 4 buttons beneath the **Plot Attribute Filter**. They are:

a. **<Execute SQL>**: Click this button to view records selected by the SQL currently in the Plot Attribute filter.
b. **<Edit Current SQL>**: Opens the interactive SQL Builder



```
SELECT @@PlotTable@@.* FROM @@PlotTable@@
WHERE @@PlotTable@@.plot_accessible_yn='Y'
```

Figure 6.13: SQL Builder window for the Plot Attribute Filter

c. **<Audit>**: Tests the plot filter SQL to ensure that it is valid

d. **<View Previous SQL>**: Lists the previous Plot SQL statements by scenario. The current scenario SQL has a current_yn field value of 'Y.' To select the plot SQL, highlight the row and click the **<Select>** button. To delete the item, highlight the row and click the **<Delete>** button. Deleted records are only marked for deletion and can be recalled by pressing the **<Recall>** button. Using a plot filter from a previous scenario is often a good starting point for novice FIA Biosum users.
e. **<Create New SQL>**: Begins creation of a new plot SQL statement by asking you to select the tables that should be included in the filter and adding them to the SQL Builder window that subsequently opens.

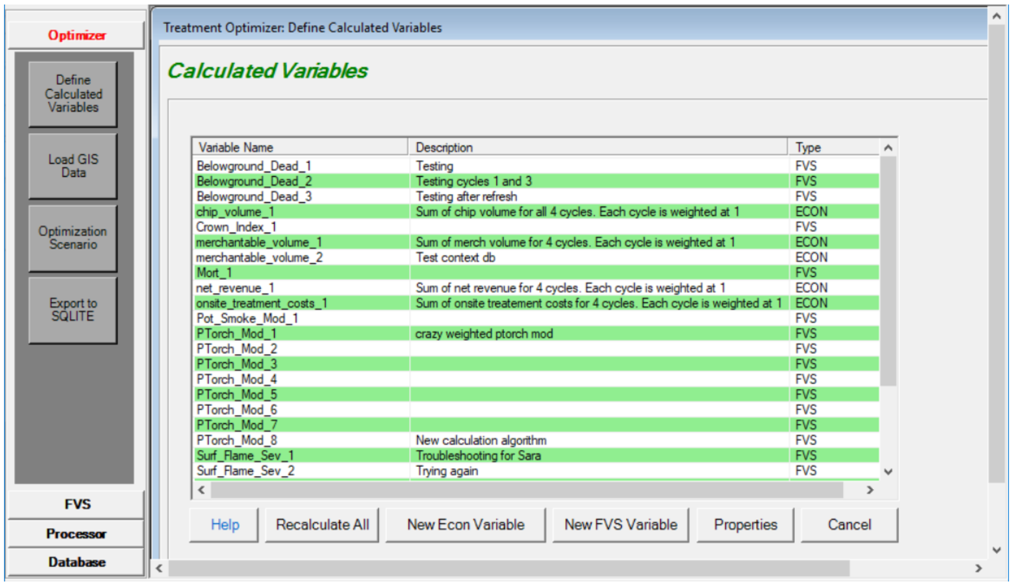
5. **Filter Condition Records**: Allows you to build a SQL filter to include/exclude conditions (stands) from the optimization analysis. See the section above on filtering plot records for details on building/updating a condition filter. The interfaces are almost identical.

   In addition, the Filter Condition Records window provides two fields that specify the **Maximum Yarding Distance Allowed (Feet)**. Conditions exceeding these maximum yarding distances are excluded from optimization scenario processing. There are separate boxes for **Low Slope** and **Steep Slope** conditions. The slope degrees value that determines whether a slope is steep is configured in the processor scenario associated with the current optimization scenario.
   Use the **<Default Values>** to set the **Maximum Yarding Distance Allowed (Feet)** values to the FIA Biosum default. Currently these default values are 2500 feet for both slope categories.

Define Calculated Variables

There are two categories of calculated variables (weighted FVS and weighted economic). Weighted FVS variables are derived from values in the PRE/POST tables that are generated by the FVS module. These tables are located in the /fvs/data/<variant> folder.
Weighted FVS variables can be used in an optimization scenario when defining the elements of an effective treatment, as the optimization variable, or as the first tie-breaker variable. Weighted economic variables can be used in an optimization scenario as the optimization variable or as the first tie-breaker variable. In addition, weighted economic variables derived from net revenue or onsite treatment costs can be used to filter packages in the optimization settings.

1. Click the <**Define Calculated Variables**> button to review the list of calculated variables. This screen lists all of the calculated variables available for use along with their descriptions and category/type.



Variable Name	Description	Type
Belowground_Dead_1	Testing	FVS
Belowground_Dead_2	Testing cycles 1 and 3	FVS
Belowground_Dead_3	Testing after refresh	FVS
chip_volume_1	Sum of chip volume for all 4 cycles. Each cycle is weighted at 1	ECON
Crown_Index_1		FVS
merchantable_volume_1	Sum of merch volume for 4 cycles. Each cycle is weighted at 1	ECON
merchantable_volume_2	Test context db	ECON
Mort_1		FVS
net_revenue_1	Sum of net revenue for 4 cycles. Each cycle is weighted at 1	ECON
onsite_treatment_costs_1	Sum of onsite treatement costs for 4 cycles. Each cycle is weighted at 1	ECON
Pot_Smoke_Mod_1		FVS
PTorch_Mod_1	crazy weighted ptorch mod	FVS
PTorch_Mod_2		FVS
PTorch_Mod_3		FVS
PTorch_Mod_4		FVS
PTorch_Mod_5		FVS
PTorch_Mod_6		FVS
PTorch_Mod_7		FVS
PTorch_Mod_8	New calculation algorithm	FVS
Surf_Flame_Sev_1	Troubleshooting for Sara	FVS
Surf_Flame_Sev_2	Trying again	FVS

Figure 6.1: List of calculated variables

2. From this window, you have the option of (1) viewing the **Help** for this screen (2) **Recalculating All** existing FVS weighted variables (3) configuring a **New Econ Variable**, (4) configuring a **New FVS Variable**, (5) viewing the **Properties** of a variable, or (6) selecting **Cancel** to close the window.

Recalculate All

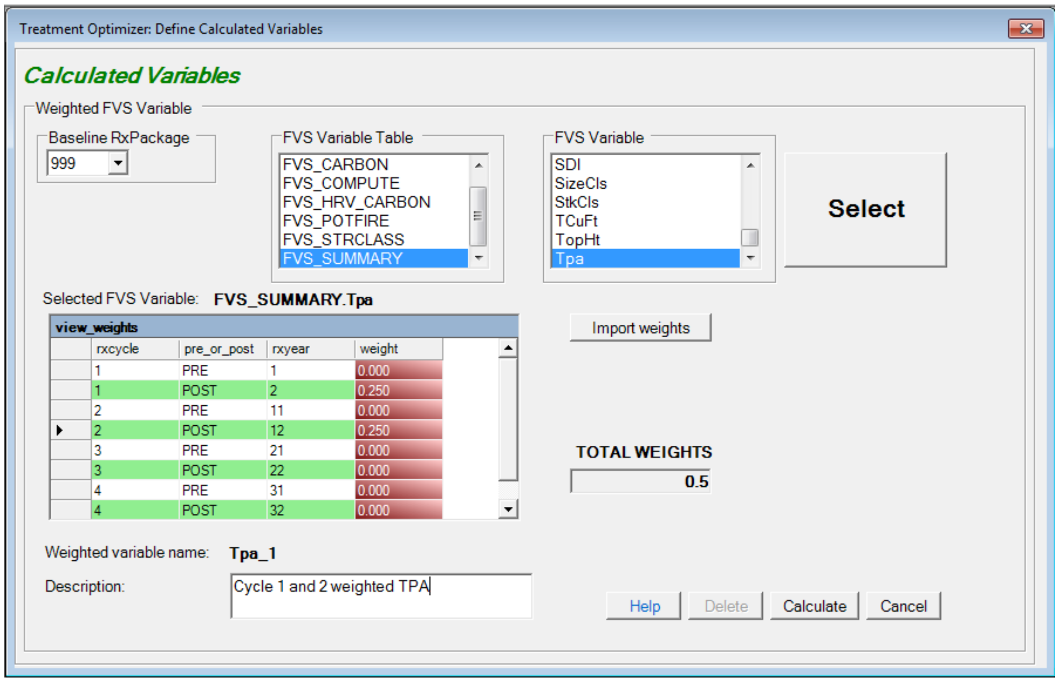
1. FVS-based calculated variables are calculated at the time they are created resulting in faster processing times for Treatment Optimizer scenarios. If the underlying FVS output tables change because FVS is re-run, the contents of the weighted variable tables will be invalid. The most common reason for re-running FVS after working in Treatment Optimizer is to modify the list of prescriptions.

2. Click the **<Recalculate All>** button to commence the recalculation process. Because this process cannot be reversed, you must confirm that you do wish to recalculate. A copy of the existing prepost_fvs_weighted.accdb is saved under the name prepost_fvs_weighted_backup.accdb before the recalculation starts in case of a problem.

3. This process may take several minutes to run depending on the size of the FVS tables and the number of weighted variables.

Configuring a New FVS Variable

1. Click the **<New FVS Variable>** button to configure a new weighted FVS variable.



Baseline RxPackage: 999

FVS Variable Table: FVS_CARBON, FVS_COMPUTE, FVS_HRV_CARBON, FVS_POTFIRE, FVS_STRCLASS, FVS_SUMMARY

FVS Variable: SDI, SizeCls, StkCls, TCuFt, TopHt, Tpa

Selected FVS Variable: **FVS_SUMMARY.Tpa**

view_weights

rxcycle	pre_or_post	rxyear	weight
1	PRE	1	0.000
1	POST	2	0.250
2	PRE	11	0.000
2	POST	12	0.250
3	PRE	21	0.000
3	POST	22	0.000
4	PRE	31	0.000
4	POST	32	0.000

TOTAL WEIGHTS: 0.5

Weighted variable name: **Tpa_1**

Description: Cycle 1 and 2 weighted TPA

Figure 6.2: New weighted FVS variable properties screen

2. The fields on the weighted FVS variable properties screen are:
    a. **Baseline RxPackage**: Without weighted variables, the effectiveness of optimization scenarios is based on the concept of comparing pre and post values for stand attributes for cycle 1; that is, comparing the values before and immediately (1 year) after treatment. With weighted variables, because you will be comparing summary values across multiple cycles/treatments, the first step is to choose a baseline RxPackage. Treatment Optimizer will compare the metrics for each RxPackage to the results from the Baseline package. A common scenario is to compare active treatment packages with a grow-only package to isolate the effects of the treatments. Biosum grow-only packages are customarily assigned the RxPackage number 999, but this is configurable.
    b. **FVS Variable Table**: This is a list of all the FVS Pre/Post tables that are available to be used as the basis for an FVS weighted variable. If you do not see a table or field that you expect, check the specifications of database output tables in your FVS KCP files. These tables are generated from FVS database output by the BioSum FVS module. Click on a table name to select it, and it will display in highlight.

c. **FVS Variable**: When you select an FVS Variable Table, this box will populate with a list of fields from that table. After you have selected your desired variable, click the **<Select>** button.
d. **Selected FVS Variable:** This read-only field will populate with the table and name of the FVS variable you select from the dropdown lists. This field must be populated before clicking the **<Calculate>** button and before any weights can be imported.
e. **view_weights:** This table lists the rxcycle, and rxyear associated with each analysis point. The rxyear should be used only as a point of reference as it could vary across variants in the project. Each weight is linked to a cycle, Pre or Post. For example: cycle 1 Pre and cycle 1 Post. Enter the appropriate values in the editable (red) column for each analysis point. Below are two possible weighted variable configurations:

Table 6.1: Sample weights for an FVS variable summed after each treatment

Cycle	Weight
1 (Pre)	0
1 (Post)	1
2 (Pre)	0
2 (Post)	1
3 (Pre)	0
3 (Post)	1
4 (Pre)	0
4 (Post)	1

Table 6.2: Sample weights for an FVS variable averaged over 8 cycle time points

Cycle	Weight
1 (Pre)	0.125
1 (Post)	0.125
2 (Pre)	0.125
2 (Post)	0.125
3 (Pre)	0.125
3 (Post)	0.125
4 (Pre)	0.125
4 (Post)	0.125

f. **TOTAL WEIGHTS**: This read-only field will automatically add the weight values together as you enter weights into the view_weights grid. This is a running total.

g. **Weighted Variable Name:** This read-only field contains the name of the weighted variable that displays on the Treatment Optimizer scenario screens. FIA Biosum generates the Weighted Variable Name by appending a suffix to the selected FVS variable name. For example: the first weighted variable associated with Surf_Flame_Sev is Surf_Flame_Sev_1, the second Surf_Flame_Sev_2 and so on.

h. **Description:** This is an optional, but strongly recommended, free form text field where you can type a description of the weighted FVS variable. This description appears on the main Calculated Variables screen and some Treatment Optimizer scenario configuration screens.

3. There are four enabled buttons on the new weighted FVS variable screen. They are:

a. **<Import weights>**: See the section below on Exporting and Importing variable weights for guidance on using this button

b. **<Help>**: As on other FIA Biosum screens, clicking on this button opens a new window that displays help text and instructions associated with this screen.

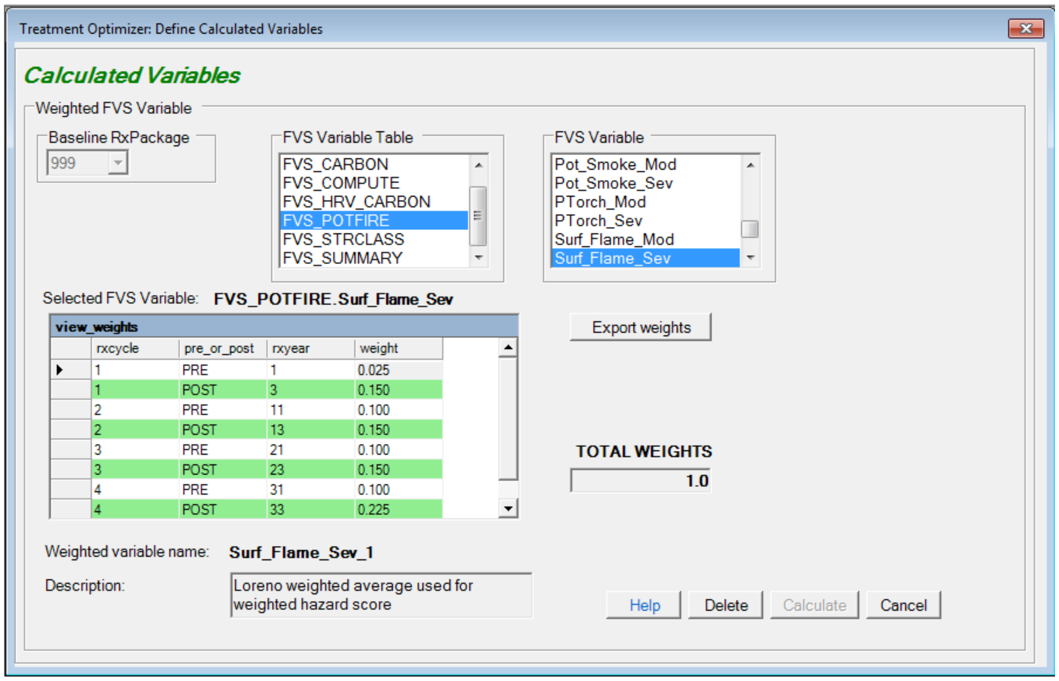
c. **<Calculate>**: The **<Calculate>** button saves the settings for the calculated variable to the project database. This button also calculates the weighted values for each stand + rxPackage + rxCycle and saves them to the PRE_<FVS_TABLE_NAME>_WEIGHTED and POST_<FVS_TABLE_NAME>_WEIGHTED tables where FVS_TABLE_NAME is the name of the source FVS output table. For example, if the source variable is from the FVS_POTFIRE table, the name of the PRE_<FVS_TABLE_NAME>_WEIGHTED table will be PRE_FVS_POTFIRE_WEIGHTED. These tables may be found in the /optimizer/db/ prepost_fvs_weighted.accdb.

Because the values are calculated when the FVS weighted variable is created, they can be shared across multiple optimization scenarios and will not be re-calculated when a scenario is run, shortening processing time.

d. **<Cancel>**: Closes the FVS variable properties screen without creating the weighted variables or saving any parameters specified and returns to the main Calculated Variables screen. This action will not save changes and cannot be undone.

Viewing the Properties of an existing weighted FVS Variable

1. Select the variable you want to review in the main Calculated Variables screen and click the <**Properties**> button. FVS variables can be identified by the 'FVS' value in the **Type** column.



Treatment Optimizer: Define Calculated Variables

Calculated Variables

Weighted FVS Variable

Baseline RxPackage: 999

FVS Variable Table: FVS_CARBON, FVS_COMPUTE, FVS_HRV_CARBON, FVS_POTFIRE, FVS_STRCLASS, FVS_SUMMARY

FVS Variable: Pot_Smoke_Mod, Pot_Smoke_Sev, PTorch_Mod, PTorch_Sev, Surf_Flame_Mod, Surf_Flame_Sev

Selected FVS Variable: FVS_POTFIRE.Surf_Flame_Sev

view_weights

rxcycle	pre_or_post	rxyear	weight
1	PRE	1	0.025
1	POST	3	0.150
2	PRE	11	0.100
2	POST	13	0.150
3	PRE	21	0.100
3	POST	23	0.150
4	PRE	31	0.100
4	POST	33	0.225

TOTAL WEIGHTS: 1.0

Weighted variable name: Surf_Flame_Sev_1

Description: Loreno weighted average used for weighted hazard score

Figure 6.3: Existing weighted FVS variable properties screen

2. The screen for viewing the properties for an existing FVS variable is almost identical to the screen for creating a new weighted FVS variable. See the section immediately above for detailed descriptions of each field.

   The weighted FVS variable properties screen is read-only. As weighted FVS variables can be shared across scenarios, it could lead to data corruption if a weighted FVS variable were re-configured and recalculated if it had been previously used in an optimization scenario.
3. There are four enabled buttons on the view weighted FVS variable properties screen. They are:
   a. **<Export weights>**: See the section below on Exporting and Importing variable weights for guidance on using this button
   b. **<Help>**: As on other FIA Biosum screens, clicking on this button opens a new window that displays help text and instructions associated with this screen.
   c. **<Delete>**: The **<Delete>** button deletes the weighted FVS variable configurations from the project databases and the weighted value columns from the PRE_<FVS_TABLE_NAME>_WEIGHTED and POST_<FVS_TABLE_NAME>_WEIGHTED tables

Weighted FVS variables can only be deleted if they are not associated with any optimization scenarios. FIA Biosum will prevent you from deleting an FVS variable if it is in use by an optimization scenario**.**

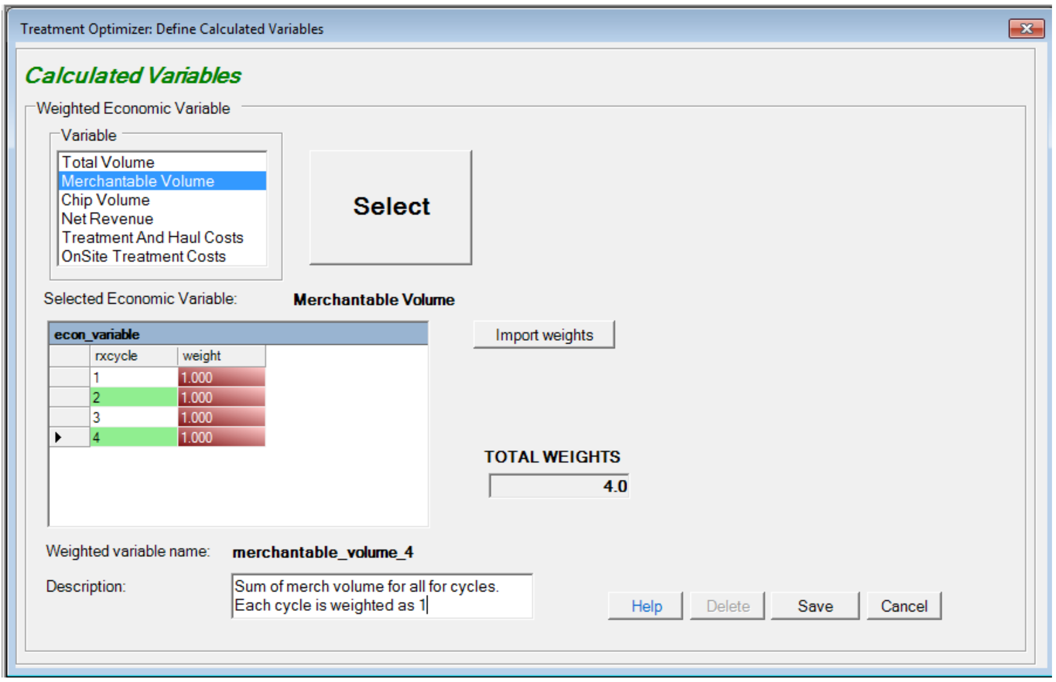
d. **<Cancel>**: Closes the FVS variable properties screen without saving the contents and returns to the main Calculated Variables screen.

Configuring a New Economic Variable

This function is similar to viewing the properties of an existing economic variable except that the screen is enabled to allow for variable customization. Economic variables are calculated when a scenario is run and their values are written to the post_economic_weighted table in the optimizer\db\scenario1\scenario_results.mdb.

Viewing the Properties of an Existing Economic Variable

1. Select the variable you want to review in the Calculated Variables tables and click the <**Properties**> button. FVS variables can be identified by the 'ECON' value in the **Type** column.



Selected Economic Variable: **Merchantable Volume**

econ_variable

rxcycle	weight
1	1.000
2	1.000
3	1.000
4	1.000

TOTAL WEIGHTS: 4.0

Weighted variable name: **merchantable_volume_4**

Description: Sum of merch volume for all for cycles. Each cycle is weighted as 1

Figure 6.4: Weighted Economic variable properties window

2. The fields on the weighted Economic variable properties screen are:
   a. **Variable**: The list of weighted economic variables that can be computed and tracked in BioSum:

Table 6.3: Six economic variables available in FIA Biosum

Variable	Description
Chip Volume	Chip volume generated from the stand/rxPackage
Merchantable Volume	Merchantable volume generated from the stand/rxPackage
Total Volume	Sum of chip and merchantable volume
Net Revenue	Revenue generated from the stand/package less the treatment and haul costs
Treatment and Haul Cost	Sum of harvest costs and haul costs
Onsite Treatment Cost	Costs of implementing the harvest and any surface fuel treatments

b. **Selected economic variable**: This read-only field shows the selected economic variable
c. **econ_variable**: This table displays the weight applied to each rxCycle. Below are two possible weighted variable configurations:

Table 6.4: Sample weights for an economic variable summed over 4 cycles

Cycle	Weight
1	1
2	1
3	1
4	1

Table 6.5: Sample weights for an economic variable averaged over 4 cycles

Cycle	Weight
1	0.25
2	0.25
3	0.25
4	0.25

d. **TOTAL WEIGHTS**: Sum of weights for all 4 cycles
e. **Weighted variable name**: This read-only field contains the name of the weighted variable that is displayed on the Treatment Optimizer scenario screens and as the column name in the post_economic_weighted table. FIA Biosum generates the weighted variable name by appending a suffix to the selected economic variable. For example: the first weighted variable associated with Merchantable Volume is merchantable_volume_1, the second merchantable_volume_2 and so on.

   f. **Description**: This is an optional, but highly recommended, free form text field that may contain a description of the weighed economic variable. This description appears on the initial weighted variables screen and some Treatment Optimizer scenario configuration screens.

4. There are four enabled buttons on the weighted economic variable properties screen. They are:
   a. **<Export weights>**: See the section below on Exporting and Importing variable weights for guidance on using this button
   b. **<Help>**: As on other FIA Biosum screens, clicking on this button opens a new window that displays help text and instructions associated with this screen.
   c. **<Delete>**: The **<Delete>** button deletes the weighted economic variable configurations from the project databases. Weighted economic variables can only be deleted if they are not associated with any existing Treatment Optimizer scenarios. FIA Biosum will prevent you from deleting an economic variable if it is in use by an optimization scenario**.** You are also prevented from deleting built-in weighted economic variables (described below).
   d. **<Cancel>**: Closes the economic variables property screen and returns to the main Calculated Variables screen.

Built-in Weighted Economic Variables

FIA Biosum provides a set of default weighted economic variables for your use. These weighted variables assign a weight of 1 to each cycle resulting in the sum of the selected economic variable over all 4 cycles. For example, **net_revenue_1** is the sum of net revenue for all 4 cycles. These built-in variables cannot be deleted. Thus the first custom net_revenue variable that can be created will be labeled net_revenue_2.

Exporting and Importing Variable Weights

Variable weights can be exported from existing weighted variables or imported into new variables. Exported weights are stored in a comma-separated text file to a computer file location specified by the analyst. A project's \optimizer\db folder is one logical location for storing these text files.

To export weights that have already been defined for or associated with an existing weighted variable, view the Calculate Variables dialog for that weighted variable. Then click the **<Export weights>** button and specify a location for the text file. The process should complete in a few seconds. To import weights into a new weighted variable, click the **<Export weights>** button and specify the location of the text file. This process should also complete in a few seconds and the imported weights will appear on the **view_weights** grid. Weighted FVS variables have eight weights and weighted economic variables have four weights, so their weight text files are not interchangeable.

To create a new weight text file without exporting weights, follow the specifications below for weighted FVS and economic variables and save the file with a **.txt** extension. The first value is always the RxCycle number. If BioSum can't read the import file, it will show a message and abort the import process.

Weighted FVS variable example

```
1,PRE,0.025
1,POST,0.15
2,PRE,0.1
2,POST,0.15
3,PRE,0.1
3,POST,0.15
4,PRE,0.1
4,POST,0.225
```

Weighted Economic variable example

```
1,0.33
2,0.33
3,0.33
4,0.01
```

FVS Variables - Overview

The FVS Variables tab is used to select a set of parameters that determines the best RxPackage for each stand. The FVS Variables tab has 3 child tabs. They are **Effective**, **Optimization** and **Tie Breaker**. **Effective** settings support the configuration of up to 4 stand attributes that can be used as a basis for determining whether or not an RxPackage is effective.

Only effective RxPackages are passed through to the **Optimization** analysis step. If no RxPackages are found to be effective for a stand, that stand will be dropped from further analysis. **Optimization** Settings allow the selection of one stand attribute of paramount interest to determine the best RxPackage, if more than one RxPackage is deemed effective. This window optionally allows you to filter on a weighted economic attribute, for example: net revenue > 0 or onsite treatment costs < 200.

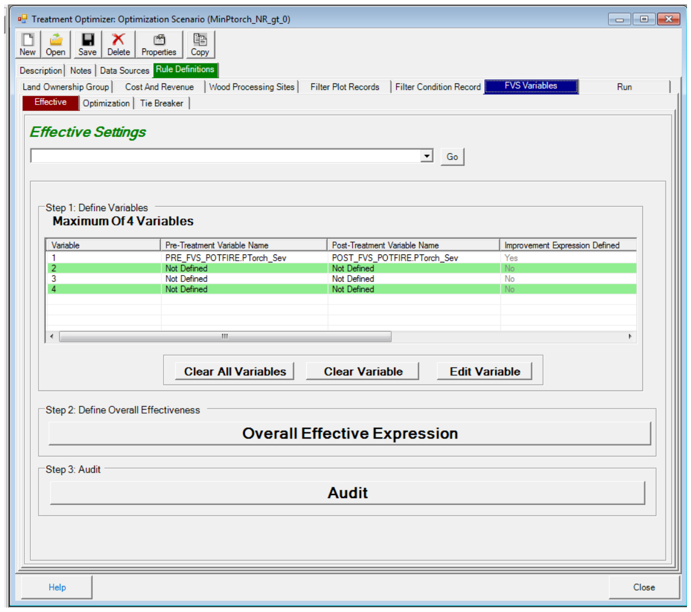
If multiple effective RxPackages for a stand have the same optimization attribute value and optionally, pass the weighted economic attribute filter, the **Tie Breaker** attribute will be applied in an effort to obtain a unique best selection for the stand. This attribute can be an FVS stand attribute or a weighted economic attribute. If more than one RxPackage remains after applying the **Tie Breaker**, the tie will be resolved by the **Last Tie-Break Rank**, a number, unique for each RxPackage, that can be assigned by the analyst. If no Rank is assigned, the value defaults to the nominal RxPackage number. The lowest **Last Tie-Break Rank** value resolves any remaining ties and determines the best RxPackage for that stand.

The output of a Treatment Optimizer scenario run is saved to \optimizer\scenario1\db\ scenario_results.mdb. There is also a runlog.txt file in the \optimizer\scenario1\db\ that may be helpful when troubleshooting Treatment Optimizer issues.

FVS Variables – Effective Settings

The Effective tab provides for specification of up to 4 stand attributes, generated in FVS output, to be used individually or in combination to determine whether treatment results in improvement or disimprovement. At least one stand attribute must be defined and used directly, or indirectly (via improvement and/or disimprovement) to effectiveness; however, as long as that requirement is met, there can be attributes specified that are not used in defining improvement, disimprovement, or effectiveness. For example, one might be used as a tie-breaker or in optimization.

Unless the selected FVS variable table name ends in "_WEIGHTED", only the FVS output for cycle 1 will be included in the analysis. Be cautious when combining cycle 1 and weighted variables in the same analysis as they cover different time periods.



Variable	Pre-Treatment Variable Name	Post-Treatment Variable Name	Improvement Expression Defined
1	PRE_FVS_POTFIRE.PTorch_Sev	POST_FVS_POTFIRE.PTorch_Sev	Yes
2	Not Defined	Not Defined	No
3	Not Defined	Not Defined	No
4	Not Defined	Not Defined	No

Figure 6.14: The Effective Settings main menu

The collection of Effective Settings variables is summarized on a table on the Effective Settings main menu. The columns are:

1. **Variable**: The identifier of the variable. Each value is unique and ranges between 1 and 4. FIA Biosum supports a maximum of 4 FVS output stand attributes.
2. **Pre-Treatment Variable Name**: The concatenated names (table.field) of the table and field containing the pre-treatment stand attribute. When working with a weighted variable, note that all variable names in the Optimizer workflow that begin with "pre" or “PRE” are, in fact, weighted combinations over all 4 cycles for the silvicultural sequence (package) selected as “base” for that weighted variable (often that is the grow-only sequence, 999).
3. **Post-Treatment Variable Name**: The concatenated names of the table and field of the post-treatment attribute. Table location conventions match those of Column 2. When working with a weighted variable, note that all variable names in the Optimizer workflow that begin with "post" or “POST” are, in fact, weighted combinations over all 4 cycles for the silvicultural sequence (package) being evaluated.
4. **Improvement Expression Defined**: Indicates if an improvement expression is defined for this variable
5. **Disimprovement Expression Defined**: Indicates if an disimprovement expression is defined for this variable
6. **Effective Expression Defined**: Indicates if an effective expression is defined for this variable

Note: At least one effective expression must be defined using at least one of the 4 variables, for their calculated improvement/disimprovement yes/no attributes for an optimization scenario to be valid.

The initial page of the Effective Settings screen has 6 buttons:

1. **<Go>**: Use this button, after making a selection from the dropdown list to its left to gain immediate access to any step in this "wizard"
2. **<Clear All Variables>**: Clears configuration for all variables displayed in the **Step 1: Define Variables** table
3. **<Clear Variable>**: Clears configuration for the variable currently selected via the **Step 1: Define Variables** table
4. **<Edit Variable>**: Opens the first page of the configuration wizard for the variable selected in the **Step 1: Define Variables** table
5. **<Overall Effective Expression>**: Opens the overall effective expression builder which determines if an RxPackage is effective for the stand.
6. **<Audit>**: Runs a series of validation checks to ensure that the Effective Settings can be used by a scenario run.

Defining or editing an effectiveness variable is a 4 step process:

1. Select the source table and field for the stand attribute or variable
2. Define a SQL expression for what constitutes variable post-treatment improvement (shifting the variable to better value)
3. Define a SQL expression for what constitutes variable post-treatment disimprovement (shifting the variable to a worse value)
4. Define a SQL expression for what constitutes variable effective treatment. This SQL expression can utilize the pre- and post- values of the variable, the change in the variable, and the outcomes of the better/worse expressions from steps 2 and 3, if defined.

Selecting an FVS Stand Attribute

To select a new FVS stand attribute (Step 1) choose **Select Variable X…** from the dropdown list at the top of the screen and click the **<Go>** button. To edit an existing FVS stand attribute, select row containing the attribute from the table on the **Effective Settings** main menu and click the <**Edit Variable**> button.

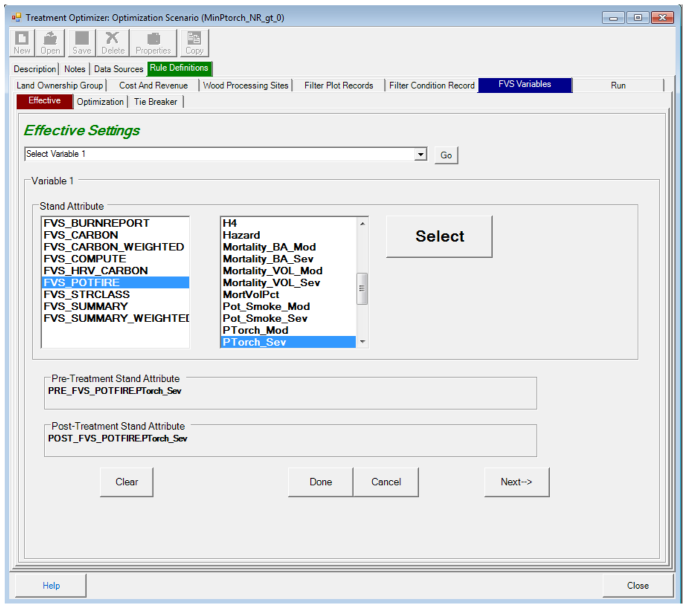


Figure 6.15: Selecting an FVS Stand Attribute

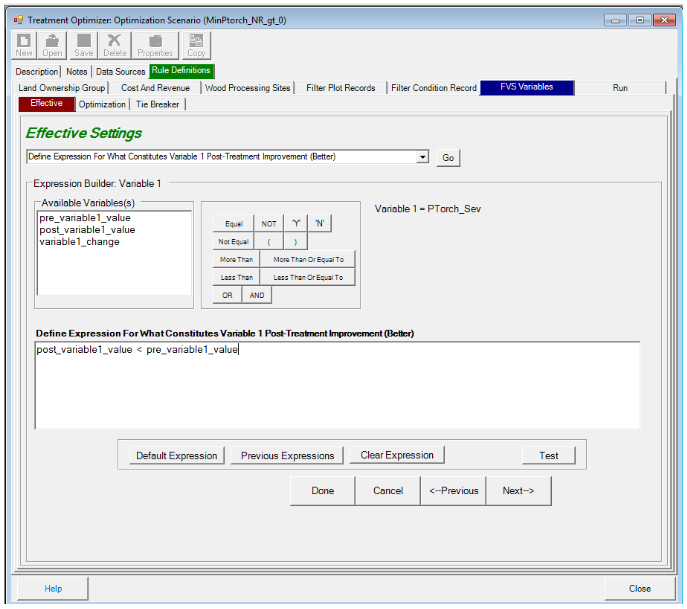
When you select an FVS output table from the first list, the second list will automatically populate with the names of the columns available in the selected table. Choose a stand attribute from the right-hand list and then click the <**Select**> button to select it. The **Pre** and **Post Treatment Stand Attribute** fields will display the name of the selected element. Click the <**Next->**> button to configure the expression for the selected Effectiveness Variable (step 2). Note that the window and method for selecting an FVS Stand Attribute are identical on the Effectiveness, Optimization, and Tie Breaker stand attribute tabs. As a point of reference, table names that include the suffix "_WEIGHTED" are located in the [projectname]/optimizer/db/prepost_fvs_weighted.accdb database; other tables are in the FVS output directory: [projectname]/fvs/db.

Effectiveness Variable Expression Builder

The expression builder is used in steps 2, 3, and 4 of editing an effectiveness variable and also when defining the overall effectiveness expression. It contains the following elements:

1. **<Go>**: Use this button with the dropdown list on the left to gain immediate access to any wizard prompt location
2. **Available Variable(s)**: List the variables that can be used to generate your expression. Use the buttons next to the list to add operators and qualifiers to your expression.
3. **Define expression for what constitutes** …: Contains the expression that will be used during a Treatment Optimizer run. The text can either be built using the buttons above or manipulated directly by typing in the box.
4. **<Test>**: Use this button to test your expression and make sure it is valid
5. **<Default Expression>**: Generates a sample expression in the **Define expression for what constitutes** … box. This should be used as example syntax for composing your own expressions but is not recommended for actual use in most scenarios.

6. **<Previous Expression>**: Lists the previous expression SQL statements by scenario. Current expressions are indicated with a ‘Y’ in the current_YN field. Using an expression from a previous scenario is often a good starting point for novice FIA Biosum users.
7. **<Clear Expression>**: Clears the **Define expression for what constitutes** … box.
8. **<Done>**: Saves your expression to memory and closes the expression builder. Use the **<Save>** button in the toolbar after clicking **<Done>** to save your configuration changes to your computer. Your changes are also automatically saved before an optimization scenario is run.
9. **<Cancel>**: Closes the expression builder without saving your changes. Note that when you are in edit mode, the settings accessed via the **Optimization** and **Tie Breaker** tabs will be read-only until you use the **<Done>** or **<Cancel>** buttons to exit the expression builder.



Treatment Optimizer: Optimization Scenario (MinPtorch_NR_gt_0)

Effective Settings

Define Expression For What Constitutes Variable 1 Post-Treatment Improvement (Better)

Expression Builder: Variable 1

Available Variables(s)
pre_variable1_value
post_variable1_value
variable1_change

Variable 1 = PTorch_Sev

Define Expression For What Constitutes Variable 1 Post-Treatment Improvement (Better)

```
post_variable1_value < pre_variable1_value
```

Figure 6.16: Defining variable 1 effective treatment expression using the expression builder

Audit for Effective Variables

The Effective Variables configuration must pass the following tests to be valid:

1. Ensure the user-defined FVS variable tables exist
2. Ensure the user-defined FVS columns exist
3. Ensure overall effectiveness expression exists
4. Ensure no FVS variables are defined more than once
5. Ensure effective expressions all have valid variable references. Example of an invalid variable reference: you delete FVS variable #4 but forget to remove FVS variable #4 from the effective expression.

Effective Variables output

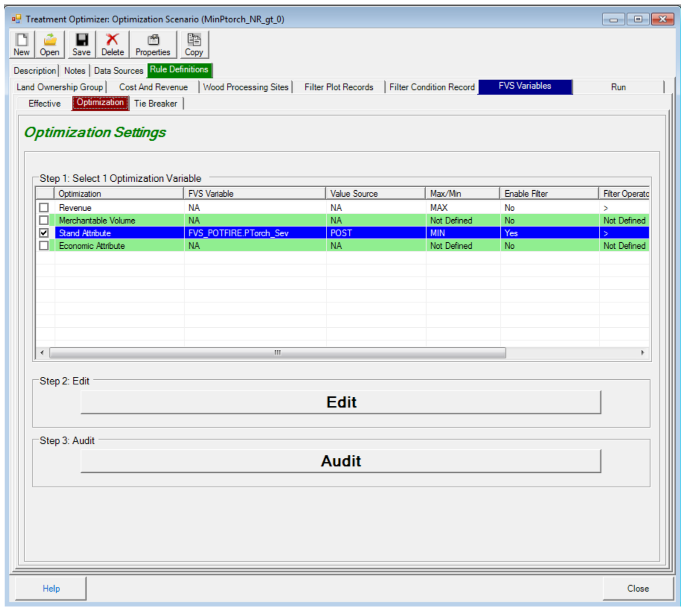
The output from Effective Variables processing is written to the cycle1_effective table in the scenario_results.mdb. The following columns may benefit from a brief description:

1. **nr_dpa**: net revenue for cycle 1
2. If an Economic filter was configured in the **Optimization Settings**, the name of the filter, for example **net_revenue_1**, and the weighted values will be in this table
3. **overall_effective_yn**: indicates if the rxPackage was deemed effective according to the selected criteria. Only stands with a value of 'Y' will be included in downstream analysis

FVS Variables – Optimization Settings

The **Optimization Settings** require you to select a single optimization variable of interest. The variable can be an FVS stand attribute, an economic stand attribute, revenue, or merchantable volume. The FVS stand attribute, revenue, and merchantable volume options apply only to cycle 1. The FVS stand attribute may apply to all 4 cycles if a WEIGHTED stand attribute is selected. The economic stand attribute is always WEIGHTED and applies to 4 cycles. However, depending on the weights supplied (zeros are valid), it could apply to any single cycle or subset of cycles.

The optimization variable configuration includes options for favoring Maximum or Minimum aggregate values and, optionally, excluding stands that do not meet a defined Economic threshold filter.



	Optimization	FVS Variable	Value Source	Max/Min	Enable Filter	Filter Operatc
☐	Revenue	NA	NA	MAX	No	>
☐	Merchantable Volume	NA	NA	Not Defined	No	Not Defined
☑	Stand Attribute	FVS_POTFIRE.PTorch_Sev	POST	MIN	Yes	>
☐	Economic Attribute	NA	NA	Not Defined	No	Not Defined

Figure 6.17: Optimization Settings main menu

The Optimization settings are summarized in a table on the Optimization settings main menu. The columns are:

1. The first unlabeled column is a checkbox column. Only one optimization attribute per optimization scenario can be selected.
2. **Optimization**: Lists the optimization attribute type; the options are Revenue, Merchantable Volume, Stand Attribute, and Economic Attribute

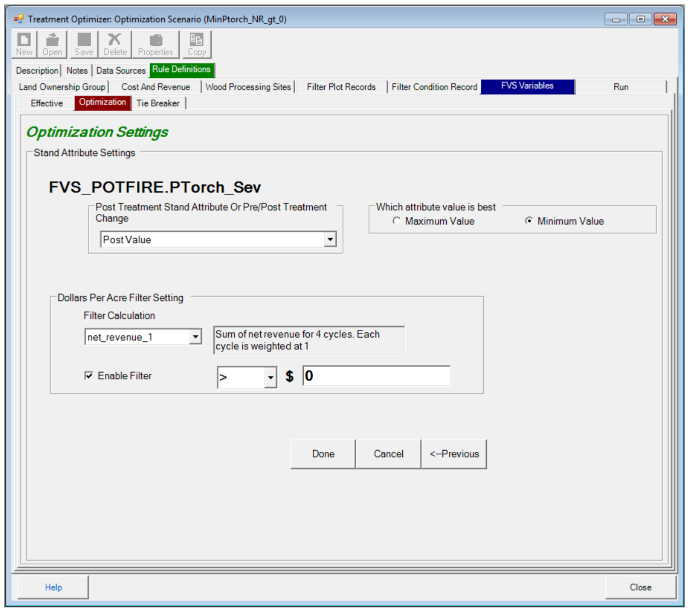
3. **FVS Variable**: For stand attributes and economic attributes, the name of the selected variable appears. Stand attributes include the FVS table and field name, concatenated by a period '.'. Economic attributes show the name of the weighted economic variable. 'NA' appears in this column for the cycle 1 not FVS-based Revenue and Merchantable volume variables.
4. **Value Source**: Only populated for stand attributes. The values will be either POST (treatment value) or POST-PRE (treatment change value). 'NA' appears in this column for all other optimization attribute types.
5. **Max/Min**: Should the maximum or minimum value of the attribute be considered optimal?
6. **Enable Filter**: Has an economic filter been configured?
7. **Filter Operator**: The operator for the economic filter
8. **Filter Value**: The threshold value set for the economic filter. If the filter is not enabled, this value will be ignored

The Optimization Settings main menu has 2 buttons:

1. **<Edit>**: Opens the configuration page(s) for the attribute selected in the Optimization Variable table
2. **<Audit>**: Runs a series of checks to ensure that the Optimization attribute and, optionally economic filter, are configured correctly

Editing the Optimization Attribute

If you select a Stand Attribute as an optimization attribute, the first step is to select the table and column for the attribute. The follow-on configuration window allows you to specify additional details. The configuration screens for Economic Attribute, Revenue, and Merchantable Volume are similar to the follow-on configuration window although they may not contain all of the fields as not all fields are relevant to all variable types.



Optimization Settings

Stand Attribute Settings

FVS_POTFIRE.PTorch_Sev

Post Treatment Stand Attribute Or Pre/Post Treatment Change: Post Value

Which attribute value is best: ( ) Maximum Value [✓ Minimum Value]

Dollars Per Acre Filter Setting

Filter Calculation: net_revenue_1 — Sum of net revenue for 4 cycles. Each cycle is weighted at 1

☑ Enable Filter > $ 0

Figure 6.18: Follow-on stand attribute settings configuration window

The name of the selected Optimization variable of interest appears near the top of the configuration window. It will be the FVS variable table and field name, the economic attribute name, Merchantable Volume, or Revenue. The other fields on this screen are:

1. **Post Treatment Stand Attribute Or Pre/Post Treatment Change**: Choose between the post-treatment stand attribute value or the difference between the post and pre-treatment values. The pre-treatment value is subtracted from the post-treatment value.
2. **Which Attribute Value is Best**: Select minimum or maximum value for the stand attribute, economic attribute, revenue, or merchantable volume
3. The following 4 fields work together to configure the optional Economic filter threshold. If a stand and RxPackage combination fails to meet the threshold, it will be excluded from downstream processing.
    a. **Filter Calculation**: This is a select list containing all available net revenue and onsite treatment cost calculations. When you select an item from the list, the calculation description appears next to the select list. Currently net_revenue_1 and onsite_treatment_costs_1 are the only calculations available. These are the sum of net revenue or onsite treatment costs across all 4 cycles. In a future release, you will be able to configure custom calculations using the FIA Biosum calculated variable functionality.
    b. **Dollars Per Acre Filter Setting**: Checking this checkbox enables the Economic filter threshold. There is a select list containing a choice of operators (>, <, >=, <=, <>). Select the appropriate operator from this list. The last field is a freeform numeric field that allows you to enter the threshold value. Negative numbers are permitted.

Below are 3 Optimization attribute configuration examples:

- **Optimization**: Stand Attribute
- **FVS Variable**: FVS_POTFIRE_WEIGHTED.Torch Index
- **Pre/Post Treatment Change**
- **Which Attribute Value is Best**: Maximum
- **Economic Filter:** enabled where net_revenue_1 > 100
- **Synopsis:** Find the treatment for each stand where the weighted torch index increased the most after treatment. Only consider stands where the sum of net revenue over 4 cycles is greater than 100.

- **Optimization**: Merchantable Volume
- **FVS Variable**: N/A
- **Which Attribute Value is Best**: Minimum
- **Economic Filter:** disabled
- **Synopsis:** Find the treatment for each stand that cuts the least amount of merchantable volume for cycle 1. *Note that merchantable volume and revenue optimizations apply to cycle 1 only.*

- **Optimization**: onsite_treatment_costs_1
- **FVS Variable**: N/A
- **Which Attribute Value is Best**: Minimum
- **Economic Filter:** enabled where net_revenue_1 > 0
- **Synopsis:** Find the treatment for each stand that has the lowest onsite treatment costs where net revenue is greater than 0. These are both built-in economic variables and include the sum of all 4 cycles in their calculation.

Audit for Optimization Attribute
The Optimization attribute must pass the following tests to be valid:

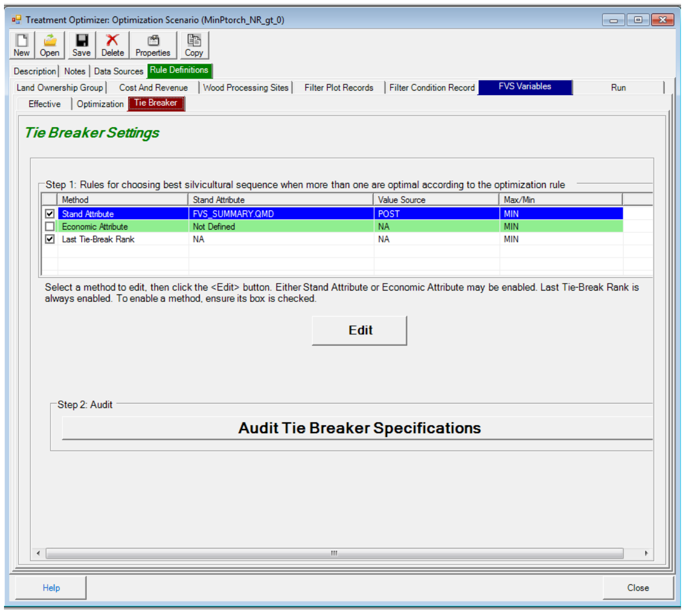
1. One optimization attribute of interest must be checked
2. If the optimization is the FVS Post variable then ensure the post column exists
3. If the optimization is the FVS Post-Pre change than ensure both the PRE and POST columns exist
4. The optimization variable has an aggregate definition of MAX or MIN
5. If enabled, ensure the filter operator is valid

Optimization Attribute output
The output from Optimization processing is written to the cycle1_optimization table in the scenario_results.mdb. All stand RxPackage combinations deemed to be effective are included in this table along with their optimization attribute values.

The **affordable_yn** column indicates if a stand RxPackage combination met the economic filter threshold. If an economic filter isn't configured, this field will always be set to 'Y'. If an economic filter is configured, there will be a column with the name of the selected economic attribute, for example: net_revenue_1. This column will be populated with the values of the economic attribute.
If a weighted variable is used anywhere in a scenario configuration, the prefix cycle1_ will be replaced with the prefix all_cycles_.

FVS Variables – Tie Breaker Settings

One and only one RxPackage per stand can be selected as "best" during a Treatment Optimizer scenario run. It is highly probable that multiple RxPackages for a stand may meet the Effectiveness criteria, exceed the economic filter threshold, and have the same Optimization attribute value. For example, if two RxPackages differ only in that one has a diameter cap of, say 21 inches and the other does not, the effectiveness, economics and optimization outcomes are identical for stands containing no trees over 21 inches. In these cases, the **Tie Breaker** settings are used to break any ties.
Tie breaking can be a two-step process. Optionally, you can select a Stand Attribute or Economic Attribute to break any ties. That would not break a tie in the case of the example just described, but might break the tie if, say, two treatments were effective, met the economic filter criterion and happened to achieve the same predicted basal area mortality (if that were the optimization variable). This tie-breaker attribute cannot be the same data element as the Optimization attribute (because the ties that need to be broken arise after application of optimization). In this example Tie Breaker might be set to an economic variable so as to minimize treatment cost or maximize net revenue, or to an Stand attribute such as minimizing a pTorch attribute (probability of torching).
If multiple RxPackages are still tied or a Tie Breaker attribute is not selected, the required Last Tie-Break Rank value will be used to select the "best" RxPackage. On the Tie-Break Rank screen, a unique, integer value is assigned to each RxPackage. The RxPackage with the lowest Tie-Break Rank value will be selected.



Tie Breaker Settings

Step 1: Rules for choosing best silvicultural sequence when more than one are optimal according to the optimization rule

	Method	Stand Attribute	Value Source	Max/Min
☑	Stand Attribute	FVS_SUMMARY.QMD	POST	MIN
☐	Economic Attribute	Not Defined	NA	MIN
☑	Last Tie-Break Rank	NA	NA	MIN

Select a method to edit, then click the <Edit> button. Either Stand Attribute or Economic Attribute may be enabled. Last Tie-Break Rank is always enabled. To enable a method, ensure its box is checked.

Step 2: Audit

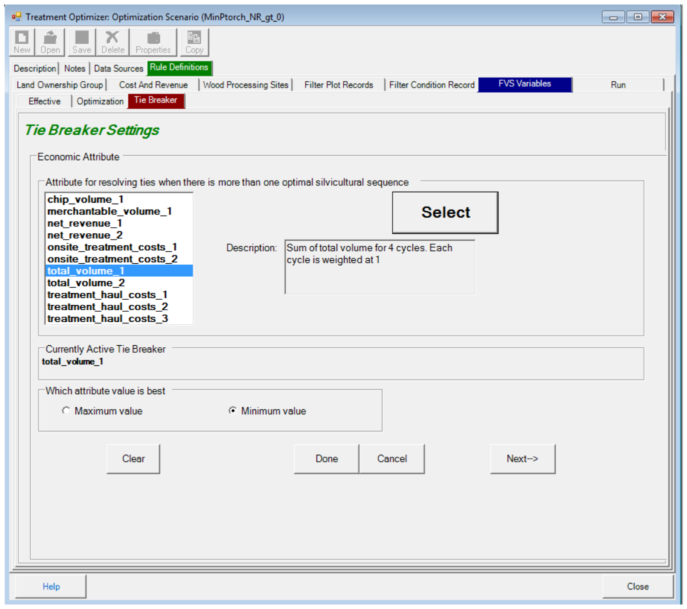
Figure 6.19: Tie Breaker settings main menu

The centerpiece of the Tie Breaker settings main menu screen is a table that lists the potential Tie Breaker attributes of interest: Stand Attribute, Economic Attribute, and Last Tie-Break Rank. The only required attribute is Last Tie-Break Rank (hence it cannot be unchecked). The columns on this table are:

1. The first unlabeled column is a checkbox column. If the checkbox is checked, it means the attribute is selected as a Tie Breaker. In addition to Last Tie-Break Rank, you can select a stand attribute OR and economic attribute but not both.
2. **Method**: Lists the Tie Breaker attribute type; the choices are Stand Attribute, Economic Attribute, and Last Tie-Break Rank.
3. **Stand Attribute**: For stand attributes and economic attributes, the name of the selected variable appears. Stand attributes include the FVS table and field name, concatenated by a period '.'. Economic attributes show the name of the weighted economic variable. The field is not applicable 'NA' for Last Tie-Break Rank.
4. **Value Source**: Only populated for stand attributes. The values will be either POST (treatment value) or POST-PRE (change in treatment value). 'NA' appears in this column for all other Tie Breaker attribute types.
5. **Max/Min**: Should the maximum or minimum value of the attribute be considered best? For Last Tie-Break Rank, this will always be set to MIN.

Editing an Economic or Stand Attribute Tie Breaker

If you wish to use an Economic Attribute as Tie Breaker attribute, the first step is to select the weighted economic attribute. When you select an attribute from the list, a description appears in the **Description** box. After you use the **<Select>** button to finalize your choice, the name of the attribute will appear as the **Currently Active Tie Breaker**. After selecting the attribute, choose either **Minimum** or **Maximum Value** as best. Important Note: The choice is not implemented until you click Select and the name of the attribute appears in the Currently Active Tie Breaker display in this dialog. Running Optimizer with a checked Tie Breaker attribute that does not have a selected attribute identified (i.e., Stand Attribute shows as “Not Defined”) will lead to an error relating to “primary key cannot contain a Null value” when executing the last workflow task in Optimizer’s Run tab.



Tie Breaker Settings

Economic Attribute

Attribute for resolving ties when there is more than one optimal silvicultural sequence

- chip_volume_1
- merchantable_volume_1
- net_revenue_1
- net_revenue_2
- onsite_treatment_costs_1
- onsite_treatment_costs_2
- total_volume_1
- total_volume_2
- treatment_haul_costs_1
- treatment_haul_costs_2
- treatment_haul_costs_3

Description: Sum of total volume for 4 cycles. Each cycle is weighted at 1

Currently Active Tie Breaker
total_volume_1

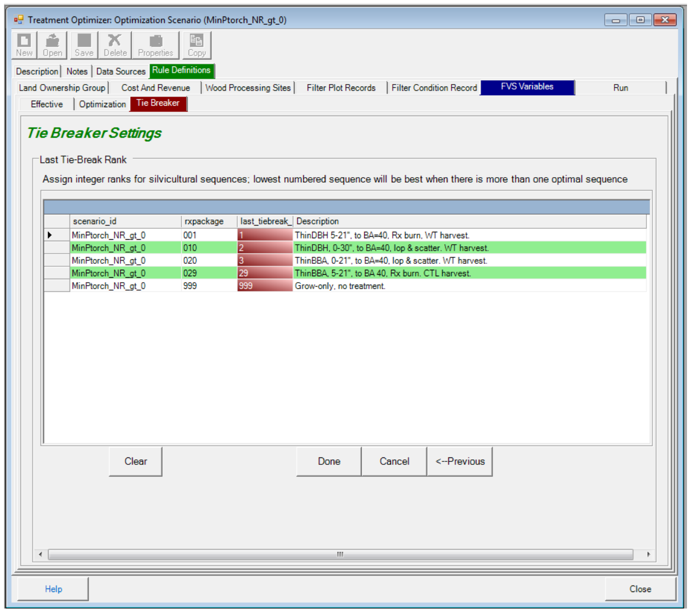
Which attribute value is best
Maximum value [✓ Minimum value]

Figure 6.20: Editing an Economic Tie Breaker attribute

The Stand Attribute Tie Breaker screen is similar but also requires you to select **Post Treatment Stand Attribute Or Pre/Post Treatment Change**: Choose between the post-treatment stand attribute value or the difference between the post and pre-treatment values. The pre-treatment value is subtracted from the post-treatment value.
There are four buttons at the base of the Tie Breaker **Stand** or **Economic** attribute screen. They are:

1. <**Clear**>: Clears your selections from the screen
2. **<Done>**: Saves your selections to the computer’s memory and returns to the Tie Breaker settings main menu. You must use the **<Save>** button at the top of the Treatment Optimizer dialog for such changes in the project configuration to persist after FIA Biosum is closed.
3. **<Cancel>**: Discards your changes and returns to the Tie Breaker settings main menu.
4. **<Next->**: Saves your selections and proceeds to the Last Tie-Break Rank window. You must use the **<Save>** button to save your configuration choices to your computer.

The **Last Tie-Break Rank** tab requires assigning a unique rank to each rxPackage in the red-highlighted column. Following FIA Biosum convention, only the red column is editable. The **last_tiebreak_rank** value is the ultimate tiebreaker if more than one rxPackage in a stand has the same **Optimization** and **Tie Breaker** attribute values. The rxPackage with the lowest **last_tiebreak_rank** value will be assigned as best.



Tie Breaker Settings

Last Tie-Break Rank

Assign integer ranks for silvicultural sequences; lowest numbered sequence will be best when there is more than one optimal sequence

scenario_id	rxpackage	last_tiebreak_	Description
MinPtorch_NR_gt_0	001	1	ThinDBH 5-21", to BA=40, Rx burn, WT harvest.
MinPtorch_NR_gt_0	010	2	ThinDBH, 0-30", to BA=40, lop & scatter. WT harvest.
MinPtorch_NR_gt_0	020	3	ThinBBA, 0-21", to BA=40, lop & scatter. WT harvest.
MinPtorch_NR_gt_0	029	29	ThinBBA, 5-21", to BA 40, Rx burn. CTL harvest.
MinPtorch_NR_gt_0	999	999	Grow-only, no treatment.

Figure 6.21: The Last Tie-Break Rank window

There are four buttons at the base of the **Last Tie-Break Rank** screen. They are:

1. <**Clear**>: Clears your selections from the screen
2. **<Done>**: Saves your selections to the computer's memory and returns to the Tie Breaker settings main menu. You must use the **<Save>** button to persist your configurations after FIA Biosum is closed.
3. **<Cancel>**: Discards your changes and returns to the Tie Breaker settings main menu.
4. **<-Previous**>: Saves your selections to the computer's memory and returns to either the Stand or Economic attribute window (depending on which was previously selected). You must use the **<Save>** button to save your configuration choices to your computer.

The <**Audit Tie Breaker Specifications**> button runs an audit on the Tie Breaker settings to ensure that the following requirements are met:

1. A last tie-break rank value must be populated for each rxPackage
2. Last tie-break rank values must be unique (one per rxPackage).
3. The stand or economic attribute must be defined if it is checked
4. If enabled, the Tie Breaker attribute cannot be the same as the Optimization attribute

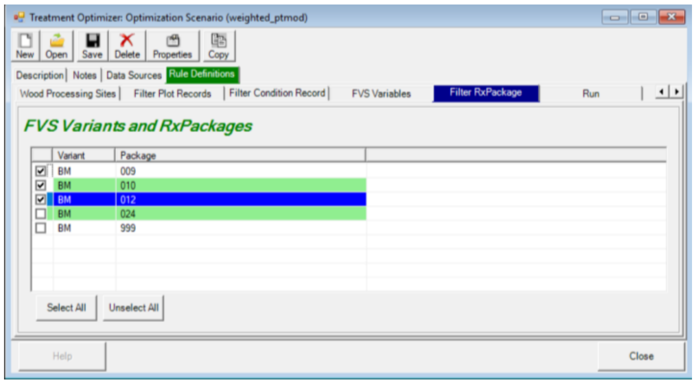
Tie Breaker output

The output from Tie Breaker processing is written to the cycle1_best_rx_summary_before_tiebreaks, cycle1_best_rx_summary, and cycle1_best_rx_summary_air_dest tables in the scenario_results.mdb. The cycle1_best_rx_summary_before_tiebreaks table includes all stand-RxPackage combinations that passed the Optimization attribute and revenue filter threshold tests. It appends the tiebreaker_value (if configured) and last_tiebreak_rank value to each row in the RxPackage table.

The cycle1_best_rx_summary table applies the Tie Breaker settings and only includes rows with the 'best' RxPackage for each stand. The cycle1_best_rx_summary_air_dest does the same for air curtain destruction plots.
If a weighted variable is used anywhere in a scenario configuration, the prefix cycle1_ will be replaced with the prefix all_cycles_.

Filter RxPackage

It is often useful to include a broad spectrum of silvicultural sequence options in a BioSum project to address questions and promote learning through experimentation and the answering of "what-if" questions. It is also frequently useful to exclude some sequences from consideration when optimizing. For example, a sequence that includes clearcut treatments might have been modeled to gauge the maximum potential wood flow potential of a landscape, but the analyst would like to exclude it from being selected under a certain policy scenario. This screen (Figure 6.22) provides for selecting which sequences (packages) to consider as valid options for the optimization. Packages with ticked selection boxes will be considered; those with blank boxes will not.



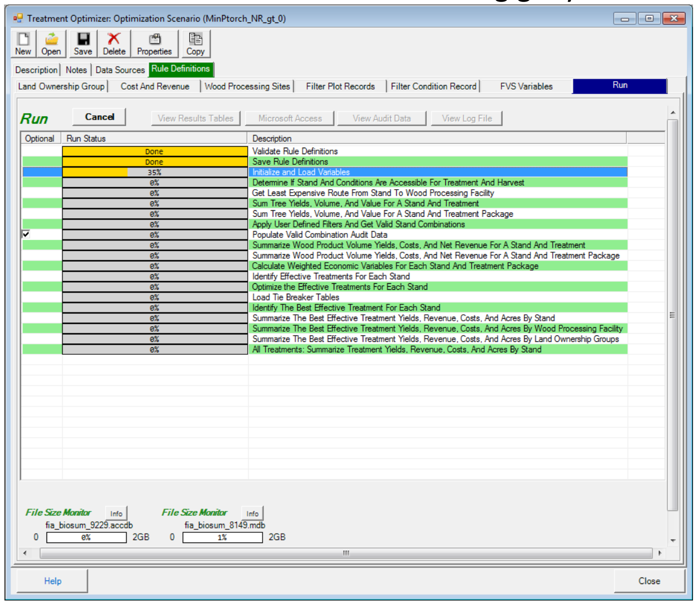
	Variant	Package
☑	BM	009
☑	BM	010
☑	BM	012
☐	BM	024
☐	BM	999

Figure 6.22: The FVS Variants and RxPackages selection grid

For projects with multiple variants, one would typically select the same packages to consider in each of the variants, but because sometimes this assumption might not hold (for example, it might be desirable to run an optimization for less than all the variants in a project), BioSum provides the flexibility for considered packages to differ among variants, so the analyst must make sure to make package choices separately for each variant via these selection boxes.

Run

The Run tab executes the optimization scenario that you have configured and writes out the results to a set of tables in the \optimizer\scenario1\db\scenario_results.mdb. Before commencing the scenario calculations, a series of audits is run to ensure that the optimization configuration is valid. At the base of the run screen are two file size monitors that report the current size of the temporary databases used when calculating an optimization scenario. The maximum size of an MS Access database is 2 gigabytes.



Optional	Run Status	Description
	Done	Validate Rule Definitions
	Done	Save Rule Definitions
	35%	Initialize and Load Variables
	0%	Determine If Stand And Conditions Are Accessible For Treatment And Harvest
	0%	Get Least Expensive Route From Stand To Wood Processing Facility
	0%	Sum Tree Yields, Volume, And Value For A Stand And Treatment
	0%	Sum Tree Yields, Volume, And Value For A Stand And Treatment Package
	0%	Apply User Defined Filters And Get Valid Stand Combinations
✓	0%	Populate Valid Combination Audit Data
	0%	Summarize Wood Product Volume Yields, Costs, And Net Revenue For A Stand And Treatment
	0%	Summarize Wood Product Volume Yields, Costs, And Net Revenue For A Stand And Treatment Package
	0%	Calculate Weighted Economic Variables For Each Stand And Treatment Package
	0%	Identify Effective Treatments For Each Stand
	0%	Optimize the Effective Treatments For Each Stand
	0%	Load Tie Breaker Tables
	0%	Identify The Best Effective Treatment For Each Stand
	0%	Summarize The Best Effective Treatment Yields, Revenue, Costs, And Acres By Stand
	0%	Summarize The Best Effective Treatment Yields, Revenue, Costs, And Acres By Wood Processing Facility
	0%	Summarize The Best Effective Treatment Yields, Revenue, Costs, And Acres By Land Ownership Groups
	0%	All Treatments: Summarize Treatment Yields, Revenue, Costs, And Acres By Stand

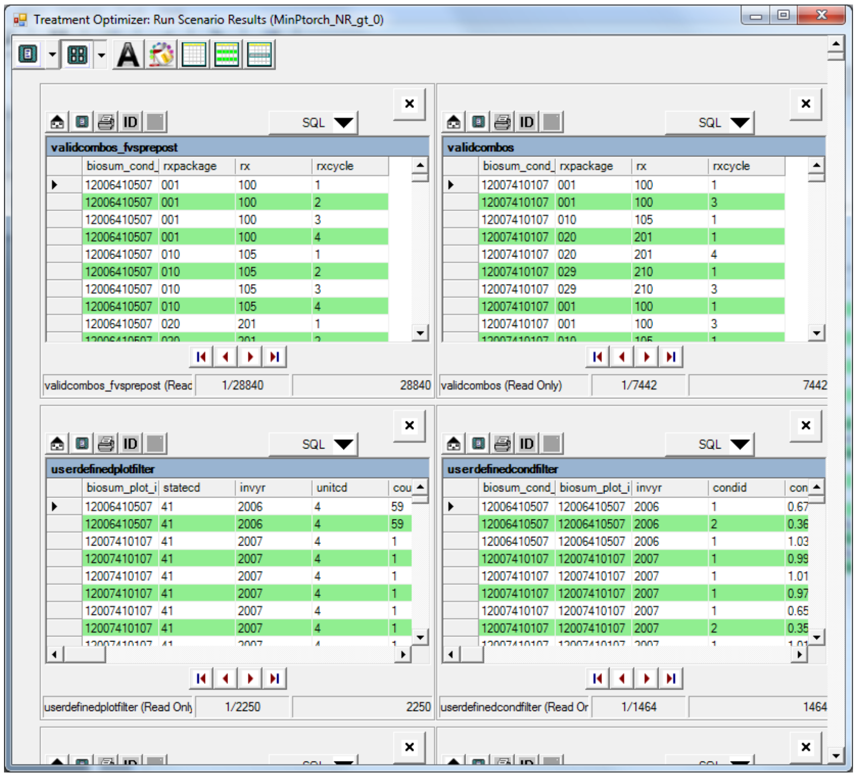
fia_biosum_9229.accdb 0 0% 2GB

fia_biosum_8149.mdb 0 1% 2GB

Figure 6.23: The Run scenario window

There are 5 buttons above the table on the **Run** screen. They are:

1. **<Start>**: Clicking this button begins an optimization scenario analysis run. While the scenario is running, the label of this button changes to <**Cancel**>. Click the <**Cancel**> button to interrupt the optimization scenario run for any reason.
2. **<View Results Tables>**: This button is enabled if the scenario_results.mdb exists for the current scenario. Click this button to view the output tables from within FIA Biosum. If all of the tables aren't visible in the viewer, click and drag the lower right-hand corner to enlarge the window.


Treatment Optimizer: Run Scenario Results (MinPtorch_NR_gt_0)

validcombos_fvsprepost

biosum_cond_	rxpackage	rx	rxcycle
12006410507	001	100	1
12006410507	001	100	2
12006410507	001	100	3
12006410507	001	100	4
12006410507	010	105	1
12006410507	010	105	2
12006410507	010	105	3
12006410507	010	105	4
12006410507	020	201	1

validcombos_fvsprepost (Read | 1/28840 | 28840

validcombos

biosum_cond_	rxpackage	rx	rxcycle
12007410107	001	100	1
12007410107	001	100	3
12007410107	010	105	1
12007410107	020	201	1
12007410107	020	201	4
12007410107	029	210	1
12007410107	029	210	3
12007410107	001	100	1
12007410107	001	100	3

validcombos (Read Only) | 1/7442 | 7442

userdefinedplotfilter

biosum_plot_i	statecd	invyr	unitcd	cou
12006410507	41	2006	4	59
12006410507	41	2006	4	59
12007410107	41	2007	4	1
12007410107	41	2007	4	1
12007410107	41	2007	4	1
12007410107	41	2007	4	1
12007410107	41	2007	4	1
12007410107	41	2007	4	1

userdefinedplotfilter (Read Only | 1/2250 | 2250

userdefinedcondfilter

biosum_cond_	biosum_plot_i	invyr	condid	con
12006410507	12006410507	2006	1	0.67
12006410507	12006410507	2006	2	0.36
12006410507	12006410507	2006	1	1.03
12007410107	12007410107	2007	1	0.99
12007410107	12007410107	2007	1	1.01
12007410107	12007410107	2007	1	0.97
12007410107	12007410107	2007	1	0.65
12007410107	12007410107	2007	2	0.35

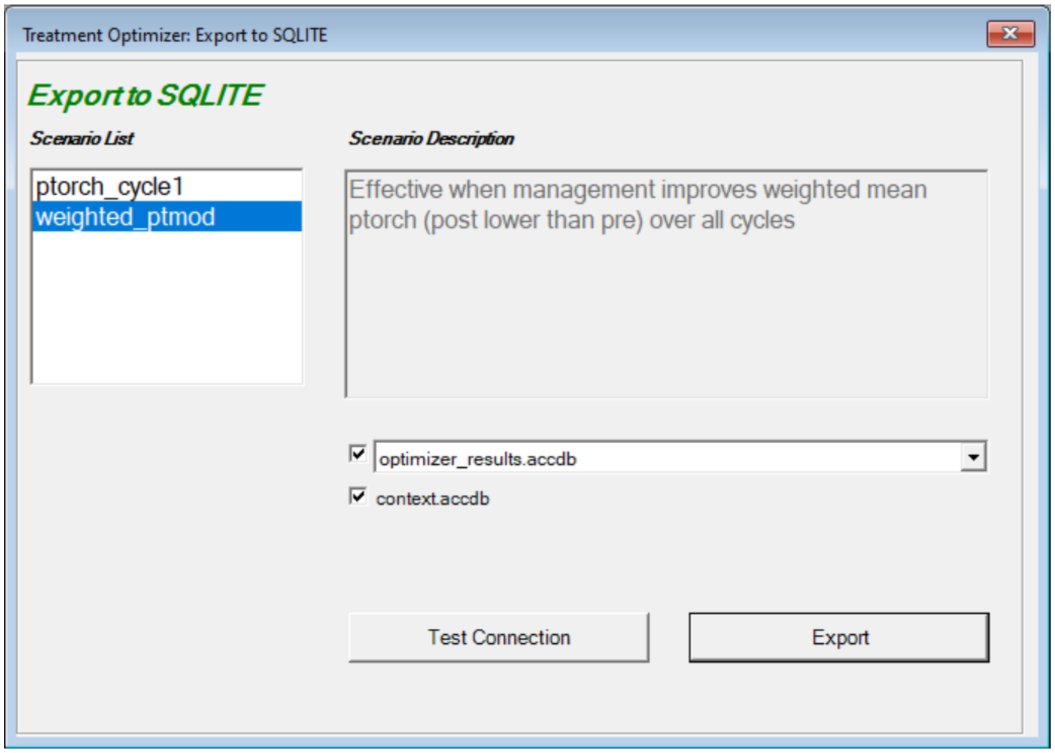
userdefinedcondfilter (Read Or | 1/1464 | 1464

Figure 6.24: The Results Tables viewer

3. <**Microsoft Access**>: Launches Microsoft Access on your computer and opens the scenario_results.mdb, which contains the same tables displayed in the viewer.
4. <**View Audit Data**>: Review the MS Access Treatment Optimizer audit tables from within FIA Biosum
5. <**View Log File**>: Open, in a viewer, the optimization scenario log file (runlog.txt). This log is the place to start troubleshooting software exceptions and anomalies experienced while running an optimization scenario.

Export to SQLite

The OKPyT (Olaf Kuegler Python Based Tabler) Python script offers the ability to "table" BioSum outputs in formatted Excel tables that display means and totals by user-specified groups, accompanied by estimates of the sampling error associated with the means and totals in every table cell. OKPyT relies on databases stored as SQLite, so a utility is provided in Optimizer to export BioSum outputs from the Access database to which they have been written to an SQLite database. Templates for BioSum output tables that we have found to be useful are still being developed at this time, and documentation of how to run the script (which is provided as an executable program) with these BioSum export files will be provided in a future BioSum release. For now, the export capability may be useful for users who prefer to store and work with data in SQLite rather than MS Access.
When you click the <**Export to SQLite**> button, the Export to SQLite screen will appear:



ptorch_cycle1
weighted_ptmod

Effective when management improves weighted mean ptorch (post lower than pre) over all cycles

☑ optimizer_results.accdb
☑ context.accdb

Figure 6.25: Export to SQLite selection dialog

The Export to SQLite selection dialog has the following fields:

1. **Scenario List**: This is the list of all the optimization scenarios available in the project. When a scenario in this list is selected (highlighted), the Scenario Description updates to reflect the analyst’s description of the selected scenario.
2. **Database to Export**: The name of the Access database containing the optimization output to be exported to an SQLite database, optionally including the context database (context.accdb) if its box is checked.

   Typically, optimizer_results.accdb will hold the most recently completed optimization for the scenario, but earlier simulations are stored in the same folder, with a simulation date and time stamp comprising part of the database file name and there may be times when it is desirable to export one of these instead. However, note that it is the responsibility of the analyst to ensure that the context database is current and consistent with the optimizer_results output. For example, it is possible that the context database in the scenario folder contains data from an earlier simulation and that some of the simulation parameters have been changed such that some of the context information has changed and does not reflect the most recent optimization output for the scenario. For this reason, the best practice is to create a context database during any optimization that you may wish to export to an SQLite database so that context and output are consistent.

Two buttons at the bottom of the dialog provide for testing that an SQLite ODBC driver is installed and functioning (**<Test Connection>**) and initiating the database export (**<Export>**). The export process is quite slow and users should be prepared for it to take 15-25 minutes, depending on whether the context database is also to be exported.

Chapter 7: Treatment Optimizer: Analysis Database and Its Derivation

Introduction

Chapter 6 was designed as a "how-to" for specifying and executing a treatment optimization. This chapter describes the output generated by that optimization, documents the data sources used by this module and internal module workflow, and offers some examples of how optimizer output and input can be linked to produce information useful for decision support. As of 29 January 2022, this chapter is incomplete and in revision, but may nonetheless be helpful. Through page 9, this was a draft of a chapter being built to document the outputs of earlier versions of BioSum, this some of the table names and fields may have changed between how they are referred to here and what you will see in the output databases. Beginning on page 10, there is documentation of the content of all tables in the optimizer_results database, but with little narrative about the organization and linkage among the tables in this database. Beginning on page 42 is documentation of the table structure of what we are now calling a context database, containing key inputs that go with the outputs and which an analyst might wish to link to the outputs when preparing to summarize or present a BioSum analysis. We anticipate fleshing out and QAing this chapter as soon as time can be found, and are including it in this very rough form only in the hope that it may be useful to some users. Please don't hesitate to ask questions about anything that is unclear.

Optimizer Outputs

Executing optimizer creates (or refreshes) the database optimizer_results.accdb in the folder bearing the optimization scenario name. This database contains 17 tables, several of which contain a core set of stand-level results for the scenario that can be summarized for the forested landscape or subsets thereof. Others exist mainly to support the workflow that produces that core set. Both stand-level and summarized results have been found useful for many purposes; however valid interpretation of these tables, and the views and queries constructed using them, requires a clear understanding of their derivation, as some of the embedded assumptions are subtle yet impactful. The most important objective of this chapter is to facilitate that understanding.

Tables in the core set must often be linked with one another and/or with data sources used by Optimizer, to generate complete analyses that can be handed off to decision-makers or used to inform policy analysis. Optimization builds towards a "final answer" of which treatment is best for each forested condition (stand); this final answer also serves as a convenient point of entry for describing this core table set is that end-point. We can then work backward through the intermediate result tables that inform the optimization, and ultimately establish back-links to many of these intermediate results tables and model inputs to synthesize the desired analysis. Considered broadly, there are 5 types of stand-level, BioSum output data found in the optimizer_results.accdb:

Optimal Treatment Assignment

all_cycles_best_rx_summary contains one row for every condition (stand) for which an effective treatment exists, with the identity of the optimal treatment and the values of the optimization and tiebreaker attributes. For scenarios that do not rely on any weighted variables, this table is named cycle1_best_rx_summary instead. all_cycles_optimization contains the name of the analyst-selected optimization attribute and its cycle 1 value pre- and post-treatment or, if a weighted attribute, the weighted multi-cycle combination of that attribute for the base (e.g., grow-only) and treatment scenarios. For scenarios that do not rely on any weighted variables, this table is named cycle1_optimization, instead.

Effectiveness Determination

all_cycles_effective (cycle1_effective in scenarios that do not rely on any weighted variables) contains net revenue and, for up to 4 analyst selected attributes, the cycle 1 pre- and post-values (or multi-cycle base and treatment values, for attributes where the weighted option is deployed), attribute change (pre to post, or base to treatment), and better, worse and effective yes/no determinations. It also contains an overall effectiveness determination according to logic specified by the analyst and coded in tables within \[project_name]\optimizer\db\scenario_optimizer_rule_definitions.mdb in tables like scenario_fvs_variables, scenario_fvs_variables_optimization, scenario_fvs_variables_overall_effective, etc.). Especially when multiple scenarios have been developed, it can be useful to establish backlinks to these "rule definitions" to pair them with optimization output to ensure that assumptions are transparently conveyed.

Haul Cost

haul_costs typically contains two records for every plot (note that there may be multiple stands per plot when there is more than one FIA condition on the plot)—one each for merchantable wood (M) and chipped biomass (C) for the unit haul cost, in dollars per green ton, for moving such wood from all stands represented on that plot to the facility with the lowest cost of access from that plot. These are calculated by optimizer from databases in the [project_name]\gis folder, and are useful for understanding where gaps in the processing facility network may elevate haul costs enough to affect feasibility of utilizing harvested material.

Yield, Value and Onsite Treatment Cost

Econ_by_rx_cycle contains, for each stand, silvicultural sequence (whether optimal or not), and cycle in which treatment occurs, the stand level data on all of the economic consequences of treatment, including per acre volume, biomass and value of harvested merchantable and non-merchantable wood biomass, along with haul and onsite treatment costs and net revenue obtainable from treatment (which may take on negative values when costs exceed the value of wood delivered). This table also reports (1) whether nonmerchantable biomass could be utilized (if not, owing to excessive unit haul cost relative to unit value, the ACD— Air Curtain Destruction flag is set to Y and the chip biomass shows up in the chip_acd_wt field), (2) the processing facility to which merchantable wood is delivered and (3) the processing facility to which chipped, nonmerchantable wood biomass is delivered. ADD DOCUMENTATION OF HOW NET REV CALCULATED WHEN CHIPS ARE AND ARE NOT USED.

Stand Level Inputs

The 5th type of data is "output" only in the sense that in the course of analysis, inputs get filtered. These tables, RULEDEFINITIONSCONDFILTER and RULEDEFINITIONSPLOTFILTER are filtered versions of the COND and PLOT tables where what is filtered out depends on the user specifications (on cond and plot) when optimizer was run and the user specified owner group filters (i.e., any owner groups with unchecked boxes are dropped). These tables, which can be thought of and used in queries interchangeably with their counterparts (PLOT and COND) in the Master database except that these filtered versions contain only those records used in the analysis. For example, these prove useful when analyzing stands with respect to site class, initial volume or stand density or for filtering on stands within a specified distance from the nearest road.

BioSum analyses also generate a wealth of information via tasks earlier in the work flow that does not end up in the optimizer_results database. Prominent examples include the specification of prescriptions and silvicultural sequences, Forest Status

The pre-post tables in the [project_name]\fvs\db folder contain a wealth of information, derived from FVS output, about each stand under each silvicultural sequence at each time point in the simulation

The latter are stored in the scenario_results.mdb database, within the Project**core***scenario_name*\db directory. Do <u>not</u> confuse this scenario_results.mdb database with the scenario_results.mdb database contained within the project**processor***scenario_name*\db directory—these files are not at all similar and are generated for completely different purposes. Some of the tables described here are intermediate in nature, in that they are produced by Core Analysis for use by Core Analysis in subsequent calculations. Some of these will rarely be of interest to the user, but occasionally be essential for debugging and analysis or for trying to understand results that are different than expected.

Table descriptions and process flow are organized into 8 sections, each covering a different set of related tables: Core Analysis Data Sources, Filters, Value, Costs, Valid Combinations, Effectiveness, Net Revenue Optimal, and Landscape Economics. The reader may wish to consult figure XX (Data Flow Diagram) as a roadmap through this narrative.

All references to data tables that are not within project_name/core/scenario_name/db/scenario_results.mdb rely on database.table syntax. All references to data items (referenced interchangeably as variables, attributes, and columns) use the database.table.column syntax, where strings consisting of 'X's represent 'wildcards' that would be replaced with locally appropriate strings. For example, XX_TREE_CUTLIST.FVSTREE.drybiot generically references bone dry total tree biomass in pounds for a tree in FVS variant region XX, where XX could be replaced with WS for Western Sierra or NC for Nothern California. Where file, table or column names contain spaces, this User Guide replaces spaces with underscores to facilitate distinguishing names from adjacent text and avoid confusion (e.g., GIS_TRAVEL_TIMES.TRAVEL_TIME.biosum_plot_id rather than GIS_TRAVEL_TIMES. TRAVEL TIME.biosum_plot_id). It is assumed and implied that all database

files are in MS Access format with the extension .MDB or .ACCDB; the former may be internally formatted as MDB or ACCDB files.
Note that this version of the documentation covers only the primary tables that Core Analysis relies on for source data (these are spread across a few different databases) and the output tables in scenario_results.mdb, which is located within project_name/core/scenario_name/db. It does NOT include the 34 temporary work tables generated during a core analysis run—these are stored, along with several dozen links to tables of BioSum input and output data in other databases, temporarily, until the user exits the BioSum software, in (under Windows 7) the c:\user\user_name\AppData\Local\Temp\1 folder with a filename in the format of fia_biosum_XXXX.mdb, where XXXX is an apparently random number up to 4 digits in length and the file size is large (tens of megabytes). Many other temporary Access files (with .accdb file name extensions) appear in this same directory, but are small (kilobytes, not megabytes) and uninteresting. To preserve fia_biosum_XXXX.mdb for future inspection or analysis, copy it to another location and/or name BEFORE exiting the BioSum software.

Core Analysis Data Sources

A valid Core Analysis requires prior completion of all the other BioSum components in their entirety: Database, FVS and Processor and the geoprocessing protocols that populate records in GIS_TRAVEL_TIMES.TRAVEL_TIME. Upon selecting the rules tab in Core Analysis, this module generates a large temporary database filled with work tables that will be populated during the course of the core analysis run, links to various source tables and links to output tables in the scenario_results database that will be populated by the end of the run (see Introduction, above). The key inputs for each stand/package combination, for each cycle, comes from four sources: 1) project_name/processor/scenario_name/scenario_results.harvest_costs, which contains, the on-site harvest costs simulated by OpCost and the supplemental per acre costs assigned by the analyst; 2) project_name/processor/scenario_name/scenario_results.tree_vol_val_by_species_diameter_groups, which contains the volume, biomass and value produced, by species and diameter group; 3) project_name/gis/gis_travel_times.travel_time, which contains the travel time from each plot to each processing site; and 4) project_name/fvs/db/biosum_fvsout_prepost_rx, which contains links to PRE and POST treatment versions of FVS output tables, such as FVS_Summary, FVS_StrClass, FVS_PotFire, FVS_Carbon, and FVS_Hrv_Carbon and FVS_Compute.
Note that any stand-level attribute can be used in core analysis as a basis for effectiveness rating or for choosing among multiple effective alternatives. Those attributes that are NOT generated via FVS—for example, attributes in the master.cond (condition) table or hazard scores that the user estimates, or calculates, outside of FVS, for example as a combination of FVS attributes—can be either 1) attached as additional columns to one of the existing FVS output tables linked into biosum_fvsout_prepost_rx or 2) by defining two new tables in biosum_fvsout_prepost_rx named PRE_FVS_XXXXXXXXXX and POST_FVS_XXXXXXXXXX, where the 'X's can be any desired name, provided that a) the following columns are included in these new tables: biosum_cond_id, rxpackage, rx, rxcycle, fvs_variant, CaseID, StandID plus whatever attributes are to be used in the core analysis and b) the all cases (combinations of the

aforementioned columns) that appear in the PRE_FVS_SUMMARY and POST_FVS_SUMMARY table also appear in the respective 'XXXXXXXXXX' table.

Processor Source 1: scenario_results.harvest_costs

This table is produced by the OpCost harvest cost calculator, an R script that is called from BioSum's Processor module, to estimate harvest costs, including move-in costs, specific to the analyst requested harvest system (specified either in the prescription or in the Processor scenario), cut list, and site factors associated with each stand.

Processor Source 2: scenario_results.tree_vol_val_by_species_diameter_groups

The second major intermediate output of the Processor module is a table containing, for all silvicultural sequences and stands in the project, estimates of the quantities and value of wood produced, by diameter class and species group, at each cycle (time of potential activity) of the simulation. Each row of the table contains a native key (ID), and the following foreign keys: stand identifier (biosum_cond_id), silvicultural sequence (RxPackage), prescription (Rx), and Cycle (RxCycle) for which the estimates are made, along with diameter group (Diam_Group) and species group (Species_Group). The per-acre estimates that appear here are for chip (a.k.a. energy wood) and merchantable volume (cubic feet), biomass (green tons), and value (dollars) along with "brush cut" volume and biomass that is cut but not removed from the forest. The merch_To_ChipBin_YN holds the users choice (Yes or No), made when defining value per species and diameter class by checking a box, or not, telling Processor whether or not to redirect wood from that species and diameter class to use as chips (e.g., for bioenergy or biochar), in which case the values shown for merchantable wood will be zero because the entirety of such trees that are removed is assumed to be processed as chips (case 2 in the example below).

Id	1	2
biosum_cond_id	1201141020103300712140002	1201206060204500710100002
Rxpackage	8	8
Rx	800	800
rxcycle	1	1
species_group	3	4
diam_group	5	4
chip_vol_cf	6.506042	1767.528
chip_wt_gt	0.145722	44.03141
chip_val_dpa	1.821527	550.3926
chip_mkt_val_pgt	12.5	12.5
merch_vol_cf	23.88637	0
merch_wt_gt	0.595057	0
merch_val_dpa	79.78047	0
merch_to_chipbin_YN	N	Y
bc_wt_gt		
bc_vol_cf		

FVS Output Sources: the fvsout_prepost databases

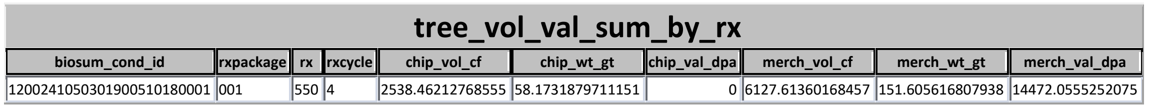
When FVS Output data is loaded into BioSum, all tables within Project\FVS\Data\XX\FVSOUT_XX_PYYY-ZZZ-ZZZ-ZZZ-ZZZ.MDB (where XX is the variant code, YYY is the RxPackage, and ZZZ represents the prescription number) EXCEPT for FVS_ATRT_List, FVS_Cases, FVS_CutList, and FVS_TreeList are processed, based on user defined SEQNUM choices, to pull the PRE and POST treatment values into separate PRE_FVS_* and POST_FVS_* tables (where * represents a table name such as SUMMARY) in Project\FVS\DB\PREPOST_FVS_*.ACCDB. Then, BioSum constructs a biosum_fvsout_prepost_rx.mdb database that links to each of these ACCDB prepost databases. The non-key attributes within these tables, and any other compatible tables that the user decides to build in biosum_fvsout_prepost_rx.mdb, are available in core analysis to determine the effectiveness of each silvicultural sequence, the best (most optimal) sequence when more than one is effective, and to break ties so as to achieve a single optimal sequence for each stand.

Filters

Somewhat counter-intuitively, user-defined plot and condition filters are applied AFTER the package outcomes have been calculated and best packages identified. They are designed to filter out from consideration for management, stands (plot-condition combinations) that don't meet user specified criteria for plots and conditions. The filters specified by the user as SQL WHERE clauses in the rules section of Core Analysis for plots and conditions generate the tables userdefinedplotfilter and userdefinedcondfilter, respectively. These are simply records from the PLOT and COND tables that meet the WHERE clause requirements. The tables ruledefinitionsplotfilter and ruledefinitionscondfilter are generated by ???LARRY???. AND HOW DO THESE INTERACT OR KNOCK OUT CONDITIONS FROM CONISDERATION?

Value and Volume

Tree_vol_val_sum_by_rx and Tree_vol_val_sum_by_rxpackage are derived from the processor output (stored in project\processor\scenario_name\db\scenario_results.tree_vol_val_by_spp_diam_class) produced by processor using the fvs_tree tables (in the variant-specific cutlist databases associated with the project, e.g., project \fvs\data\variant\BiosumCalc\XX_PYYY_TREE_CUTLIST.FVS_TREE) using the assumptions specified for tree species groups and diameter classes and prices (of merchantable and energy wood). In the example row below from tree_vol_val_sum_by_rx, chip_wt_gt is the tpa weighted sum of drybiot (for species diameter class combinations that are specified in Processor as allocated to energy wood) or (drybiot-drybiom) for all others (provided that whole tree harvest systems are utilized), adjusted by the dry to green ratio factor from the tree_species table in ref_master, for all trees harvested in this biosum_cond_id for package 1 in cycle 4, in which prescription 550 is applied. Records in Tree_vol_val_sum_by_rxpackage are calculated as the sum, for each biosum_cond_id and package, of all matching records in Tree_vol_val_sum_by_rx (i.e., summed for all cycles).



tree_vol_val_sum_by_rx

biosum_cond_id	rxpackage	rx	rxcycle	chip_vol_cf	chip_wt_gt	chip_val_dpa	merch_vol_cf	merch_wt_gt	merch_val_dpa
1200241050301900510180001	001	550	4	2538.46212768555	58.1731879711151	0	6127.61360168457	151.605616807938	14472.0555252075

Costs

Harvest and haul costs per acre, and the same volume and value information available in tree_vol_val_sum_by_rx, plus net revenue per acre (gross revenue minus harvest and haul costs) can be found in product_yields_net_rev_cost_summary_rx. The harvest cost information flows from harvest_costs and the haul costs come from travel_time and the truck and driver haul cost per green ton per hour parameter entered in the core analysis scenario. As with tree_vol_val_sum_by_rxpackage, the cost information is summed across cycles to populate product_yields_net_rev_cost_summary_rxpackage, which contains costs, volume and values summed for the four cycles in a package for each stand-package combination.

Valid Combinations

The term ValidCombos refers to combinations of stand ID (condition) and package that are determined to be valid from one or more perspectives. Validcombos_fvspre contains combinations of Biosum_conditionID, RxPackage, rx, and rxcycle along with Y/N values for 4 variables relating to the pre-treatment values at each odd-numbered cycle (I am presuming these have to do with the effectiveness criteria specified in core analysis, but can't really yet pin down exactly what the Y and N values reference—LARRY can you elaborate?). Validcombos_fvspost provides the same info for the post-treatment time points (even numbered cycles if running an 8 cycle projection). These tables have the same number of rows. Validcombos_fvsprepost has the records from these tables for which the values of the 4 variables are Y. validcombos is a subset of validcombos_fvsprepost for which there are matching (on BioSum_condtionID and RxPackage) records in the product_yields_net_revenue_costs_summary_by_rx (and …_rxpackage) tables AND matching records in the tree_vol_val_sum_by_rx (and …_rxpackage) tables. Rows must also match on the userdefinedcondfilter and userdefinedplotfilter tables to be validcombos. A subset of these makes it in to the cycle1_effective table (which contains only results for cycle 1). I am not sure what causes records on validcombos to reject and not end up in cycle1_effective. Some of the rows in cycle1_effective will have overall_effective_yn set to "Y" (if they pass the criteria specified in core analysis) and only these are considered for the cycle1_best tables—LARRY CAN YOU EXPLAIN?.

Effectiveness

Effectiveness is considered only with respect to pre- and post-treatment attributes at cycle 1 (i.e., looking at year 0 and year 1, with no FVS simulation beyond year 1). Effectiveness criteria are specified via the Core Analysis interface for up to 4 attributes and an overall effectiveness that depends on any one or a combination of the 4 effectiveness determinations or values or changes in values of any or a combination of the 4 attributes. All packages with a cycle 1 activity will appear listed in cycle1_effective with the overall_effective_yn attribute indicating whether or not that package will be considered further (i.e., only if 'Y'). In this example, only packages 1 and 4 are overall effective:

biosum_cond_id	1	1	1	1	1	1
rxpackage	001	002	003	004	005	006
rx	001	002	003	001	002	003
rxcycle	1	1	1	1	1	1
nr_dpa	-89558	-143188	-194333	-89558	-143188	-194333
pre_variable1_name	PRE_FVS_PotFire.PTorch_Sev					
post_variable1_name	POST_FVS_PotFire.PTorch_Sev					
pre_variable1_value	0.98	0.97	0.98	0.98	0.97	0.98
post_variable1_value	0.97	0.99	0.99	0.97	0.99	0.99
variable1_change	-0.01	0.02	0.00	-0.01	0.02	0.00
variable1_better_yn	Y	N	N	Y	N	N
variable1_worse_yn	N	Y	Y	N	Y	Y
variable1_effective_yn	Y	N	N	Y	N	N
pre_variable2_name	PRE_FVS_PotFire.Surf_Flame_Sev					
post_variable2_name	POST_FVS_PotFire.Surf_Flame_Sev					
pre_variable2_value	3.34	3.65	3.89	3.34	3.65	3.89
post_variable2_value	3.57	3.74	3.88	3.57	3.74	3.88
variable2_change	0.23	0.08	-0.01	0.23	0.08	-0.01
variable2_better_yn	N	N	N	N	N	N
variable2_worse_yn	N	N	Y	N	N	Y
variable2_effective_yn	Y	N	N	Y	N	N
pre_variable3_name						
....	variables 3 and 4 not shown for brevity					
variable4_effective_yn						
overall_effective_yn	Y	N	N	Y	N	N

Effectiveness determination criteria entered into the Core Analysis interface are stored in project_name/core/db/scenario_core_rule_definitions.mdb in tables such as scenario_fvs_variables_overall_effective and scenario_fvs_variables.
Cycle1_effective contains all stand-Rx (cycle 1) combinations and whether or not the treatment was effective by user-defined criteria (overall_effective). The variable(s) used to determine effectiveness is/are also included and their values pre and post.

Net Revenue Optimal

The tables that begin Cycle1_MaxNR_ are intended to contain results for the sequence that maximizes net revenue at cycle 1 (which may nonetheless be negative). Only costs and revenues incurred in cycle 1 are considered. The table name that ends in "stands" gives stand level values for the max NR sequence for net revenue per acre, but I suspect that the rest of the attributes, while labeled as pa (per acre) actually contain the expanded values (per acre times the acres represented by the stand). Own_sum contains results (totals) by owner group and psites_sum contains results (totals) by psite. Own_air_dest contains totals for plots too far from any psite (such that their wood had to be destroyed in an air curtain destructor).
Stands_air_dest shows which stands these were.
Cycle1_best_rx summary contains the best sequence (rx package) for each stand, of those deemed effective, by the user defined criteria, along with the value of the optimized and tie breaker attributes.

Landscape Economics

4 tables summarize the economics for all stand-package combos. product_yields_net_rev_costs_summary_by_rx and product_yields_net_rev_costs_summary_by_rxpackage contain per acre values for volumes, biomass and value and harvest and haul costs by cycle (by_rx) and summed over all cycles (by_rxpackage). stand_costs_revenue_volume_by_rx and stand_costs_revenue_volume_by_rxpacakge contain the expanded values for each stand (per acre values times the acres the stand represents in the larger landscape).

Treatment Optimizer Table Definitions

PLOT table

Location: db\master.mdb. This table is created when plots are loaded.

	Column name	Descriptive name	Data type
1	BIOSUM_PLOT_ID (Primary key)		Text(24)
2	STATECD	State code	Integer
3	INVYR	Inventory year	Integer
4	UNITCD	Survey unit code	Integer
5	COUNTYCD	County code	Integer
6	PLOT	An identifier for a plot	Integer
7	MEASYEAR	Measurement year	Integer
8	MEASMON	Measurement month	Integer
9	MEASDAY	Measurement day	Integer
10	ELEV	Elevation	Integer
11	FVS_VARIANT	FVS variant	Text(2)
12	FVSLOCCODE	FVS location code	Integer
13	HALF_STATE	Half state (OR/WA only)	Text(10)
14	SUBPLOT_COUNT_PLOT	Total number of subplots or proportions of subplots	Byte
15	GIS_YARD_DIST_FT	Yarding distance from plot to the nearest road in feet.	Double
16	NUM_COND	Number of conditions on the plot	Byte
17	ONE_COND_YN	One condition on the plot?	Text(1)
18	LAT	Latitude	Double
19	LON	Longitude	Double
20	MACRO_BREAKPOINT_DIA	Macroplot breakpoint diameter	Integer
21	BIOSUM_STATUS_CD		Byte
22	CN	Sequence number	Text(34)

1. BIOSUM_PLOT_ID: Unique plot id field. Used as the primary key for the plot table.

Column	Length	Description	Values
01	01	Inventory Id: As of v5.8.6 only FIADB is supported	"1" = FIADB
02	04	Secondary Inventory id: Identify the specific inventory	The year the plot is measured or "9999" = Inventory Unknown
06	02	State code	
08	02	Nims Cycle	"00"= PNW IDB
10	02	Nims Subcycle	"00 = PNW IDB
12	03	County Code	
15	07	Plot number	
22	03	PNW IDB forest or BLM district	"000" = FIADB

2. STATECD: Bureau of the Census Federal Information Processing Standards (FIPS) two-digit code for each State.

3. INVYR: The year when the inventory data were scheduled to be collected. INVYR is often (but not necessarily) the same as MEASYEAR, which is the year when the plot was actually visited and measured.
4. UNITCD: Forest Inventory and Analysis survey unit identification number. Survey units are usually groups of counties within each State.
5. COUNTYCD: The identification number for a county, parish, watershed, borough, or similar governmental unit in a State. FIPS codes from the Bureau of the Census are used.
6. PLOT: Phase 2 public plot number. For PNW states, the combination of INVYR, STATECD and PLOT will uniquely identify a plot record in the database. It is usually more convenient to use CN (see above) to identify unique plots in the inventory (called PLT_CN in other tables).
7. MEASYR: The year in which the plot was completed. MEASYEAR may differ from INVYR.
8. MEASMON: The month in which the plot was completed where January is 01
9. MEASDAY: The day of the month in which the plot was completed
10. ELEV: The distance the plot is located above sea level, recorded in feet (NAD 83). Negative values indicate distance below sea level. Elevation is stored to the nearest 100 feet.
11. FVS_VARIANT: A growth and mortality model calibrated to a specific geographic area of the United States. There are 20 different FVS variants.
12. FVSLOCCODE: The location code is a 3-digit code where, in general, the first digit of the code represents the USDA Forest Service Region Number, and the last two digits represent the Forest Number within that region. In some cases, a location code beginning with a "7" or "8" is used to indicate an administrative boundary that doesn't use a Forest Service Region number (according to the 2015 PNW-FIADB Users Manual) .
13. HALF_STATE: Code for East or West of the Cascades (Oregon and Washington only).
14. SUBPLOT_COUNT_PLOT: A count of the total number of subplots or proportions of subplots installed on the plot, regardless of condition class.
15. GIS_YARD_DIST_FT:
16. NUM_COND:
17. ONE_COND_YN: "Y" = Only one condition exists for the plot, "N" = Multiple conditions on the plot.
18. LAT: The approximate latitude of the plot in decimal degrees using NAD 83 datum.
19. LON: The approximate longitude of the plot in decimal degrees using NAD 83 datum.
20. MACRO_BREAKPOINT_DIA: A macroplot breakpoint diameter is the diameter (either DBH or DRC) above which trees are measured on the plot extending from 0.01 to 58.9 feet horizontal distance from the center of each subplot. If macroplots are not installed, this value will be null.
21. BIOSUM_STATUS_CD: For internal use only. Reports the state of BioSum data. During plot import, rows that have been inserted have a status code of 9. If import completes successfully this field is set to 1.

22. CN: A unique number that identifies every record in the PLOT table. This column appears as PLT_CN in other FIA database tables and is one of the key columns you will use to link to most other tables.

Note: Many of these field definitions are derived from the PNW-FIADB Users Manual published in August 2015.

COND table

Location: db\master.mdb. This table is created when plots are loaded.

	Column name	Descriptive name	Data type
1	BIOSUM_COND_ID (Primary key)		Text(25)
2	BIOSUM_PLOT_ID (Foreign key)	See PLOT table	Text(24)
3	INVYR	Inventory year	Integer
4	CONDID	Condition class number	Byte
5	CONDPROP	Condition proportion	Double
6	LANDCLCD	Land class code	Byte
7	FORTYPCD	Forest type code	Integer
8	GROUND_LAND_CLASS_PNW	Present ground land class	Text(3)
9	OWNCD	Owner class code	Integer
10	OWNGRPCD	Owner group code	Integer
11	RESERVCD	Reserved status code	Byte
12	SITECLCD	Site productivity class code	Byte
13	SIBASE	Site index base age	Integer
14	SICOND	Site index for the condition	Integer
15	SISP	Site index species code	Integer
16	SLOPE	Slope	Integer
17	ASPECT	Aspect	Integer
18	STDAGE	Stand age	Integer
19	STDSZCD	Stand-size class code	Byte
20	HABTYPCD1	Habitat type code 1	Text(10)
21	ADFORCD	Administrative forest code	Integer
22	QMD_TOT_CM	Quadratic mean diameter for both hardwoods and softwoods	Single
23	HWD_QMD_TOT_CM	Quadratic mean diameter for hardwoods	Single
24	SWD_QMD_TOT_CM	Quadratic mean diameter for softwoods	Single
25	ACRES	Acres	Double
26	UNITCD	Unit code	Integer
27	VOL_LOC_GRP	Volume location group	Text(10)
28	TPACURR	Trees per acre	Double
29	HWD_TPACURR	Hardwood trees per acre	Double
30	SWD_TPACURR	Softwood trees per acre	Double
31	BA_FT2_AC	Basal area per acre	Double
32	HWD_BA_FT2_AC	Basal area per acre (hardwoods)	Double
33	SWD_BA_FT2_AC	Basal area per acre (softwoods)	Double
34	VOL_AC_GRS_STEM_TTL_FT3	Gross stem volume per acre	Double
35	HWD_VOL_AC_GRS_STEM_TTL_FT3	Gross stem volume per acre (hardwoods)	Double
36	SWD_VOL_AC_GRS_STEM_TTL_FT3	Gross stem volume per acre (softwoods)	Double
37	VOL_AC_GRS_FT3	Gross volume per acre	Double
38	HWD_VOL_AC_GRS_FT3	Gross volume per acre (hardwoods)	Double
39	SWD_VOL_AC_GRS_FT3	Gross volume per acre (softwoods)	Double
40	VOLCSGRS	Gross sawlog volume	Double
41	HWD_VOLCSGRS	Gross sawlog volume (hardwoods)	Double
42	SWD_VOLCSGRS	Gross sawlog volume (softwoods)	Double

43	GSSTKCD	Growing-stock stocking code	Double
44	ALSTKCD	All live stocking code	Double
48	CONDPROP_UNADJ	Condition proportion unadjusted	Double
49	MICRPROP_UNADJ	Microplot proportion unadjusted	Double
50	SUBPPROP_UNADJ	Subplot proportion unadjusted	Double
51	MACRPROP_UNADJ	Microplot proportion unadjusted	Double
53	CN	Condition sequence number	Text(34)
54	BIOSUM_STATUS_CD		Byte
55	DWM_FUELBED_TYPCD	DWM condition fuelbed type code	Text(3)
56	MODEL_YN	Reserved for future use	Text(1)

1. BIOSUM_COND_ID: Unique condition id field. Used as the primary key for the cond table.

Column	Length	Description	Values
01	01	Inventory Id: As of v5.8.6 only FIADB is supported	"1" = FIADB
02	03	Secondary Inventory id: Identify the specific inventory	The year the plot is measured
06	02	State code	
08	02	Nims Cycle	"00"= PNW IDB
10	02	Nims Subcycle	"00 = PNW IDB
12	03	County Code	
15	07	Plot number	
22	03	PNW idb forest or blm district	"000" = FIADB
25	01	Condition Number	1 to 5

2. BIOSUM_PLOT_ID: See PLOT table
3. INVYR: The year when the inventory data were scheduled to be collected. INVYR is often (but not necessarily) the same as MEASYEAR, which is the year when the plot was actually visited and measured.
4. CONDID: Unique identifying number assigned to each condition on a plot. A condition is initially defined by condition class status. Differences in reserved status, owner group, forest type, stand-size class, regeneration status, and stand density further define condition for forest land. Mapped nonforest conditions are also assigned numbers. On a plot, each sampled condition class must have a unique number that can change at remeasurement to reflect new conditions on the plot. Use the combination of PLT_CN and CONDID to link to other tables (i.e. TREE table).
5. CONDPROP: Percentage of the proportional weight of the condition on the plot
6. LANDCLCD: A code indicating the basic land cover. 1 = Accessible Forest Land, 2 = Nonforest Land, 3 = Non Census Water, 4 = Census Water, 5 = Nonsampled
7. FORTYPCD: Codes for National Forest or BLM. Represents the dominant tree species in a condition class

8. GROUND_LAND_CLASS_PNW: A refinement of forest land that distinguishes timberland and a variety of forest land types. Each code, and corresponding ground land class (GLC) name and description are listed. See the PNW FIADB manual for a list of codes and classes.
9. OWNCD: A code indicating the class in which the landowner (at the time of the inventory) belongs.
10. OWNGRPCD: A broader group of landowner classes.
11. RESERVCD: Reserved land is land that is withdrawn by law(s) prohibiting the management of the land for the production of wood products. Timberland is always RESERVCD=0.
12. SITECLCD: A classification of forest land in terms of inherent capacity to grow crops of industrial wood.
13. SIBASE: The base age (sometimes called reference age), in years, of the site index curve used to derive site index. This attribute is blank (null) when no site tree data are available.
14. SICOND: This represents the average total length in feet that dominant and co-dominant trees are expected to attain in well-stocked, even-aged stands at the specified base age (SIBASE This attribute is blank (null) when no site index data are available.
15. SISP: The species upon which the site index is based. This attribute is blank (null) when no site tree data are available.
16. SLOPE: The angle of slope, in percent, of the condition. Valid values are 000 through 155 .
17. ASPECT: The direction of slope, to the nearest degree, for most of the condition. North is recorded as 360.
18. STDAGE: Stand age.
19. STDSZCD: A classification of the predominant (based on stocking) diameter class of live trees within the condition assigned using an algorithm. See the PNW FIADB manual for a list of codes and classes.
20. HABTYPCD1: A code indicating the primary habitat type (or community type) for this condition.
21. ADFORCD: Identifies the administrative unit (Forest Service Region and National Forest) in which the condition is located. The first two digits of the four digit code are for the region number and the last two digits are for the Administrative National Forest number.
22. QMD_TOT_CM: SQR(c.ba_ft2_ac/(.005454154 * c.tpacurr)) for all live trees with a dia >= 5 inches
23. HWD_QMD_TOT_CM: SQR(c.hwd_ba_ft2_ac/(.005454154 * c.hwd_tpacurr)) for all live trees with a dia >= 5 inches
24. SWD_QMD_TOT_CM: SQR(c.swd_ba_ft2_ac/(.005454154 * c.swd_tpacurr)) for all live trees with a dia >= 5 inches
25. ACRES: The number of acres the condition represents. (Plot.expall * cond.condprop). Used as the Expansion Factor.

26. UNITCD: FIA survey unit identification number. Survey units are usually groups of counties within each State.
27. VOL_LOC_GRP: An identifier indicating what equations are used for volume, biomass, site index, etc. A volume group is usually designated for a geographic area, such as a State, multiple States, a group of counties, or an ecoregion. See the PNW FIADB manual for a list of codes.
28. TPACURR: Sum of all the live trees TPACURR field for the condition with a diameter of >= 5 inches in diameter.
29. HWD_TPACURR: Sum of all the hardwood live trees TPACURR field for the condition with a diameter of >= 5 inches in diameter.
30. SWD_TPACURR: Sum of all the softwood live trees TPACURR field for the condition with a diameter of >= 5 inches in diameter.
31. BA_FT2_AC: The amount of basal area a tree represents per acre in the condition class on the plot, in square feet per acre. Sum of all the live trees BA_FT2_AC field for the condition: (SUM((.005454154 * dia^2) * tpacurr)).
32. HWD_BA_FT2_AC: The amount of basal area a tree represents per acre in the condition class in square feet per acre. Sum of all the hardwood live trees BA_FT2_AC field for the plot + condition: (SUM((.005454154 * dia^2) * tpacurr)).
33. SWD_BA_FT2_AC: The amount of basal area a tree represents per acre in the condition class in square feet per acre. Sum of all the softwood live trees BA_FT2_AC field for the plot + condition: (SUM((.005454154 * dia^2) * tpacurr)).
34. VOL_AC_GRS_STEM_TTL_FT3: (Drybiom/volcfgrs) gives the weighted ratio of dry merch volume to gross cubic foot volume. The total dry volume is then divided by the weighted ratio to get the total gross stem volume. The value is expanded to an acre by multiplying by tpacurr. Gross volume of the total stem is the volume of the entire tree (ground to tip), in cubic feet/acre. This volume is calculated for all live trees >= 1" (2.54cm) dbh. NOTE: FIADB has null values in drybiom and volcfgrs for trees with a diameter < 5 inches. Therefore, a mean constant value is used in the equation. Sum of all the live trees VOL_AC_GRS_STEM_TTL_FT3 field for the plot + condition.
35. HWD_VOL_AC_GRS_STEM_TTL_FT3: Sum of all the hardwood live trees VOL_AC_GRS_STEM_TTL_FT3 field for the condition.
36. SWD_VOL_AC_GRS_STEM_TTL_FT3: Sum of all the softwood live trees VOL_AC_GRS_STEM_TTL_FT3 field for the condition.
37. VOL_AC_GRS_FT3: Gross volume is the volume of a tree, from a 12" high stump to a 4" diameter top, in cubic feet/acre. This volume is calculated for all live trees >= 5" dbh and has not been reduced by the percent of sound or rotten cull recorded for the tree. SUM(volcfgrs * tpacurr) for a live tree with a diameter >= 5 inches.

38. HWD_VOL_AC_GRS_FT3: Sum of all the hardwood live trees VOL_AC_GRS_FT3 field for the condition.
39. SWD_VOL_AC_GRS_FT3: Sum of all the softwood live trees VOL_AC_GRS_FT3 field for the condition.
40. VOLCSGRS: Gross sawlog volume is the cubic volume of the sawlog portion of a tree, in cubic feet/acre. The sawlog is measured from the top of a 12" high stump to a 6-inch diameter top for softwoods and to an 8-inch diameter top for hardwoods. This volume is calculated on all live trees that are sawtimber sized as follows: softwoods >= 9" dbh or hardwoods >= 11" (SAWTIMBER_YN=Y). Sum of all the live trees VOLCSGRS field for the plot + condition
41. HWD_VOLCSGRS: Sum of all the hardwood live trees VOLCSGRS field for the condition.
42. SWD_VOLCSGRS: Sum of all the softwood live trees VOLCSGRS field for the condition.
43. GSSTKCD: A code indicating the stocking of the condition by growing-stock trees, including seedlings. See the PNW FIADB manual for a list of codes and definitions.
44. ALSTKCD: A code indicating the stocking of the condition by live trees, including seedlings. See the PNW FIADB manual for a list of codes and definitions.
45. CONDPROP_UNADJ: The unadjusted proportion of the plot that is in the condition. This variable is retained for ease of area calculations. It is equal to either SUBPPROP_UNADJ or MACRPROP_UNADJ, depending on the value of PROP_BASIS. The sum of all condition proportions for a plot = 1. When generating population area estimates, this proportion is adjusted by either the POP_STRATUM.ADJ_FACTOR_MACR or the ADJ_FACTOR_SUBP to account forpartially nonsampled plots (access denied or hazardous portions).
46. MICRPROP_UNADJ: The unadjusted proportion of the microplots that are in the condition. The sum of all microplot condition proportions for a plot = 1.
47. SUBPPROP_UNADJ: The unadjusted proportion of the subplots that are in the condition. The sum of all subplot condition proportions for a plot = 1.
48. MACRPROP_UNADJ: The unadjusted proportion of the macroplots that are in the condition. When macroplots are installed, the sum of all macroplotcondition proportions for a plot = 1; otherwise this attribute is left blank (null).
49. CN: A unique number used to identify a condition record. This will appear as CND_CN in related FIADB tables.
50. BIOSUM_STATUS_CD: Reports the state of BioSum data. During plot import, rows that have been inserted have a status code of 9. If import completes successfully this field is set to 1.
51. DWM_FUELBED_TYPCD: A code indicating the fuels available for consumption by fire. See the FIADB Database Description and User Guide for the codes and definitions.
52. MODEL_YN: Reserved for future use. Default = ‘Y’. If field is set to ‘N’, condition will not be sent to FVS for modeling.

Note: Many of these field definitions are derived from the PNW-FIADB Users Manual published in August 2015.

TRAVEL_TIME table

Location: db\gis\gis_travel_times.mdb. This table may be populated using the 'Load GIS Data' button in Treatment Optimizer to record the travel times between each plot and psite

	Column name	Descriptive name	Data type
1	TRAVELTIME_ID (Primary key)		Integer
2	PSITE_ID (Foreign key)	"From" processing site id	Integer
3	BIOSUM_PLOT_ID (Foreign key)	See PLOT table	Text(24)
4	COLLECTOR_ID (Foreign key)	"From" processing site id	Integer
5	RAILHEAD_ID (Foreign key)	"To" processing site id	Integer
6	TRAVEL_MODE	Road or railroad travel?	Byte
7	ONE_WAY_HOURS	One-way travel hours. Old name was TRAVEL_TIME.	Double
8	PLOT	PLOT.PLOT. Old name was PLOT_ID	Integer
9	STATECD	PLOT.STATECD. State code	Integer

1. TRAVELTIME_ID: Unique ID for records in this table.
2. PSITE_ID: Wood processing facility. Can be accessed by road, rail, or both. Represents the "FROM" side of the travel time, given that travel times are calculated along the road network TO each plot.
3. BIOSUM_PLOT_ID: Foreign key to link with records in plot and haul_costs tables. May be null for RAILHEAD_ID to COLLECTOR_ID records where TRAVEL_MODE = 2.
4. COLLECTOR_ID: A subcategory of psite (wood processing facility) with both road and rail access. Represents the "FROM" side of the travel time, given that travel times are calculated along the road network TO each plot or railhead.
5. RAILHEAD_ID: A psite that is a transfer site from road to rail. Represents the "FROM" side of the travel time when storing the travel time to the plot (TRAVEL_MODE = 1) and the "TO" side when storing the travel time from the railhead to the collector, along the rail network (TRAVEL_MODE = 2).
6. TRAVEL_MODE: "1" indicates Road Travel (plot to psite). "2" indicates Railroad Travel (railhead to collector site).
7. ONE_WAY_HOURS: Time to travel from plot to processing site or railhead (if TRAVEL_MODE=1) or from railhead to collector site (if TRAVEL_MODE=2)
8. PLOT: See PLOT table
9. STATECD: See PLOT table

Note: The travel_time from a plot to a collector site via a rail network requires two records in this table. The first is a road access record representing the time to travel from the plot to the railhead. The second is a rail access record with the one_way_hours, via the rail network, from the railhead to the collector.

PROCESSING_SITE table

Location: db\gis\gis_travel_times.mdb. This table may be populated using the 'Load GIS Data' button. Information for the psites identified in the TRAVEL_TIME table are added to this table.

Column name	Description	Data type
PSITE_ID (Primary key)		Integer
NAME		Text (100)
TRANCD		Integer
TRANCD_DEF		Text (30)
BIOCD		Integer
BIOCD_DEF		Text (15)
EXISTS_YN		Text (1)
LAT		Double
LON		Double
STATE		Text (2)
CITY		Text (40)
MILL_TYPE		Text (40)
COUNTY		Text (40)
STATUS		Text (40)
NOTES		Text (50)

1. PSITE_ID: Unique ID number for each "processing site", which may be an actual facility or a railhead
2. NAME: Facility or railhead name
3. TRANCD: 1 = Regular (Processing Site Only Accessible By Road), 2 = Railhead (Transfer site of biomass from truck to rail), 3 = Rail Collector (Processing Site Accessible By both Road And Rail)
4. TRANCD_DEF: "Regular", "Railhead", "Rail Collector"
5. BIOCD: 1 = Merchantable, 2 = Chips, 3 = Both
6. BIOCD_DEF: "Merchantable", "Chips", "Both"
7. EXISTS_YN: Whether or not the facility exists (Y for yes; N for no)
8. LAT: Latitude of facility
9. LONG: Longitude of facility
10. STATE: Postal code for state where facility is located. Example: OR
11. CITY: City where facility is located
12. MILL_TYPE: Mill type as stated in UMontana BBER's dataset
13. COUNTY: County where facility is located
14. STATUS: Is the facility operational, closed, a proposal, idle, etc.
15. NOTES: Analyst notes

Note: Tables from this point forward are generated during a Treatment Optimizer scenario run. They are listed in the order that they are generated.

HAUL_COSTS table

Location: optimizer\db\scenario1\optimizer_results.accdb. This table is one of the first that is created when a Treatment Optimizer scenario is run. It contains 2 rows for each plot with haul costs (**M**erch and **C**hip) calculated for the lowest cost route for each plot for each wood category, considering all enabled facilities and travel modes.

	Column name	Descriptive name	Data type
1	HAUL_COST_ID (Primary key)		Integer
2	BIOSUM_PLOT_ID (Foreign key)	See PLOT table	Text(24)
3	RAILHEAD_ID		Integer
4	PSITE_ID		Integer
5	TRANSFER_COST_DPGT	Cost to transfer a load Old name was TRANSFER_COST	Double
6	ROAD_COST_DPGT	Truck and driver haul cost. Old name was ROAD_COST	Double
7	RAIL_COST_DPGT	Cost by railroad. Old name was RAIL_COST	Double
8	COMPLETE_HAUL_COST_DPGT	Total cost to haul a load. Old name: TOTAL_HAUL_COST	Double
9	MATERIALCD	(M)erch or (C)hip facility	Text(2)

1. HAUL_COST_ID: Unique ID for records in this table.
2. BIOSUM_PLOT_ID: Foreign key to link with records in travel_time table.
3. RAILHEAD_ID: Foreign key to link with records in travel_time table. When doing rail travel times the railhead_id value corresponds to the value in the travel_time.RAILHEAD_ID field.
4. PSITE_ID: IF road travel THEN TRAVEL_TIME.PSITE_ID ELSE IF rail travel THEN TRAVEL_TIME.COLLECTOR_ID. This foreign key links to the final destination wood processing facility (psite).
5. TRANSFER_COST_DPGT: IF RAILHEAD_ID IS NOT NULL AND MATERIALCD IS 'M' THEN TRANSFER_COST_DPGT = SCENARIO_COSTS.RAIL_MERCH_TRANSFER_PGT
   ELSE IF RAILHEAD_ID IS NOT NULL AND MATERIALCD IS 'C' THEN TRANSFER_COST_DPGT = SCENARIO_COSTS.RAIL_CHIP_TRANSFER_PGT. Merch and chips can have different transfer costs, and these are specified by the user.
6. ROAD_COST_DPGT: SCENARIO_COSTS.ROAD_HAUL_COST_PGT_PER_HOUR * TRAVEL_TIME. ONE_WAY_HOURS. Truck and driver haul cost, as entered by the user, should be 2 * (two times) the dollars per green ton per hour, to account for cost of travel loaded and unloaded.
7. RAIL_COST_DPGT: (TRAVEL_TIME.ONE_WAY_HOURS * 45 MPH) * SCENARIO_COSTS.RAIL_HAUL_COST_PGT_PER_MILE. Cost for hauling on railroad segments is calculated using cost per mile, rather than per hour, because rail computations in GIS (and haul pricing) are typically performed based on distance, without regard to speed. However,

for consistency with the parameterization of travel to wood processing facilities not on the rail network, entries in the TRAVEL_TIME table for rail travel are in ONE_WAY_HOURS based on an assumed, or nominal, speed of 45 MPH.

8. COMPLETE_HAUL_COST_DPGT: TRANSFER_COST_DPGT + ROAD_COST_DPGT + RAIL_COST_DPGT
9. MATERIALCD: Indicates ‘M’ for merch or ‘C’ for chips (biomass). BioSum will consider treatment potentially valid only on plots for which a record exists in this table with MATERIALCD=’M’.

Note: With respect to fields 5, 6, and 7, the scenario_costs table resides in optimizer\db\scenario_optimizer_rule_definitions.mdb.

USERDEFINEDPLOTFILTER table

Location: optimizer\scenario1\db\optimizer_results.accdb. This table has the same structure as the PLOT table but only includes records that meet the requirements imposed by the user-defined plot filter

RULEDEFINITIONSCONDFILTER table

Location: optimizer\scenario1\db\optimizer_results.accdb. This table has the same structure as the COND table but only includes records that pass the user-defined condition filter sql from Rule Definitions > Filter Condition and meet the Land Ownership Groups requirements.

RULEDEFINITIONSPLOTFILTER table

Location: optimizer\scenario1\db\optimizer_results.accdb. This table has the same structure as the PLOT table but only includes conditions that that meet the requirements imposed by the user-defined plot filter and exist in RULEDEFINITIONSCONDFILTER. This table is used as a filter when creating the VALIDCOMBOS table. Records that don't exist in RULEDEFINITIONSPLOTFILTER are excluded from VALIDCOMBOS.

VALIDCOMBOS_FVSPREPOST table

Location: optimizer\scenario1\db\optimizer_results.accdb. BioSum creates a temporary work table named VALIDCOMBOS_FVSPRE that has 4 columns indicating whether or not an effectiveness variable exists in the FVS PRE table for each biosum_cond_id, rxpackage, rx, and rxcycle. A similar table called VALIDCOMBOS_FVSPOST is created against the FVS POST tables. BioSum supports up to 4 effectiveness variables. If records for an effectiveness variable exist in both the VALIDCOMBOS_FVSPRE and VALIDCOMBOS_FVSPOST tables, a row is added to the VALIDCOMBOS_FVSPREPOST table. The VALIDCOMBOS_FVSPRE and VALIDCOMBOS_FVSPOST tables validate the existence, in FVS output, of the PRE and POST treatment tables/columns the user selected in treatment optimizer.
FVS Variant + rxpackage combinations that are excluded because of the Optimizer rxpackage filter are excluded from this table and are not included in downstream Optimizer processing.

	Column name	Descriptive name	Data type
1	BIOSUM_COND_ID	See COND table	Text(25)
2	RXPACKAGE		Text(3)
3	RX		Text(3)
4	RXCYCLE		Text(1)
5	FVS_VARIANT		Text(2)

VALIDCOMBOS table

Location: optimizer\scenario1\db\optimizer_results.accdb. The table structure is the same as the VALIDCOMBOS_FVSPREPOST table except that it does not include the fvs_variant. Conditions in this table must exist in the following tables. (Tables that include rxpackage, rx, and rxcycle fields are also joined on that field):

1. RULEDEFINITIONSPLOTFILTER (Treatment Optimizer user-defined plot filter)
2. RULEDEFINITIONSCONDFILTER (Treatment Optimizer user-defined cond filter and ownership)
3. VALIDCOMBOS_FVSPREPOST (Pre/Post treatment tables exist in FVS Output for selected variables)
4. TREE_VOL_VAL_SUM_BY_RX_CYCLE_WORK (Summation of Processor Scenario Output). This table is not present in optimizer_results.accdb.
5. HARVEST_COSTS (Processor Scenario Output)
6. PSITE_ACCESSIBLE_WORKTABLE.cond_accessible_yn='Y'
7. PSITE_ACCESSIBLE_WORKTABLE.merch_haul_psite has valid values

COND_PSITE table

Location: optimizer\scenario1\db\optimizer_results.accdb. This table contains a row for every condition in the valid_combos table

	Column name	Descriptive name	Data type
1	BIOSUM_COND_ID	See COND table	Text(25)
2	MERCH_PSITE_NUM	Selected merch psite number	Integer
3	MERCH_PSITE_NAME	Selected merch psite name	Text(255)
4	CHIP_PSITE_NUM	Selected chip psite number	Integer
5	CHIP_PSITE_NAME	Selected chip psite name	Text(255)

1. BIOSUM_COND_ID:
2. MERCH_PSITE_NUM: HAUL_COST.PSITE_ID where HAUL_COSTS.BIOSUM_PLOT_ID = COND.BIOSUM_PLOT_ID and COND.BIOSUM_COND_ID = BIOSUM_COND_ID and HAUL_COSTS.MATERIALCD = “M”
3. MERCH_PSITE_NAME: PROCESSING_SITE.NAME where PROCESSING_SITE.PSITE_ID = MERCH_PSITE_NUM
4. CHIP_PSITE_NUM: HAUL_COST.PSITE_ID where HAUL_COSTS.BIOSUM_PLOT_ID = COND.BIOSUM_PLOT_ID and COND.BIOSUM_COND_ID = BIOSUM_COND_ID and HAUL_COSTS.MATERIALCD = “C”.
5. CHIP_PSITE_NAME: PROCESSING_SITE.NAME where PROCESSING_SITE.PSITE_ID = CHIP_PSITE_NUM. If CHIP_PSITE_NUM is NULL then NULL.

COND_AUDIT table

Location: db\audit_VAR.mdb where VAR is and FVS variant. This table was previously called PLOT_AUDIT. This table will be relocated to optimizer\scenario_1\audit.accdb and all variants will be included in a single database. The source of this table is the RULEDEFINITIONSCONDFILTER table. This means that conditions excluded by the user plot or condition filters will not exist on this table.

	Column name	Descriptive name	Data type
1	BIOSUM_COND_ID	See COND table	Text(25)
2	FVS_PREPOST_VARIABLES_YN		Text(1)
3	GIS_TRAVEL_TIMES_YN		Text(1)
4	PROCESSOR_TREE_VOL_VAL_YN		Text(1)
5	HARVEST_COSTS_YN		Text(1)
6	COND_TOO_FAR_STEEP_YN		Text(1)
7	PSITE_MERCH_YN		Text(1)
8	PSITE_CHIP_YN		Text(1)

1. BIOSUM_COND_ID:
2. FVS_PREPOST_VARIABLES_YN: If BIOSUM_COND_ID exists in VALIDCOMBOS_FVSPREPOST then 'Y' else 'N'
3. GIS_TRAVEL_TIMES_YN: If BIOSUM_COND_ID exists in TRAVEL_TIME then 'Y' else 'N'. Note that travel_time doesn't contain BIOSUM_COND_ID so the query uses RULEDEFINITIONSCONDFILTER to complete the join to BIOSUM_PLOT_ID
4. PROCESSOR_TREE_VOL_VAL_YN: If BIOSUM_COND_ID exists in TREE_VOL_VAL_SUM_BY_RX_CYCLE_WORK then 'Y' else 'N'
5. HARVEST_COSTS_YN: If BIOSUM_COND_ID exists in HARVEST_COSTS (Processor) then 'Y' else 'N'
6. COND_TOO_FAR_STEEP_YN: PSITE_ACCESSIBLE_WORKTABLE.COND_TOO_FAR_STEEP_YN
7. PSITE_MERCH_YN: If PSITE_ACCESSIBLE_WORKTABLE.MERCH_HAUL_PSITE is null then 'N' else 'Y'
8. PSITE_CHIP_YN: If PSITE_ACCESSIBLE_WORKTABLE.CHIP_HAUL_PSITE is null then 'N' else 'Y'. Note that that a chip psite is not required for inclusion in the analysis.

COND_RX_AUDIT table

Location: db\audit_VAR.mdb where VAR is and FVS variant. This table was previously called PLOT_COND_RX_AUDIT. This table will be relocated to optimizer\scenario_1\audit.accdb and all variants will be included in a single database. The source of this table is the COND_AUDIT table but it is joined with the rxpackage table to generate a row for each BIOSUM_COND_ID, RXPACKAGE, RX, RXCYCLE where a valid RX is defined.

	Column name	Descriptive name	Data type
1	BIOSUM_COND_ID	See COND table	Text(25)
2	RXPACKAGE		Text(3)
3	RX		Text(3)
4	RXCYCLE		Text(1)
5	FVS_PREPOST_VARIABLES_YN		Text(1)
6	PROCESSOR_TREE_VOL_VAL_YN		Text(1)
7	HARVEST_COSTS_YN		Text(1)

1. BIOSUM_COND_ID:
2. RXPACKAGE:
3. RX:
4. RXCYCLE:
5. FVS_PREPOST_VARIABLES_YN: If BIOSUM_COND_ID exists in VALIDCOMBOS_FVSPREPOST then ‘Y’ else ‘N’
6. PROCESSOR_TREE_VOL_VAL_YN: If BIOSUM_COND_ID exists in TREE_VOL_VAL_SUM_BY_RX_CYCLE_WORK then ‘Y’ else ‘N’
7. HARVEST_COSTS_YN: If BIOSUM_COND_ID exists in HARVEST_COSTS (Processor) then ‘Y’ else ‘N’

ECON_BY_RX_CYCLE table

Location: optimizer\scenario1\db\optimizer_results.accdb. Stands are added to this table if they are in the VALIDCOMBOS, COND_PSITE, TREE_VOL_VAL, and HARVEST_COSTS tables

	Column name	Descriptive name	Data type
1	BIOSUM_COND_ID	See COND table	Text(25)
2	RXPACKAGE		Text(3)
3	RX		Text(3)
4	RXCYCLE		Text(1)
5	CHIP_VOL_CF		Double
6	MERCH_VOL_CF		Double
7	CHIP_WT_GT		Double
8	MERCH_WT_GT		Double
9	CHIP_VAL_DPA		Double
10	MERCH_VAL_DPA		Double
11	HARVEST_ONSITE_COST_DPA		Double
12	CHIP_HAUL_COST_DPA		Double
13	MERCH_HAUL_COST_DPA		Double
14	MERCH_CHIP_NR_DPA		Double
15	MERCH_NR_DPA		Double
16	USEBIOMASS_YN		Text(1)
17	MAX_NR_DPA		Double
18	ACRES	Acres	Double
19	OWNGRPCD	See COND table	Integer
20	HAUL_COSTS_DPA		Double

1. BIOSUM_COND_ID:
2. RXPACKAGE: TREE_VOL_VAL_BY_SPECIES_DIAM_GROUPS.RXPACKAGE
3. RX: TREE_VOL_VAL_BY_SPECIES_DIAM_GROUPS.RX
4. RXCYCLE: TREE_VOL_VAL_BY_SPECIES_DIAM_GROUPS.RXCYCLE
5. CHIP_VOL_CF: TREE_VOL_VAL_SUM_BY_RX_CYCLE.CHIP_VOL_CF
6. MERCH_VOL_CF: TREE_VOL_VAL_SUM_BY_RX_CYCLE.MERCH_VOL_CF
7. CHIP_WT_GT: TREE_VOL_VAL_SUM_ BY_RX_CYCLE.CHIP_WT_GT
8. MERCH_WT_GT: TREE_VOL_VAL_SUM_ BY_RX_CYCLE.MERCH_WT_GT
9. CHIP_VAL_DPA: TREE_VOL_VAL_SUM_ BY_RX_CYCLE.CHIP_VAL_DPA
10. MERCH_VAL_DPA: TREE_VOL_VAL_SUM_ BY_RX_CYCLE.MERCH_VAL_DPA
11. HARVEST_ONSITE_COST_DPA: HARVEST_COSTS.COMPLETE_CPA
12. CHIP_HAUL_COST_DPA: IF RXCYCLE = '1' THEN PSITE_ACCESSIBLE_WORKTABLE.CHIP_HAUL_COST_DPT * CHIP_WT_GT ELSE IF RXCYCLE = '2' THEN PSITE_ACCESSIBLE_WORKTABLE.CHIP_HAUL_COST_DPT * CHIP_WT_GT *

SCENARIO_COST_REVENUE_ESCALATORS.ESCALATOROPERATINGCOSTS_CYCLE2
ELSE IF RXCYCLE = ‘3’ THEN PSITE_ACCESSIBLE_WORKTABLE.CHIP_HAUL_COST_DPT * CHIP_WT_GT *
SCENARIO_COST_REVENUE_ESCALATORS.ESCALATOROPERATINGCOSTS_CYCLE3
ELSE IF RXCYCLE = ‘4’ THEN PSITE_ACCESSIBLE_WORKTABLE.CHIP_HAUL_COST_DPT * CHIP_WT_GT *
SCENARIO_COST_REVENUE_ESCALATORS.ESCALATOROPERATINGCOSTS_CYCLE4

13. MERCH_HAUL_COST_DPA: IF RXCYCLE = ‘1’ THEN PSITE_ACCESSIBLE_WORKTABLE.MERCH_HAUL_COST_DPT * MERCH_WT_GT
ELSE IF RXCYCLE = ‘2’ THEN PSITE_ACCESSIBLE_WORKTABLE.MERCH_HAUL_COST_DPT * MERCH_WT_GT *
SCENARIO_COST_REVENUE_ESCALATORS.ESCALATOROPERATINGCOSTS_CYCLE2
ELSE IF RXCYCLE = ‘3’ THEN PSITE_ACCESSIBLE_WORKTABLE.MERCH_HAUL_COST_DPT * MERCH_WT_GT *
SCENARIO_COST_REVENUE_ESCALATORS.ESCALATOROPERATINGCOSTS_CYCLE3
ELSE IF RXCYCLE = ‘4’ THEN PSITE_ACCESSIBLE_WORKTABLE.MERCH_HAUL_COST_DPT * MERCH_WT_GT *
SCENARIO_COST_REVENUE_ESCALATORS.ESCALATOROPERATINGCOSTS_CYCLE4
14. MERCH_CHIP_NR_DPA: MERCH_VAL_DPA + CHIP_VAL_DPA – HARVEST_ONSITE_COST_DPA – (CHIP_HAUL_COST_DPA + MERCH_HAUL_COST_DPA)
15. MERCH_NR_DPA: MERCH_VAL_DPA – HARVEST_ONSITE_COST_DPA – MERCH_HAUL_COST_DPA
16. USEBIOMASS_YN:
    a. IF CHIP_PSITE_NUM IS NULL THEN USEBIOMASS_YN = “N”
    b. IF RXCYCLE = ‘1’ THEN IF scenario_tree_species_diam_dollar_values.CHIP_VALUE > PSITE_ACCESSIBLE_WORKTABLE.CHIP_HAUL_COST_DPT THEN USEBIOMASS_YN = “Y” ELSE “N”
    ELSE IF RXCYCLE = ‘2’ THEN IF scenario_tree_species_diam_dollar_values.CHIP_VALUE > PSITE_ACCESSIBLE_WORKTABLE.CHIP_HAUL_COST_DPT * SCENARIO_COST_REVENUE_ESCALATORS.ESCALATOROPERATINGCOSTS_CYCLE2 THEN USEBIOMASS_YN = “Y” ELSE “N”
    ELSE IF RXCYCLE = ‘3’ THEN IF scenario_tree_species_diam_dollar_values.CHIP_VALUE > PSITE_ACCESSIBLE_WORKTABLE.CHIP_HAUL_COST_DPT * SCENARIO_COST_REVENUE_ESCALATORS.ESCALATOROPERATINGCOSTS_CYCLE3 THEN USEBIOMASS_YN = “Y” ELSE “N”
    ELSE IF RXCYCLE = ‘4’ THEN IF

scenario_tree_species_diam_dollar_values.CHIP_VALUE > PSITE_ACCESSIBLE_WORKTABLE.CHIP_HAUL_COST_DPT * SCENARIO_COST_REVENUE_ESCALATORS.ESCALATOROPERATINGCOSTS_CYCLE4 THEN USEBIOMASS_YN = “Y” ELSE “N”

   c. IF CHIP_WT_GT = 0 THEN USEBIOMASS_YN = “N”

   These conditions are listed in the order by which they are tested. Once USEBIOMASS_YN is set to ‘N’, it cannot be changed to ‘Y’

17. MAX_NR_DPA: IF USEBIOMASS_YN =”Y” THEN MERCH_CHIP_NR_DPA ELSE MERCH_NR_DPA
18. ACRES: COND.ACRES where COND.BIOSUM_COND_ID = BIOSUM_COND_ID
19. OWNGRPCD: COND.OWNGRPCD where COND.BIOSUM_COND_ID = BIOSUM_COND_ID
20. HAUL_COSTS_DPA: IF USEBIOMASS_YN =”Y” THEN MERCH_HAUL_COST_DPA + CHIP_HAUL_COST_DPA ELSE MERCH_HAUL_COST_DPA

Order of second-pass field calculations, due to dependencies

a) USEBIOMASS_YN
b) HAUL_COSTS_DPA and MAX_NR_DPA

ECON_BY_RX_UTILIZED_SUM table

Location: optimizer\scenario1\db\optimizer_results.accdb. This table was previously called ECON_BY_RX_SUM.

	Column name	Descriptive name	Data type
1	BIOSUM_COND_ID	See COND table	Text(25)
2	RXPACKAGE		Text(3)
3	CHIP_VOL_CF_UTILIZED	Old name: CHIP_VOL_CF	Double
4	MERCH_VOL_CF		Double
5	CHIP_WT_GT_UTILIZED	Old name: CHIP_WT_GT	Double
6	MERCH_WT_GT		Double
7	CHIP_VAL_DPA_UTILIZED	Old name: CHIP_VAL_DPA	Double
8	MERCH_VAL_DPA		Double
9	HARVEST_ONSITE_COST_DPA		Double
10	CHIP_HAUL_COST_DPA_UTILIZED	Old name: CHIP_HAUL_COST_DPA	Double
11	MERCH_HAUL_COST_DPA		Double
12	MERCH_CHIP_NR_DPA		Double
13	MERCH_NR_DPA		Double
14	MAX_NR_DPA		Double
15	ACRES	Acres	Double
16	TREATED ACRES	Treated acres	Double
17	OWNGRPCD	See COND table	Integer
18	HAUL_COSTS_DPA		Double
19	HVST_TYPE_BY_CYCLE		Text(4)

1. BIOSUM_COND_ID:
2. RXPACKAGE: ECON_BY_RX_CYCLE.RXPACKAGE
3. CHIP_VOL_CF_UTILIZED: SUM(ECON_BY_RX_CYCLE.CHIP_VOL_CF) WHERE ECON_BY_RXCYCLE. USEBIOMASS_YN = ‘Y’
4. MERCH_VOL_CF: SUM(ECON_BY_RX_CYCLE.MERCH_VOL_CF)
5. CHIP_WT_GT_UTILIZED: SUM(ECON_BY_RX_CYCLE.CHIP_WT_GT) WHERE ECON_BY_RXCYCLE.USEBIOMASS_YN = ‘Y’
6. MERCH_WT_GT: SUM(ECON_BY_RX_CYCLE.MERCH_WT_GT)
7. CHIP_VAL_DPA_UTILIZED: SUM(ECON_BY_RX_CYCLE.CHIP_VAL_DPA) WHERE ECON_BY_RXCYCLE.USEBIOMASS_YN = ‘Y’
8. MERCH_VAL_DPA: SUM(ECON_BY_RX_CYCLE.MERCH_VAL_DPA)
9. HARVEST_ONSITE_COST_DPA: SUM(ECON_BY_RX_CYCLE.HARVEST_ONSITE_COST_DPA)
10. CHIP_HAUL_COST_DPA_UTILIZED: SUM(ECON_BY_RX_CYCLE.CHIP_HAUL_COST_DPA) WHERE ECON_BY_RXCYCLE.USEBIOMASS_YN = ‘Y’
11. MERCH_HAUL_COST_DPA: SUM(ECON_BY_RX_CYCLE.MERCH_HAUL_COST_DPA)
12. MERCH_CHIP_NR_DPA: SUM(ECON_BY_RX_CYCLE.MERCH_CHIP_NR_DPA)
13. MERCH_NR_DPA: SUM(ECON_BY_RX_CYCLE.MERCH_NR_DPA)

14. MAX_NR_DPA: SUM(ECON_BY_RX_CYCLE.MAX_NR_DPA)
15. ACRES: COND.ACRES where COND.BIOSUM_COND_ID = BIOSUM_COND_ID
16. TREATED_ACRES: SUM(ECON_BY_RX_CYCLE.ACRES)
17. OWNGRPCD: COND.OWNGRPCD where COND.BIOSUM_COND_ID = BIOSUM_COND_ID
18. HAUL_COSTS_DPA: SUM(ECON_BY_RX_CYCLE.HAUL_COSTS_DPA)
19. HVST_TYPE_BY_CYCLE: Indicates which cycles had non-zero harvest as determined from the ECON_BY_RX_CYCLE table, for example: '3030' indicates both merch and chip volume harvested in cycles 1 and 3; '1002' indicates merch vol harvested at cycle 1 (but not chips) and chip volume at cycle 4 (but not merch).

POST_ECONOMIC_WEIGHTED table

Location: optimizer\scenario1\db\optimizer_results.accdb. The schema of this table will vary depending upon the economic variable selected as the basis of the custom economic variable. Five fields will be added for each custom weighted variable calculated (4 cycles and a total). The name of the fields is based on the economic variable type selected. This schema includes an example of a custom total_volume calculation.

	Column name	Descriptive name	Data type
1	BIOSUM_COND_ID	See COND table	Text(25)
2	RXPACKAGE		Text(3)
3	C1_TOTAL_VOLUME_2		Double
4	C2_TOTAL_VOLUME_2		Double
5	C3_TOTAL_VOLUME_2		Double
6	C4_TOTAL_VOLUME_2		Double
7	TOTAL_VOLUME_2		Double

1. BIOSUM_COND_ID:
2. RXPACKAGE:
3. C1_TOTAL_VOLUME_2: Cycle 1 value for sample custom weighted economic variable
4. C2_TOTAL_VOLUME_2: Cycle 2 value for sample custom weighted economic variable
5. C3_TOTAL_VOLUME_2: Cycle 3 value for sample custom weighted economic variable
6. C4_TOTAL_VOLUME_2: Cycle 4 value for sample custom weighted economic variable
7. TOTAL_VOLUME_2: Total value for sample custom weighted economic variable

The basis for all custom weighted economic variable calculations is the ECON_BY_RX_CYCLE table. Weights are applied as configured for each condition/cycle

VARIABLE NAME PREFIX	VARIABLE_VALUE
chip_volume_	ECON_BY_RX_CYCLE.CHIP_WT_GT
merchantable_volume_	ECON_BY_RX_CYCLE.MERCH_WT_GT
total_volume_	ECON_BY_RX_CYCLE.CHIP_WT_GT + ECON_BY_RX_CYCLE.MERCH_WT_GT
net_revenue_	ECON_BY_RX_CYCLE.MAX_NR_DPA
treatment_haul_costs_	ECON_BY_RX_CYCLE.HARVEST_ONSITE_COST_DPA + ECON_BY_RX_CYCLE.MERCH_HAUL_COST_DPA + (IF (ECON_BY_RX_CYCLE. MERCH_CHIP_NR_DPA < ECON_BY_RX_CYCLE.MAX_NR_DPA THEN 0, ELSE ECON_BY_RX_CYCLE. CHIP_HAUL_COST_DPA))

onsite_treatment_costs_	ECON_BY_RX_SUM. HARVEST_ONSITE_COST_DPA

EFFECTIVE table

Location: optimizer\scenario1\db\optimizer_results.accdb. The name of this table will be prefixed with either CYCLE1_ (no weighted variables are used) or ALL_CYCLES_ (at least one weighted variable is used in the analysis). The source of this table are the conditions listed in the VALIDCOMBOS table. At least one FVS Effective Variable is required. FVS Variables 2 - 4 are optional.

	Column name	Descriptive name	Data type
1	BIOSUM_COND_ID	See COND table	Text(25)
2	RXPACKAGE		Text(3)
3	RX		Text(3)
4	RXCYCLE		Text(1)
5	NR_DPA		Double
6	NET_REVENUE_X	Optional net revenue filter	Double
7	PRE_VARIABLE1_NAME	Required Pre-treatment FVS Effective Variable 1	Text(100)
8	POST_VARIABLE1_NAME	Required Post-treatment FVS Effective Variable 1	Text(100)
9	PRE_VARIABLE1_VALUE		Double
10	POST_VARIABLE1_VALUE		Double
11	VARIABLE1_CHANGE		Double
12	VARIABLE1_BETTER_YN		Text(1)
13	VARIABLE1_WORSE_YN		Text(1)
14	VARIABLE1_EFFECTIVE_YN		Text(1)
15	PRE_VARIABLE2_NAME	Optional pre-treatment FVS Effective Variable 2	Text(100)
16	POST_VARIABLE2_NAME	Optional post-treatment FVS Effective Variable 2	Text(100)
17	PRE_VARIABLE2_VALUE		Double
18	POST_VARIABLE2_VALUE		Double
19	VARIABLE2_CHANGE		Double
20	VARIABLE2_BETTER_YN		Text(1)
21	VARIABLE2_WORSE_YN		Text(1)
22	VARIABLE2_EFFECTIVE_YN		Text(1)
23	PRE_VARIABLE3_NAME	Optional pre-treatment FVS Effective Variable 3	Text(100)
24	POST_VARIABLE3_NAME	Optional post-treatment FVS Effective Variable 3	Text(100)
25	PRE_VARIABLE3_VALUE		Double
26	POST_VARIABLE3_VALUE		Double
27	VARIABLE3_CHANGE		Double
28	VARIABLE3_BETTER_YN		Text(1)
29	VARIABLE3_WORSE_YN		Text(1)
30	VARIABLE3_EFFECTIVE_YN		Text(1)
31	PRE_VARIABLE4_NAME	Optional pre-treatment FVS Effective Variable 4	Text(100)
32	POST_VARIABLE4_NAME	Optional post-treatment FVS Effective Variable 4	Text(100)
33	PRE_VARIABLE4_VALUE		Double
34	POST_VARIABLE4_VALUE		Double
35	VARIABLE4_CHANGE		Double

36	VARIABLE4_BETTER_YN		Text(1)
37	VARIABLE4_WORSE_YN		Text(1)
38	VARIABLE4_EFFECTIVE_YN		Text(1)
39	OVERALL_EFFECTIVE_YN		Text(1)

1. BIOSUM_COND_ID:
2. RXPACKAGE:
3. RX:
4. RXCYCLE: Due to legacy architecture, rxcycle will always be ‘1’ even if weighted variables are used.
5. NR_DPA: ECON_BY_RX_CYCLE.MAX_NR_DPA. Because rxcycle is always ‘1’, this will always be ECON_BY_RX_CYCLE.MAX_NR_DPA for cycle 1.
6. NET_REVENUE_X: If a weighted net revenue filter is used, this field contains the value of that filter for all 4 cycles. The name of this field is the same as the weighted net revenue filter. The default net revenue filter is NET_REVENUE_FILTER_1 which is ECON_BY_RX_SUM.MAX_NR_DPA where ECON_BY_RX_SUM.BIOSUM_COND_ID = BIOSUM_COND_ID
7. PRE_VARIABLE1_NAME: Contains the pre-FVS table and variable names concatenated with a “.” for FVS Effective Variable 1
8. POST_VARIABLE1_NAME: Contains the post-FVS table and variable names concatenated with a “.” for FVS Effective Variable 1
9. PRE_VARIABLE1_VALUE: The value from the pre-FVS table for FVS Effective Variable 1. The pre-FVS table is joined to the EFFECTIVE table USING BIOSUM_COND_ID, RXPACKAGE, RX, AND RXCYCLE
10. POST_VARIABLE1_VALUE: The value from the post-FVS table for FVS Effective Variable 2. The post-FVS table is joined to the EFFECTIVE table USING BIOSUM_COND_ID, RXPACKAGE, RX, AND RXCYCLE
11. VARIABLE1_CHANGE: If PRE_VARIABLE1_VALUE IS NOT NULL AND POST_VARIABLE1_VALUE IS NOT NULL then POST_VARIABLE1_VALUE - PRE_VARIABLE1_VALUE
12. VARIABLE1_BETTER_YN: If the Variable 1 Post-Treatment Improvement Expression = TRUE then ‘Y’ else ‘N’. This improvement expression is defined by the analyst on the Treatment Optimizer FVS Variables Effective tab.
13. VARIABLE1_WORSE_YN: If the Variable 1 Post-Treatment Disimprovement Expression = TRUE then ‘Y’ else ‘N’. This disimprovement expression is defined by the analyst on the Treatment Optimizer FVS Variables Effective tab.
14. VARIABLE1_EFFECTIVE_YN: If the Variable 1 Effective Treatment Expression = TRUE then ‘Y’ else ‘N’. This effective treatment expression is defined by the analyst on the Treatment Optimizer FVS Variables Effective tab.

15. PRE_VARIABLE2_NAME: See PRE_VARIABLE1_NAME but source is FVS Effective Variable 2
16. POST_VARIABLE2_NAME: See PRE_VARIABLE1_NAME but source is FVS Effective Variable 2
17. PRE_VARIABLE2_VALUE: See PRE_VARIABLE1_VALUE but source is FVS Effective Variable 2
18. POST_VARIABLE2_VALUE: See POST_VARIABLE1_VALUE but source is FVS Effective Variable 2
19. VARIABLE2_CHANGE: If PRE_VARIABLE2_VALUE IS NOT NULL AND POST_VARIABLE2_VALUE IS NOT NULL then POST_VARIABLE2_VALUE - PRE_VARIABLE2_VALUE
20. VARIABLE2_BETTER_YN: See VARIABLE2_BETTER_YN but source is Variable 2 Post-Treatment Improvement Expression
21. VARIABLE2_WORSE_YN: See VARIABLE1_WORSE_YN but source is Variable 2 Post-Treatment Disimprovement Expression
22. VARIABLE2_EFFECTIVE_YN: See VARIABLE1_EFFECTIVE_YN but source is Variable 2 Effective Treatment Expression
23. PRE_VARIABLE3_NAME: See PRE_VARIABLE1_NAME but source is FVS Effective Variable 3
24. POST_VARIABLE3_NAME: See PRE_VARIABLE1_NAME but source is FVS Effective Variable 3
25. PRE_VARIABLE3_VALUE: See PRE_VARIABLE1_VALUE but source is FVS Effective Variable 3
26. POST_VARIABLE3_VALUE: See POST_VARIABLE1_VALUE but source is FVS Effective Variable 3
27. VARIABLE3_CHANGE: If PRE_VARIABLE3_VALUE IS NOT NULL AND POST_VARIABLE3_VALUE IS NOT NULL then POST_VARIABLE3_VALUE - PRE_VARIABLE3_VALUE
28. VARIABLE3_BETTER_YN: See VARIABLE1_BETTER_YN but source is Variable 3 Post-Treatment Improvement Expression
29. VARIABLE3_WORSE_YN: See VARIABLE1_WORSE_YN but source is Variable 3 Post-Treatment Disimprovement Expression
30. VARIABLE3_EFFECTIVE_YN: See VARIABLE1_EFFECTIVE_YN but source is Variable 3 Effective Treatment Expression
31. PRE_VARIABLE4_NAME: See PRE_VARIABLE1_NAME but source is FVS Effective Variable 4
32. POST_VARIABLE4_NAME: See PRE_VARIABLE1_NAME but source is FVS Effective Variable 4

33. PRE_VARIABLE4_VALUE: See PRE_VARIABLE1_VALUE but source is FVS Effective Variable 4
34. POST_VARIABLE4_VALUE: See POST_VARIABLE1_VALUE but source is FVS Effective Variable 4
35. VARIABLE4_CHANGE: If PRE_VARIABLE4_VALUE IS NOT NULL AND POST_VARIABLE4_VALUE IS NOT NULL then POST_VARIABLE4_VALUE - PRE_VARIABLE4_VALUE
36. VARIABLE4_BETTER_YN: See VARIABLE1_BETTER_YN but source is Variable 4 Post-Treatment Improvement Expression
37. VARIABLE4_WORSE_YN: See VARIABLE1_WORSE_YN but source is Variable 4 Post-Treatment Disimprovement Expression
38. VARIABLE4_EFFECTIVE_YN: See VARIABLE1_EFFECTIVE_YN but source is Variable 4 Effective Treatment Expression
39. OVERALL_EFFECTIVE_YN: If the Overall Effectiveness Expression = TRUE then ‘Y’ else ‘N’. This overall effectiveness expression is defined by the analyst on the Treatment Optimizer FVS Variables Effective tab.

OPTIMIZATION table

Location: optimizer\scenario1\db\optimizer_results.accdb. The name of this table will be prefixed with either CYCLE1_ (no weighted variables are used) or ALL_CYCLES_ (at least one weighted variable is used in the analysis). The source of this table are the condition/rx records from in the EFFECTIVE table where OVERALL_EFFECTIVE_YN = ‘Y’.

	Column name	Descriptive name	Data type
1	BIOSUM_COND_ID	See COND table	Text(25)
2	RXPACKAGE		Text(3)
3	RX		Text(3)
4	RXCYCLE		Text(1)
5	PRE_VARIABLE_NAME		Text(100)
6	POST_VARIABLE_NAME		Text(100)
7	PRE_VARIABLE_VALUE		Double
8	POST_VARIABLE_VALUE		Double
9	CHANGE_VALUE		Double
10	AFFORDABLE_YN		Text(1)
11	NET_REVENUE_X or ONSITE_TREATMENT_COSTS_X	Optional	Double

1. BIOSUM_COND_ID:
2. RXPACKAGE:
3. RX:
4. RXCYCLE: Due to legacy architecture, rxcycle will always be ‘1’ even if weighted variables are used.
5. PRE_VARIABLE_NAME: IF optimization variable is ‘Revenue’ THEN ‘ECON_BY_RX_CYCLE.MAX_NR_DPA’
   ELSE IF optimization variable is ‘Merchantable Volume’ THEN ‘ECON_BY_RX_CYCLE. MERCH_WT_GT’
   ELSE IF optimization variable is an economic attribute THEN the analyst-selected POST Optimization economic attribute, for example: ‘TOTAL_VOLUME_1’
   ELSE the analyst-selected POST or (POST-PRE) Optimization stand attribute. The format for FVS stand attributes is the pre-FVS table and variable names concatenated with a ‘.’
6. POST_VARIABLE_NAME: IF optimization variable is ‘Revenue’ THEN ‘ECON_BY_RX_CYCLE.MAX_NR_DPA’
   ELSE IF optimization variable is ‘Merchantable Volume’ THEN ‘ECON_BY_RX_CYCLE. MERCH_WT_GT’
   ELSE IF optimization variable is an economic attribute THEN the analyst-selected Optimization economic attribute, for example: ‘TOTAL_VOLUME_1’
   ELSE the analyst-selected POST or (POST-PRE) Optimization stand attribute. The format for FVS stand attributes is the post-FVS table and variable names concatenated with a ‘.’
7. PRE_VARIABLE_VALUE: IF optimization variable is ‘Revenue’ THEN ECON_BY_RX_CYCLE.MAX_NR_DPA

ELSE IF optimization variable is 'Merchantable Volume' THEN ECON_BY_RX_CYCLE. MERCH_WT_GT
ELSE IF optimization variable is an economic attribute THEN the variable value for the analyst-selected Optimization economic attribute below:

VARIABLE_NAME	VARIABLE_VALUE
chip_volume_1	ECON_BY_RX_SUM.CHIP_WT_GT
merchantable_volume_1	ECON_BY_RX_SUM.MERCH_WT_GT
total_volume_1	ECON_BY_RX_SUM.CHIP_WT_GT + ECON_BY_RX_SUM.MERCH_WT_GT
net_revenue_1	ECON_BY_RX_SUM.MAX_NR_DPA
treatment_haul_costs_1	ECON_BY_RX_SUM. HARVEST_ONSITE_COST_DPA + ECON_BY_RX_SUM. MERCH_HAUL_COST_DPA + (IF (ECON_BY_RX_SUM. MERCH_CHIP_NR_DPA < ECON_BY_RX_SUM.MAX_NR_DPA THEN 0, ELSE ECON_BY_RX_SUM. CHIP_HAUL_COST_DPA))
onsite_treatment_costs_1	ECON_BY_RX_SUM. HARVEST_ONSITE_COST_DPA
custom economic variable	POST_ECONOMIC_WEIGHTED. *VARIABLE_NAME* where variable_name is the name of custom variable.

ELSE the value from the FVS Pre table defined in PRE_VARIABLE_NAME.

8. POST_VARIABLE_VALUE: IF the optimization variable is 'Revenue', 'Merchantable Volume' or an economic attribute, see the description for PRE_VARIABLE_VALUE. For these optimization variables there is only one value per cycle. If the optimization variable is an FVS stand attribute the value is from the FVS Post table defined in POST_VARIABLE_NAME.
9. CHANGE_VALUE: IF optimization variable is 'Revenue' THEN 0
   ELSE IF optimization variable is 'Merchantable Volume' THEN 0
   ELSE IF optimization variable is an economic attribute THEN 0
   ELSE POST_VARIABLE_VALUE - PRE_VARIABLE_VALUE
10. AFFORDABLE_YN: If the conditions of the 'Dollars Per Acre' filter setting are met then 'Y' else 'N'. This optional filter is configured by the analyst in the FVS Variables Optimization settings. If no filter is configured then 'Y'.

11. NET_REVENUE_X or ONSITE_TREATMENT_COSTS_X: If a 'Dollars Per Acre' filter is configured then this field contains the value of the filter. For default economic variables see the reference table under item #7 in this list for the calculations. For custom economic variables see the definition for the POST_ECONOMIC_WEIGHTED table. If no filter is configured, this column will be absent.

TIEBREAKER table

Location: optimizer\scenario1\db\optimizer_results.accdb. The source of this table are the condition/rx records from in the OPTIMIZATION table where AFFORDABLE_YN = 'Y'.

	Column name	Descriptive name	Data type
1	BIOSUM_COND_ID	See COND table	Text(25)
2	RXPACKAGE		Text(3)
3	RX		Text(3)
4	RXCYCLE		Text(1)
5	LAST_TIEBREAK_RANK	Required	Integer
6	PRE_VARIABLE1_NAME	Optional	Text(100)
7	POST_VARIABLE1_NAME	Optional	Text(100)
8	PRE_VARIABLE1_VALUE	Optional	Double
9	POST_VARIABLE1_VALUE	Optional	Double
10	VARIABLE1_CHANGE		Double

1. BIOSUM_COND_ID:
2. RXPACKAGE:
3. RX:
4. RXCYCLE: Due to legacy architecture, rxcycle will always be '1' even if weighted variables are used.
5. LAST_TIEBREAK_RANK: Ranking assigned to the rx by the analyst in the FVS Variables Tiebreaker settings. The lowest number wins when breaking ties between multiple rx's.
6. PRE_VARIABLE_NAME1: IF tie breaker attribute is an economic attribute THEN the analyst-selected tie breaker economic attribute, for example: 'TOTAL_VOLUME_1' ELSE the analyst-selected POST or (POST-PRE) tie breaker stand attribute. The format for FVS stand attributes is the pre-FVS table and variable names concatenated with a '.' This field is optional and may be null.
7. POST_VARIABLE1_NAME: IF tie breaker attribute is an economic attribute THEN the analyst-selected tie breaker economic attribute, for example: 'TOTAL_VOLUME_1' ELSE the analyst-selected POST or (POST-PRE) tie breaker stand attribute. The format for FVS stand attributes is the post-FVS table and variable names concatenated with a '.'
8. PRE_VARIABLE1_VALUE: IF tie breaker variable is an economic attribute THEN see definition of OPTIMIZATION.PRE_VARIABLE_VALUE for calculation ELSE the value from the FVS Pre table defined in PRE_VARIABLE_NAME.

9. POST_VARIABLE1_VALUE: IF the tie breaker variable is an economic attribute, see the description for PRE_VARIABLE1_VALUE. For economic attributes, there is only one value per cycle. If the tie breaker variable is an FVS stand attribute the value is from the FVS Post table defined in POST_VARIABLE_NAME.
10. VARIABLE1_CHANGE: IF tie breaker variable is an economic attribute THEN 0 ELSE POST_VARIABLE_VALUE - PRE_VARIABLE_VALUE

BEST_RX_SUMMARY_BEFORE_TIEBREAKS table

Location: optimizer\scenario1\db\optimizer_results.accdb. The source of this table are the condition/rx records from in the OPTIMIZATION table identified as optimal, according to the optimization criteria configured by the analyst, and where AFFORDABLE_YN = ‘Y’.

	Column name	Descriptive name	Data type
1	BIOSUM_COND_ID	See COND table	Text(25)
2	RXPACKAGE		Text(3)
3	RX		Text(3)
4	ACRES	Acres	Double
5	OWNGRPCD	See COND table	Integer
6	OPTIMIZATION_VALUE		Double
7	TIEBREAKER_VALUE		Double
8	LAST_TIEBREAK_RANK		Integer

1. BIOSUM_COND_ID:
2. RXPACKAGE:
3. RX:
4. ACRES: COND.ACRES where COND.BIOSUM_COND_ID = BIOSUM_COND_ID
5. OWNGRPCD: COND.OWNGRPCD where COND.BIOSUM_COND_ID = BIOSUM_COND_ID
6. OPTIMIZATION_VALUE: OPTIMIZATION.POST_VARIABLE_VALUE where this table is joined to the OPTIMIZATION table on BIOSUM_COND_ID, RXPACKAGE, AND RX.
7. TIEBREAKER_VALUE: TIEBREAKER.POST_VARIABLE1_VALUE where this table is joined to the TIEBREAKER table on BIOSUM_COND_ID, RXPACKAGE, AND RX.
8. LAST_TIEBREAK_RANK: TIEBREAKER.LAST_TIEBREAK_RANK where this table is joined to the TIEBREAKER table on BIOSUM_COND_ID, RXPACKAGE, AND RX.

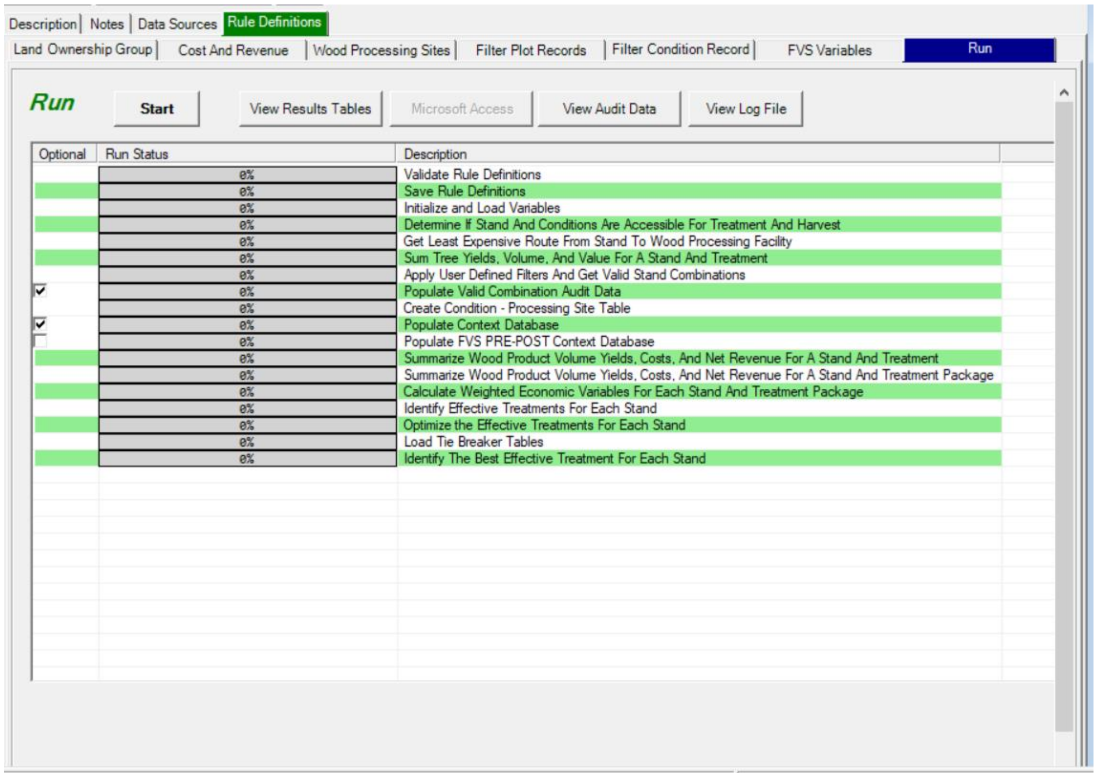
BEST_RX_SUMMARY table

Location: optimizer\scenario1\db\optimizer_results.accdb. The source of this table are the condition/rx records from in the BEST_RX_SUMMARY_BEFORE_TIEBREAKS table that win the tie according to the tie breaker criteria configured by the analyst.

	Column name	Descriptive name	Data type
1	BIOSUM_COND_ID	See COND table	Text(25)
2	RXPACKAGE		Text(3)
3	RX		Text(3)
4	ACRES	Acres	Double
5	OWNGRPCD	See COND table	Integer
6	OPTIMIZATION_VALUE		Double
7	TIEBREAKER_VALUE		Double
8	LAST_TIEBREAK_RANK		Integer

9. BIOSUM_COND_ID:
10. RXPACKAGE:
11. RX:
12. ACRES: COND.ACRES where COND.BIOSUM_COND_ID = BIOSUM_COND_ID
13. OWNGRPCD: COND.OWNGRPCD where COND.BIOSUM_COND_ID = BIOSUM_COND_ID
14. OPTIMIZATION_VALUE: OPTIMIZATION.POST_VARIABLE_VALUE where this table is joined to the OPTIMIZATION table on BIOSUM_COND_ID, RXPACKAGE, AND RX.
15. TIEBREAKER_VALUE: TIEBREAKER.POST_VARIABLE1_VALUE where this table is joined to the TIEBREAKER table on BIOSUM_COND_ID, RXPACKAGE, AND RX.
16. LAST_TIEBREAK_RANK: TIEBREAKER.LAST_TIEBREAK_RANK where this table is joined to the TIEBREAKER table on BIOSUM_COND_ID, RXPACKAGE, AND RX.

CHANGES TO UI



Run

Optional	Run Status	Description
	0%	Validate Rule Definitions
	0%	Save Rule Definitions
	0%	Initialize and Load Variables
	0%	Determine If Stand And Conditions Are Accessible For Treatment And Harvest
	0%	Get Least Expensive Route From Stand To Wood Processing Facility
	0%	Sum Tree Yields, Volume, And Value For A Stand And Treatment
	0%	Apply User Defined Filters And Get Valid Stand Combinations
☑	0%	Populate Valid Combination Audit Data
	0%	Create Condition - Processing Site Table
☑	0%	Populate Context Database
☐	0%	Populate FVS PRE-POST Context Database
	0%	Summarize Wood Product Volume Yields, Costs, And Net Revenue For A Stand And Treatment
	0%	Summarize Wood Product Volume Yields, Costs, And Net Revenue For A Stand And Treatment Package
	0%	Calculate Weighted Economic Variables For Each Stand And Treatment Package
	0%	Identify Effective Treatments For Each Stand
	0%	Optimize the Effective Treatments For Each Stand
	0%	Load Tie Breaker Tables
	0%	Identify The Best Effective Treatment For Each Stand

The context database will be generated if the checkbox is selected. This checkbox will be selected by default. The FVS PREPOST tables will be copied to the Optimizer directory in a separate database if the FVS Context Database checkbox is checked. This checkbox will be unselected by default. The FVS Context database will only contain the “real” (as opposed to linked) tables from the PRE_POST_FVS .accdbs. The state of these checkboxes will not persist between BioSum sessions like most other Optimizer scenario parameters are.

HARVEST_METHOD_REF_C

	Column name	Descriptive name	Data type
1	RX		Text(3)
2	RX_HARVEST_METHOD_LOW		Text(50)
3	RX_HARVEST_METHOD_LOW_ID		Integer
4	RX_HARVEST_METHOD_LOW_CATEGORY		Integer
5	RX_HARVEST_METHOD_LOW_CATEGORY_DESCR		Text(50)
6	RX_HARVEST_METHOD_STEEP		Text(50)
7	RX_HARVEST_METHOD_STEEP_ID		Integer
8	RX_HARVEST_METHOD_STEEP_CATEGORY		Integer
9	RX_HARVEST_METHOD_STEEP_CATEGORY_DESCR		Text(50)
10	USE_RX_HARVEST_METHOD_YN		Text(1)
11	STEEP_SLOPE_PCT		Integer
12	SCENARIO_HARVEST_METHOD_LOW		Text(50)
13	SCENARIO_HARVEST_METHOD_LOW_ID		Integer
14	SCENARIO_HARVEST_METHOD_LOW_CATEGORY		Integer
15	SCENARIO_HARVEST_METHOD_LOW_CATEGORY_DESCR		Text(50)
16	SCENARIO_HARVEST_METHOD_STEEP		Text(50)
17	SCENARIO_HARVEST_METHOD_STEEP_ID		Integer
18	SCENARIO_HARVEST_METHOD_STEEP_CATEGORY		Integer
19	SCENARIO_HARVEST_METHOD_STEEP_CATEGORY_DESCR		Text(50)

1. RXPACKAGE: RX.RX
2. RX_HARVEST_METHOD_LOW: RX.HARVESTMETHODLOWSLOPE
3. RX_HARVEST_METHOD_LOW_ID: HARVEST_METHODS.HARVESTMETHODID WHERE HARVEST_METHODS.METHOD = RX_HARVEST_METHOD_LOW and HARVEST_METHODS.METHOD.STEEP_YN = ‘N’
4. RX_HARVEST_METHOD_LOW_CATEGORY: HARVEST_METHODS.BIOSUM_CATEGORY WHERE HARVEST_METHODS.METHOD = RX_HARVEST_METHOD_LOW and HARVEST_METHODS.METHOD.STEEP_YN = ‘N’

5. RX_HARVEST_METHOD_LOW_CATEGORY_DESCR: HARVEST_METHODS.TOP_LIMB_SLOPE_STATUS WHERE HARVEST_METHODS.METHOD = RX_HARVEST_METHOD_LOW and HARVEST_METHODS.METHOD.STEEP_YN = ‘N’
6. RX_HARVEST_METHOD_STEEP: RX.HARVESTMETHODSTEEPSLOPE
7. RX_HARVEST_METHOD_STEEP_ID: HARVEST_METHODS.HARVESTMETHODID WHERE HARVEST_METHODS.METHOD = RX_HARVEST_METHOD_STEEP and HARVEST_METHODS.METHOD.STEEP_YN = ‘Y’
8. RX_HARVEST_METHOD_STEEP_CATEGORY: HARVEST_METHODS.BIOSUM_CATEGORY WHERE HARVEST_METHODS.METHOD = RX_HARVEST_METHOD_STEEP and HARVEST_METHODS.METHOD.STEEP_YN = ‘Y’
9. RX_HARVEST_METHOD_STEEP_CATEGORY: HARVEST_METHODS.BIOSUM_CATEGORY WHERE HARVEST_METHODS.METHOD = RX_HARVEST_METHOD_STEEP and HARVEST_METHODS.METHOD.STEEP_YN = ‘Y’
10. USE_RX_HARVEST_METHOD_YN: IF SCENARIO_HARVEST_METHOD.HARVEST_METHOD = ‘RX’ THEN ‘Y’ ELSE ‘N’ WHERE SCENARIO_ID = SELECTED PROCESSOR SCENARIO
11. STEEP_SLOPE_PCT: SCENARIO_HARVEST_METHOD.SteepSlope WHERE SCENARIO_ID = SELECTED PROCESSOR SCENARIO
12. SCENARIO_HARVEST_METHOD_LOW: IF USE_RX_HARVEST_METHOD_YN = ‘Y’ THEN BLANK ELSE SCENARIO_HARVEST_METHOD.HARVESTMETHODLOWSLOPE WHERE SCENARIO_ID = SELECTED PROCESSOR SCENARIO
13. SCENARIO_HARVEST_METHOD_LOW_ID: IF USE_RX_HARVEST_METHOD_YN = ‘Y’ THEN BLANK ELSE HARVEST_METHODS.HARVESTMETHODID WHERE HARVEST_METHODS. METHOD = SCENARIO_HARVEST_METHOD.HARVESTMETHODLOWSLOPE and HARVEST_METHODS.METHOD.STEEP_YN = ‘N’ and SCENARIO_HARVEST_METHOD.SCENARIO_ID = SELECTED PROCESSOR SCENARIO
14. SCENARIO_HARVEST_METHOD_LOW_CATEGORY: IF USE_RX_HARVEST_METHOD_YN = ‘Y’ THEN BLANK ELSE HARVEST_METHODS.BIOSUM_CATEGORY WHERE HARVEST_METHODS.METHOD = SCENARIO_HARVEST_METHOD.HARVESTMETHODLOWSLOPE and HARVEST_METHODS.METHOD.STEEP_YN = ‘N’ and SCENARIO_HARVEST_METHOD.SCENARIO_ID = SELECTED PROCESSOR SCENARIO
15. SCENARIO_HARVEST_METHOD_LOW_CATEGORY_DESCR: IF USE_RX_HARVEST_METHOD_YN = ‘Y’ THEN BLANK ELSE HARVEST_METHODS.TOP_LIMB_SLOPE_STATUS WHERE HARVEST_METHODS.METHOD = SCENARIO_HARVEST_METHOD.HARVESTMETHODLOWSLOPE and HARVEST_METHODS.METHOD.STEEP_YN = ‘N’ and SCENARIO_HARVEST_METHOD.SCENARIO_ID = SELECTED PROCESSOR SCENARIO

16. SCENARIO_HARVEST_METHOD_STEEP: IF USE_RX_HARVEST_METHOD_YN = 'Y' THEN BLANK ELSE SCENARIO_HARVEST_METHOD. HARVESTMETHODSTEEPSLOPE WHERE SCENARIO_ID = SELECTED PROCESSOR SCENARIO
17. SCENARIO_HARVEST_METHOD_STEEP_ID: IF USE_RX_HARVEST_METHOD_YN = 'Y' THEN BLANK ELSE HARVEST_METHODS.HARVESTMETHODID WHERE HARVEST_METHODS. METHOD = SCENARIO_HARVEST_METHOD.HARVESTMETHODSTEEPSLOPE and HARVEST_METHODS.METHOD.STEEP_YN = 'Y' and SCENARIO_HARVEST_METHOD.SCENARIO_ID = SELECTED PROCESSOR SCENARIO
18. SCENARIO_HARVEST_METHOD_STEEP_CATEGORY: IF USE_RX_HARVEST_METHOD_YN = 'Y' THEN BLANK ELSE HARVEST_METHODS.BIOSUM_CATEGORY WHERE HARVEST_METHODS.METHOD = SCENARIO_HARVEST_METHOD.HARVESTMETHODSTEEPSLOPE and HARVEST_METHODS.METHOD.STEEP_YN = 'Y' and SCENARIO_HARVEST_METHOD.SCENARIO_ID = SELECTED PROCESSOR SCENARIO
19. SCENARIO_HARVEST_METHOD_STEEP_CATEGORY_DESCR: IF USE_RX_HARVEST_METHOD_YN = 'Y' THEN BLANK ELSE HARVEST_METHODS.TOP_LIMB_SLOPE_STATUS WHERE HARVEST_METHODS.METHOD = SCENARIO_HARVEST_METHOD.HARVESTMETHODSTEEPSLOPE and HARVEST_METHODS.METHOD.STEEP_YN = 'Y' and SCENARIO_HARVEST_METHOD.SCENARIO_ID = SELECTED PROCESSOR SCENARIO

Notes: This logic has the possibility of being incorrect. The Processor outputs can be run under different sets of parameters but still be in the output tables for the selected Processor scenario. Processor scenarios can be run by variant/package but Optimizer scenarios include ALL variant packages. ~~An alternative approach would pull the 'Harvesting System' field from the OpCost input database but this would mean a row for every stand in a table. Not sure if this is worth the effort for this possibility. Also, we would have to link back to see if the Harvesting System came from RX or Processor scenario but would have no way to know for sure if both were set to the same system.~~
Add to documentation: caveats and warnings to the effect that if Processor is run with different harvest systems for different packages, only the last harvest system run will show up.
Also, if rx = '999' print NA in harvest method fields. Our assumption is that this is a grow-only prescription. This does not affect BioSum processing/optimizing in any way.

RXPACKAGE_REF_C

	Column name	Descriptive name	Data type
1	RXPACKAGE		Text(3)
2	DESCRIPTION		Text(255)
3	SIMYEAR1_RX		Text(3)
4	SIMYEAR1_RX_CATEGORY		Text(100)
5	SIMYEAR1_RX_SUBCATEGORY		Text(100)
6	SIMYEAR1_RX_DESCRIPTION		Text(255)
7	SIMYEAR2_RX		Text(3)
8	SIMYEAR2_RX_CATEGORY		Text(100)
9	SIMYEAR2_RX_SUBCATEGORY		Text(100)
10	SIMYEAR2_RX_DESCRIPTION		Text(255)
11	SIMYEAR3_RX		Text(3)
12	SIMYEAR3_RX_CATEGORY		Text(100)
13	SIMYEAR3_RX_SUBCATEGORY		Text(100)
14	SIMYEAR3_RX_DESCRIPTION		Text(255)
15	SIMYEAR4_RX		Text(3)
16	SIMYEAR4_RX_CATEGORY		Text(100)
17	SIMYEAR4_RX_SUBCATEGORY		Text(100)
18	SIMYEAR4_RX_DESCRIPTION		Text(255)

1. RXPACKAGE: RXPACKAGE.RXPACKAGE
2. DESCRIPTION: RXPACKAGE.DESCRIPTION
3. SIMYEAR1_RX: RXPACKAGE.SIMYEAR1_RX

4. SIMYEAR1_RX_CATEGORY: FVS_RX_CATEGORY.DESC where FVX_RX_CATEGORY.CATID = RX.CATID and RX.RX = RXPACKAGE.SIMYEAR1_RX
5. SIMYEAR1_RX_SUBCATEGORY: FVS_RX_SUBCATEGORY.DESC where FVX_RX_SUBCATEGORY.CATID = RX.CATID and FVX_RX_SUBCATEGORY.SUBCATID = RX.SUBCATID and RX.RX = RXPACKAGE.SIMYEAR1_RX
6. SIMYEAR1_RX_DESCRIPTION: RX.DESCRIPTION WHERE RX.RX = RXPACKAGE.SIMYEAR1_RX
7. SIMYEAR2_RX: RXPACKAGE.SIMYEAR2_RX
8. SIMYEAR2_RX_CATEGORY: FVS_RX_CATEGORY.DESC where FVS_RX_CATEGORY.CATID = RX.CATID and RX.RX = RXPACKAGE.SIMYEAR2_RX
9. SIMYEAR2_RX_SUBCATEGORY: FVS_RX_SUBCATEGORY.DESC where FVX_RX_SUBCATEGORY.CATID = RX.CATID and FVS_RX_SUBCATEGORY.SUBCATID = RX.SUBCATID and RX.RX = RXPACKAGE.SIMYEAR2_RX
10. SIMYEAR2_RX_DESCRIPTION: RX.DESCRIPTION WHERE RX.RX = RXPACKAGE.SIMYEAR2_RX
11. SIMYEAR3_RX: RXPACKAGE.SIMYEAR3_RX
12. SIMYEAR3_RX_CATEGORY: FVS_RX_CATEGORY.DESC where FVS_RX_CATEGORY.CATID = RX.CATID and RX.RX = RXPACKAGE.SIMYEAR3_RX
13. SIMYEAR3_RX_SUBCATEGORY: FVS_RX_SUBCATEGORY.DESC where FVX_RX_SUBCATEGORY.CATID = RX.CATID and FVS_RX_SUBCATEGORY.SUBCATID = RX.SUBCATID and RX.RX = RXPACKAGE.SIMYEAR3_RX
14. SIMYEAR3_RX_DESCRIPTION: RX.DESCRIPTION WHERE RX.RX = RXPACKAGE.SIMYEAR3_RX
15. SIMYEAR4_RX: RXPACKAGE.SIMYEAR4_RX
16. SIMYEAR4_RX_CATEGORY: FVS_RX_CATEGORY.DESC where FVS_RX_CATEGORY.CATID = RX.CATID and RX.RX = RXPACKAGE.SIMYEAR4_RX
17. SIMYEAR4_RX_SUBCATEGORY: FVS_RX_SUBCATEGORY.DESC where FVX_RX_SUBCATEGORY.CATID = RX.CATID and FVS_RX_SUBCATEGORY.SUBCATID = RX.SUBCATID and RX.RX = RXPACKAGE.SIMYEAR4_RX
18. SIMYEAR4_RX_DESCRIPTION: RX.DESCRIPTION WHERE RX.RX = RXPACKAGE.SIMYEAR4_RX

Note: Prescription harvest method information can be queried by linking to the harvest_method_ref table using the rx value.

DIAMETER_SPP_GRP_REF_C

Note: this table was formerly named DIAMETER_SPECIES_GROUP_REF

	Column name	Descriptive name	Data type
1	DBH_CLASS_NUM ~~DIAM_GROUP~~		Integer
2	DBH_RANGE_INCHES ~~DIAM_CLASS~~		Text(15)
3	SPP_GRP_CODE ~~SPECIES_GROUP~~		Integer
4	SPP_GRP ~~SPECIES_GROUP_LABEL~~		Text(50)
4	TO_CHIPS ~~WOOD_BIN~~		Text(1)
5	MERCH_VAL_DpCF ~~MERCH_VALUE~~		Double
6	VALUE_IF_CHIPPED_DpGT ~~CHIP_VALUE~~		Double

1. DBH_CLASS_NUM: scenario_tree_diam_groups.DIAM_GROUP WHERE SCENARIO_ID = SELECTED PROCESSOR SCENARIO
2. DBH_RANGE_INCHES: scenario_tree_diam_groups.DIAM_CLASS WHERE SCENARIO_ID = SELECTED PROCESSOR SCENARIO. Example: 7.1 - 10
3. SPP_GRP_CODE: scenario_tree_species_diam_dollar_values.SPECIES_GROUP WHERE scenario_tree_species_diam_dollar_values.DIAM_GROUP = DIAM_GROUP AND SCENARIO_ID = SELECTED PROCESSOR SCENARIO
4. SPP_GRP: scenario_tree_species_groups.SPECIES_LABEL WHERE scenario_tree_species_groups.SPECIES_GROUP = scenario_tree_species_diam_dollar_values.SPECIES_GROUP AND SCENARIO_ID = SELECTED PROCESSOR SCENARIO
5. TO_CHIPS: scenario_tree_species_diam_dollar_values.WOOD_BIN WHERE scenario_tree_species_diam_dollar_values.DIAM_GROUP = DIAM_GROUP AND SCENARIO_ID = SELECTED PROCESSOR SCENARIO
6. MERCH_VAL_DpCF: scenario_tree_species_diam_dollar_values.MERCH_VALUE WHERE scenario_tree_species_diam_dollar_values.DIAM_GROUP = DIAM_GROUP AND SCENARIO_ID = SELECTED PROCESSOR SCENARIO
7. VALUE_IF_CHIPPED_DpGT: scenario_tree_species_diam_dollar_values.CHIP_VALUE WHERE scenario_tree_species_diam_dollar_values.DIAM_GROUP = DIAM_GROUP AND SCENARIO_ID = SELECTED PROCESSOR SCENARIO. Set to 0 if TO_CHIPS = 'Y'

FVS_WEIGHTED_VARIABLES_REF_C

	Column name	Descriptive name	Data type
1	VARIABLE_NAME		Text(40)
2	VARIABLE_DESCRIPTION		Text(255)
3	BASELINE_RXPACKAGE		Text(3)
4	VARIABLE_SOURCE		Text(100)
5	weight_1_pre		Double
6	weight_1_post		Double
7	weight_2_pre		Double
8	weight_2_post		Double
9	weight_3_pre		Double
10	weight_3_post		Double
11	weight_4_pre		Double
12	weight_4_post		Double

1. VARIABLE_NAME: calculated_optimizer_variables.VARIABLE_NAME WHERE VARIABLE_TYPE = 'FVS'
2. VARIABLE_DESCRIPTION: calculated_optimizer_variables.VARIABLE_DESCRIPTION
3. BASELINE_RXPACKAGE: calculated_optimizer_variables.BASELINE_RXPACKAGE
4. VARIABLE_SOURCE: calculated_optimizer_variables.VARIABLE_SOURCE
5. weight_1_pre: calculated_fvs_variables_definition.weight_1_pre WHERE ID = calculated_optimizer_variables.ID
6. weight_1_post: calculated_fvs_variables_definition.weight_1_post WHERE ID = calculated_optimizer_variables.ID
7. weight_2_pre: calculated_fvs_variables_definition.weight_2_pre WHERE ID = calculated_optimizer_variables.ID
8. weight_2_post: calculated_fvs_variables_definition.weight_2_post WHERE ID = calculated_optimizer_variables.ID
9. weight_3_pre: calculated_fvs_variables_definition.weight_3_pre WHERE ID = calculated_optimizer_variables.ID
10. weight_3_post: calculated_fvs_variables_definition.weight_3_post WHERE ID = calculated_optimizer_variables.ID
11. weight_4_pre: calculated_fvs_variables_definition.weight_4_pre WHERE ID = calculated_optimizer_variables.ID

12. weight_4_post: calculated_fvs_variables_definition.weight_4_post WHERE ID = calculated_optimizer_variables.ID

ECON_WEIGHTED_VARIABLES_REF_C

	Column name	Descriptive name	Data type
1	VARIABLE_NAME		Text(40)
2	VARIABLE_DESCRIPTION		Text(255)
3	VARIABLE_SOURCE		Text(100)
4	CYCLE_1_WEIGHT		Double
5	CYCLE_2_WEIGHT		Double
6	CYCLE_3_WEIGHT		Double
7	CYCLE_4_WEIGHT		Double

1. VARIABLE_NAME: calculated_optimizer_variables.VARIABLE_NAME WHERE VARIABLE_TYPE = 'ECON'
2. VARIABLE_DESCRIPTION: calculated_optimizer_variables.VARIABLE_DESCRIPTION
3. VARIABLE_SOURCE: calculated_optimizer_variables.VARIABLE_SOURCE
4. CYCLE_1_WEIGHT: calculated_econ_variables_definition.weight WHERE ID = calculated_optimizer_variables.ID and rxcycle = '1'
5. CYCLE_2_WEIGHT: calculated_econ_variables_definition.weight WHERE ID = calculated_optimizer_variables.ID and rxcycle = '2'
6. CYCLE_3_WEIGHT: calculated_econ_variables_definition.weight WHERE ID = calculated_optimizer_variables.ID and rxcycle = '3'
7. CYCLE_4_WEIGHT: calculated_econ_variables_definition.weight WHERE ID = calculated_optimizer_variables.ID and rxcycle = '4'

SPP_GRP_REF_C

	Column name	Descriptive name	Data type
1	SPP_GRP_CD		Integer
2	COMMON_NAME		Text(50)
3	FIA_SPCD		Integer
Note: Only include rows associated with the selected Processor scenario for this Optimizer scenario			

1. SPP_GRP_CD: scenario_tree_species_groups_list.species_group
2. COMMON_NAME: scenario_tree_species_groups_list.common_name
3. FIA_SPCD: scenario_tree_species_groups_list.spcd

TABLES TO COPY WITH NO CHANGES

Name	Location	Notes
SCENARIO_ADDITIONAL_ HARVEST_COSTS	\processor\db\ scenario_processor_rule_definitions.mdb	where scenario_id = selected Processor scenario
rx_harvest_cost_columns	\db\fvsmaster.mdb	
harvest_costs	\processor\scenario_id\ db\scenario_results.mdb	where scenario_id = selected Processor scenario
tree_vol_val_by_species_diam_groups	\processor\scenario_id\ db\scenario_results.mdb	where scenario_id = selected Processor scenario
All tables	\optimizer\db\prepost_fvs_weighted.accdb	
scenario_psites	\optimizer\db\ scenario_optimizer_rule_ definitions.mdb	Where scenario_id = selected Optimizer scenario
FVS PRE and POST tables	\fvs\db	Separate database from other tables

In lieu of including all of the tables in the scenario_processor_rule_definitions.mdb and scenario_optimizer_rule_definitions.mdb, which would be cryptic for most analysts, we will save two text files to the optimizer output directory that summarize the scenario parameters. The name(s) of these files will be processor_params_scenariox.txt and optimizer_params_scenariox.txt where scenario is the name of the scenario.
We will use the same algorithm to select the FVS PRE and POST tables to include in the optional fvs_context.accdb that we use when generating the table list for the Optimizer UI. This approach doesn't restrict the analyst to a particular set of tables.
An exception will be the Additional_kcp_cpa table in processor's scenario_results.accdb. That table should be provided in the optional context.accdb. It is documented as follows:

Additional_kcp_cpa Table

If additional harvest costs are assessed using FVS_Compute fields, the /processor/scenario/db/scenario_results.mdb\additional_kcp_cpa table will be updated. This is the schema for this table.

	Column name	Descriptive name	Data type
1	BIOSUM_COND_ID (Primary key)		Text(25)
2	RXPACKAGE		Text(3)
3	RX		Text(3)

4	RXCYCLE		Text(1)
5	DATETIMECREATED		Text(22)
6	BROADBUR *	Name and number of columns will vary. They depend on the additional cost components configured from KCP output.	Double
7	SCENARIO1 *	Name and presence of column will vary. It depends on if scenario-level additional cost components are configured.	Double

Key type	Column(s) order	Tables to link
Unique	BIOSUM_COND_ID, RXPACKAGE, RX, RXCYCLE	

1. BIOSUM_COND_ID: /OPCOST/Input/Opcost_10_1_5_Input_CA_PXXX_XXX_XXX_XXX_XXX.accdb/ OpCost_Output.stand
2. RXPACKAGE: OpCost_Output.RXPACKAGE
3. RX: OpCost_Output.RX
4. RXCYCLE: OpCost_Output.RXCYCLE

   Note: For stands with no harvest activity biosum_cond_id, rxpackage, rx, and rxcycle come from either the PRE_FVS_COMPUTE or POST_FVS_COMPUTE tables
5. DATETIMECREATED: The date and time of the Processor run where the record was created
6. BROADBUR: This column name is an example. The actual column name(s) depend on the name(s) configured using the KCP files that appear on the FVS_Compute table. The dollar values in these fields include the escalator values for RX years 2 – 4.
7. SCENARIO1: This column name is an example. The actual column name depends on the Processor scenario name. This is an optional column. If scenario-level additional harvest costs are not configured, this column may be missing or blank. The dollar values in this field includes the escalator values for RX years 2 – 4.

ORIGINAL REQUIREMENTS

This is an outline of the requirements as they were originally shared.
PLOT and COND are already included in optimizer results in their filtered versions
Context db should contain:
The harvest method (number and name) applied in each treatment, and whether that was specified at the rx or scenario level
Scenario_additional_harvest_costs for the processor scenario used in the optimization
Scenario_Tree_diam_groups for the processor scenario used in the optimization
scenario_tree_species_diam_dollar_values for the processor scenario used in the optimization
A join of scenario_tree_species_groups_list and scenario_tree_species_groups for the processor scenario used in the optimization
Fvs_master.rx, .rxpackage, .rx_harvest_cost_columns
processor/scenario1/db/scenario_results.mdb:

    Harvest_costs
    Tree_vol_val_by_species_diam_groups

optimizer/db/prepost_fvs_weighted.accdb

    all tables

optimizer_definitions

    all tables

FOR the scenario just run (where current=Y): optimizer_rule_definitions to include

    Relevant rows in every populated (for this scenario) table

At user option, a PRE_POST database can also be provided as a third database, containing PRE and POST tables for CUSTOM, POTFIRE, SUMMARY, COMPUTE and STRCLASS