Streamlining Vegetation Analysis for Environmental Impact Statements Using GIS and Python Scripting

A practicum report by:
Clay B. Stephenson
Cbs332@nau.edu

Completed in requirements for the Degree of Master of Science in Geography,
Department of Geography (Program of Study 2022-23)

&

Graduate Certificate in Geographic Information Systems,
Department of Geography (Program of Study 2022-2023)

Norther Arizona University (Online)

Flagstaff, AZ

Spring 2023

Advisor

Dr. Ruihong Huang

Committee Members:

Dr. David C. Folch
&
Dr. Erik Schiefer

Completed in cooperation with the Montana Department of Natural Resources and Conservation,
Forestry an Trust Lands Management Division.

Abstract

In the western US, state land management agencies will often manage their jurisdiction areas for multiple uses, with timber management being a focus in northwest Montana. Forest management aims to achieve multiple objectives, including the mitigation of wildfire risk, improving forest stand health, preservation of wildlife habitat, providing recreation opportunities, and return of economic value from forest products. When successful, the products of this management can include revenue for state trust funds as well as funds to complete the rehabilitation of forest ground and landscapes leading to increased sawlog volume production, reduced wildfire risk, and overall forest health. To complete these management activities, environmental analysis is usually completed to document potential impacts on the environment and provide an avenue for public involvement in land management that fulfills agency objectives for lands that are an important resource. Quantifying potential impacts on the vegetation structure of forest landscapes is time-consuming and requires a deep knowledge of forestry and Geographic Information Systems (GIS) to complete basic analysis. In this practicum, a variety of techniques were used to create an easily accessible and operational geoprocessing toolbox that forestry practitioners can use to complete specific vegetation analysis for environmental processes like environmental impact statements. The end goal for the geoprocessing toolbox was to decrease time spent while completing vegetation analysis and to decrease potential inconsistencies in results from the analysis completed by different team members. Upon completion, the resulting geoprocessing toolbox was able to sufficiently reduce the amount of time required to complete the base amount of vegetation analysis required for most environmental impact statements. In addition to making the vegetation analysis process more succinct, the tools allow foresters to run multiple scenarios with alternative proposed harvesting treatments to quickly see potential impacts to the landscape that they are managing. Overall, the streamlined vegetation analysis toolbox allows foresters to complete required environmental review more quickly, more accurately, and gives them the ability to develop stronger alternatives for consideration in an environmental impact statement.

Table of contents

Introduction

The landscape in the mountainous region of northwestern Montana largely consists of a variety of forested stands, surrounded by steep rocky mountain peaks at higher elevations, and grassy timbered valleys at lower elevations. The Swan Valley, located in this region, contains landowners of all types, ranging from federal and state governments, as well as private entities. Historically the Swan Valley has undergone change through forest management projects, prescribed fire, as well as natural disturbances in the form of wildfires. The state government in Montana has experienced and been a part of this change since the divestment of certain state lands from the federal government when Montana became a state in 1889 as part of the Omnibus Enabling Act (Wiles, 2017). Because of this act, Montana was granted sections sixteen and thirty-six in every township to support the common schools (Wiles, 2017). When these sections were otherwise unavailable at the time of the Enabling act, other sections were selected in lieu of these lands (*Understanding Trust Lands*, 2023) and some land was blocked to create management areas known as state forests. The Swan River State Forest was designated in 1925 in the Swan Valley and over many years and changes, the governing body of these Common Schools' lands became the Montana Department of Natural Resources and Conservation (DNRC) (Stockwell, 2013).

The mission of the DNRC in the Swan Valley is to sustainably generate revenue for the Common Schools' Trust Fund while completing forest improvement projects like forest regeneration, pre commercial timber thinning, and infrastructure improvement. To complete forest management projects that could have potential effects on the environment, DNRC is required to follow the Montana Environmental Policy Act (MEPA) and the State Forest Land Management Plan (SFLMP) by completing an environmental review of proposed projects. This generally takes the form of environmental assessments (EAs) and Environmental Impact Statements (EISs) within the DNRC. This type of environmental review gives project leaders and Interdisciplinary teams (ID teams) of forest practitioners the chance to interact with the public and ensure that environmental issues are being considered in the development of forest management projects.

On an individual project level, ID team members are faced with deadlines that make it difficult to complete the required fieldwork and data processing that is required for a quality environmental analysis. Most of the preparation time for an EIS is spent collecting data in the field and completing vegetation analysis by manipulating the data with individual geoprocessing tools and resulting summary tables. Using GIS, spatial analysis, and Python scripting, this project will create ways to quickly

summarize, and complete analysis based on potential forest management activities set by ID team members. The practicum will result in a geoprocessing toolbox that ID team members can utilize to help them streamline vegetation analysis. Though ID team members might want to tweak analysis in future projects, the project intends to allow them to consistently complete vegetation analysis efficiently which would free up more time to develop other aspects of the environmental process.

Background behind environmental analysis/scope of project

The Swan River State Forest covers approximately 56,000 acres adjacent to other management agencies and the potential effects of management must be analyzed following MEPA. To complete environmental analysis on a proposed forest management project in Montana, a forestry-focused project leader, and experts in wildlife, hydrology, soils, and forest/environmental policy gather to form an ID team. In the initial stages of project development, the project leader utilizes aerial photography, geographic information systems (GIS)[1], and geospatial data[2] to determine where to begin reconnaissance on forested stands. Walkthrough forest stand data is then collected and fed into a timber inventory dataset that shows current conditions within the stand. Once current conditions are established, potential alternative forest management treatments can then be developed. Multiple treatments can be identified for varying alternatives based on the constraints set by the decision maker, as well as issues brought forth from the public involvement process. When alternative treatments are identified, current and post-harvest vegetation conditions are completed to document potential environmental effects.

Because managed forest landscapes are dynamic, this analysis is time-consuming to complete and requires great attention to detail to ensure that an EIS is factual and produces the best picture of potential environmental effects. Information can change from one iteration of treatment to the next which can lead to inconsistencies in the summary of changes to post-treatment conditions between alternatives. To minimize these inconsistencies and make analysis more efficient, Python scripting can be utilized to create a custom vegetation analysis toolbox. With tools from this toolbox, ID team

[1] Geographic information systems (GIS): a system that creates, manages, analyzes, and maps all types of data (Esri, 2023a).
[2] Geospatial data: data that typically combines the location and attributes information about characteristics of an object.

members can define multiple alternative treatment inputs and the resulting summary tables and changes to the environment can be computed automatically.

This project created the vegetation analysis toolbox with the intent to give basic users of GIS and forestry professionals an easy-to-use toolbox where no knowledge of Python scripting would be needed. This allows users to execute complex and interrelated vegetation analysis that would normally be carried out by hand.

Literature review

The following literature review describes the intent of the project and backs up some of the core concepts that will be utilized during the creation of the vegetation analysis toolbox/ geoprocessing tools. It focuses on aspects of how and why efficiency can be found by using scripting tools for spatial analysis as well as the importance of good data management and documentation. Aspects of forest management must also be addressed even though the central theme of this report is on the creation of the tools themselves, and key points are shown for why and how forest attribute data is manipulated by the geoprocessing tool.

GIS and Python scripting language coupled together provide a platform that increases the efficiency of spatial analysis. Esri ArcGIS version 10 has been fully integrated with Python scripting using the arcpy site package (Rees, 2014) and gives GIS users the ability to create Python scripts for custom geoprocessing tools. This site package contains support and functions for individual geoprocessing tools that would normally be completed individually by hand through the ArcGIS Desktop or ArcGIS Pro interface. In 2000, Python 2, a coding language originally developed in the 1980's was released and intended to be utilized a simple and easy to use language for web development, data management/machine learning, and build efficiency into computer aided workflows (Saabith et al., 2019). Saabith et al., (2019) also explains that in 2008, Python 3 was released with intention of removing duplicate programming constructs and modules so that there would be one "obvious" way to complete workflows. Though ArcGIS version 10 was fully integrated with the Python scripting language, ArcGIS Pro utilizes the Python 3 scripting language and the geoprocessing tools created in this project will not be backwards compatible with ArcGIS desktop version 10.x (Esri, 2023b). The ability to create custom tools for specific processes was the basis for increasing the efficiency of vegetation analysis. For this project, the ArcGIS Arc Pro script tool was used where user input parameters could be defined and run through a

Python script that outputs the desired analysis. To many forestry practitioners, it is often hard to determine how to start when utilizing GIS for spatial analysis. Some practitioners Have a basic understanding of how analysis is completed, but clarity of mission and training is not always available when moving from an operational position to an analyst position. This can be attributed to the rapid change that GIS has undergone in recent years and the need for users to adapt GIS to meet their needs on the ground (Ricker et al., 2020). Practitioners can get caught up in thinking at the individual project level, and it takes time to understand which geoprocessing tools are appropriate to complete analysis. Utilizing Python scripting for vegetation analysis and strategic planning is possible and allows for a reduced need for manpower to input and manage the data (Standovár et al., 2016). Custom geoprocessing tools can also be created that can handle tasks specific to an organization (Pimpler, 2015). By using the techniques described by Pimpler and those gained through general forestry practice, vegetation analysis for environmental processes can be created.

When completing complex processes like environmental analysis, GIS can be applied as a tool, and initial Interdisciplinary (ID) team composition must be discussed to ensure teams have the necessary skills to achieve the intended goals of the project (Ricker et al., 2020). Because the traditional forestry focused ID team members are leading development of EIS's on the Swan River State Forest, a certain level of expertise in forestry can be expected for end users of the resulting geoprocessing tools. Ricker et al. (2020) also explains a difference between three types of GIS users that fit well with the project; the GIS tool user, the GIS toolmaker, and the GIScientist. Those that can navigate the GIS environment and implement processes that are developed by both the GIS tool maker and GIScientist is recognized as the GIS tool user. This project aims to create a product that can be easily utilized by the GIS tool user. A product of data and method documentation can give the GIS tool user the reference materials needed to understand why the tool is being produced and will create a jumping off point for further development of custom geoprocessing tools in the future. GIS for forest management can also be incorporated into a strategic planning and modeling process that involves making predictions about what the future forest will look like relative to management activities, and future development is needed to support choices in resource allocation (Sonti, 2015). In addition to using GIS to create geoprocessing tools designed to complete analysis, foresters and ID team members must understand why environmental analysis must be carried out in the state of Montana, the characteristics of forest attributes and the data that the tool is manipulating, as well as the documentation and organization of the data once the tool is utilized.

As technology advances and subdisciplines of science become more defined, several efficiencies could be created for traditional workflows that would once be impractical or impossible. A core goal of this project was to create a workflow for ID team members that would allow them to increase the consistency of required vegetation analysis using custom geoprocessing tools. This goal can be broken down into two varying concepts of reproducibility and replicability. Reproducibility is defined as obtaining consistent results using the same input data; computational steps, methods, and code; and conditions of analysis (National Academy of Sciences, 2019). The National Academy of Sciences (2019) also defines replicability as obtaining consistent results across studies aimed at answering the same scientific question, each of which has obtained its own data. For the purposes of this project, reproducibility is the end goal, where members of an ID team can utilize the same data and workflow process with the geoprocessing tools that result in consistent change in acres for desired forest attribute types. Replicability, though not directly examined in this project, can also be tested in the future where similar workflow can be implemented in a different study area and trends in the forest attribute acres can be examined. The expected result for reproducibility in this project would be to get the same exact acres of classified forest attribute after the analysis is completed by the vegetation analysis toolset, regardless of who is running them. In the future, the expected result for replicability would be that examiners could recognize the same trends of change in acreage when using the vegetation analysis toolset, regardless of the state land that the toolset is being completed on.

This project also aims to incorporate ideas to increase the level of access and usability of data to augment the decision-making process for forest management on the ground. It is key in analysis where reproducibility is a primary objective, that good data management must overcome obstacles where designed workflows and the data used within those workflows are inappropriately described and managed so that future "users" would not be able to use them without great effort. The concept of the findable, accessible, interoperable, and reusable (FAIR) guiding principles is described where data, code, and workflows must be clearly described with adequate metadata, be open and universally implementable, use broadly applicable language, and be released with a clearly described attributes (Wilkinson, 2016). Foresters at a state level have access to custom datasets that contain in-depth attributes about forest inventory, health, and other factors that may be affected by intended forest management. Often this type of data is utilized multiple times and is produced by one individual and used by another in an environmental analysis setting (Kim, 1999). Because vegetation analysis requires

foresters and analysts to quantify changes to acreage, there is potential for multiple copies and trials of datasets to be created, as GIS tool users work their way through alternative development. FAIR data management guiding principles can be included in this project through the use of good data management and documentation through metadata. This ensures that data is not misused and allows GIS tool users to understand the history and intent of the data they are manipulating (Kim, 1999). As project complexity increases and multiple scenarios are generated, proper documentation will provide a better understanding of the data, and how that data can be utilized in the future (Kim, 1999).

The Montana DNRC follows the Montana Environmental Policy Act (MEPA) to complete forest management projects that generate revenue for state-owned trust funds. The most common on the Swan River State Forest being the Common Schools Trust Fund which provides revenue to the K-12 school system in Montana. MEPA has been described as an innovative provision for environmental impact statements on "major actions of state government significantly affecting the quality of the human environment" (Stockwell, 2013). In short, a provision adhering to guidance from MEPA requires ID team members to quantify potential effects on the forested landscape because of direct forest management. Alternative actions for management must be developed, and relevant and accurate descriptions of the affected environment should be provided (Stockwell, 2013). This guiding policy provides the background for why foresters and ID team members must quantify potential changes to the environment as a result of forest management activities.

To adhere to MEPA and the effects of forest management on the environment, this project looked at common types of vegetation analysis used in previous forest management environmental impact statements. The change in forest attributes detailed in this research can be adapted to different types of vegetation analysis based on the needs of the ID team and the data available at the time. Vegetation analysis generally consists of describing the structure, development, distribution, and definition of plant communities (Causton, 1988). These types of analysis can be described easily using GIS that include strategic planning, map production, fire management, harvest planning and resource management through forest inventory (Sonti, 2015). The forest attribute data used in this project describes what characteristics of forested stands have been measured and the post treatment changes are easily quantified by the change in acreage that are represented by these characteristics. The project aimed to use a quantitative approach in that forest changes can qualitatively described but the acreage

of these measures is reported in environmental analysis, and any forest attributes from the state's forest attribute data can be used specifically for this type of analysis.

This literature review is aimed to highlight important aspects of the project that will be considered in the following methodology and should be implemented when GIS tool users go to run the geoprocessing tools created in this project. Moving forward, more research could be conducted on utilizing remote sensing and other forest data collection techniques to quantify attributes of the landscape. Scripts created in this project could then be modified to utilize these new datasets with the intent of creating an updateable and dynamic geoprocessing tool that can be used for future environmental impact statement analysis. Each study has merit, but for the time being, vegetation analysis using Montana's timber inventory must be completed to serve as a starting point for more complex analysis that utilizes different data sets in the future. To complete this research, the following methods will be implemented, and results shared.

Problem statement/ research question

How can forestry-focused members of an interdisciplinary team (ID team) efficiently streamline vegetation analysis for complex environmental impact statements before forest management?

Objectives/goals

This project is designed to accomplish the following objectives for foresters and ID team members working on an environmental impact statement anywhere on state trust lands within the state of Montana where state forest inventory data is available.

a. Reduce the amount of analysis that needs to be completed by hand by team members.
b. Create a toolbox that produces desired summary tables, and feature classes for acres of change of appropriate forest attributes post-harvest conditions.
c. Create a document that details the workflow/processes behind vegetation analysis.
d. Create a document that details adequate collection of data and what is expected for a data freeze[3].

[3] Data freeze: in the context of this project, a data freeze is any dataset that is inherently dynamic but must be static for use in the created geoprocessing tools. An example of this would be a standard forest inventory;

To effectively complete the above objectives and streamline analysis for an environmental impact statement, six forest attributes will be examined and utilized to create custom geoprocessing tools. Python scripting and the ArcGIS script tool will be used to create and export data tables describing the current condition and post-harvest conditions for user specified areas within the Swan River State Forest (SRSF). Though there are countless forest attributes that can be considered when examining post-treatment effects of forest management, this project will only look at attributes that are consistently examined with environmental impact statements on the SRSF. Though these tools were developed with a case study on the SRSF, the resulting geoprocessing tools are intended to be utilized on any major state vegetation analysis project. It should be noted that analysis for an EIS is driven by public comments and objectives are set by a decision maker. It is out of the scope of this project to develop a tool that is dynamic enough to take into consideration different forest attribute analyses because of issue development at the public scale. The chosen attributes are commonly analyzed at the ID team level due to their importance in forest structure and often these analyses address common responses received from public comment. Future development of the resulting geoprocessing tool and Python scripts will be needed to incorporate a more flexible level of analysis to include potential issues that are brought forward from public comment. The expected result for the project is to produce a tool that can provide consistency and reproducibility of vegetation analysis for future EISs. These geoprocessing tools will be useful to state land managers and ID team members while serving as a baseline for forest stand vegetation analysis that can then be modified and added to in the future to facilitate new and Important analysis.

Summaries of the following six forest attributes[4] will be exported to Excel tables describing the current condition and post-harvest conditions for specifically identified project areas within the SRSF.

1. Forest habitat type group
2. Desired future forest condition
3. Current forest cover type

Generally, forests change at some scale, but analysis must be carried out on a static snapshot of forest conditions. All input datasets must be part of the data freeze to ensure accurate results.

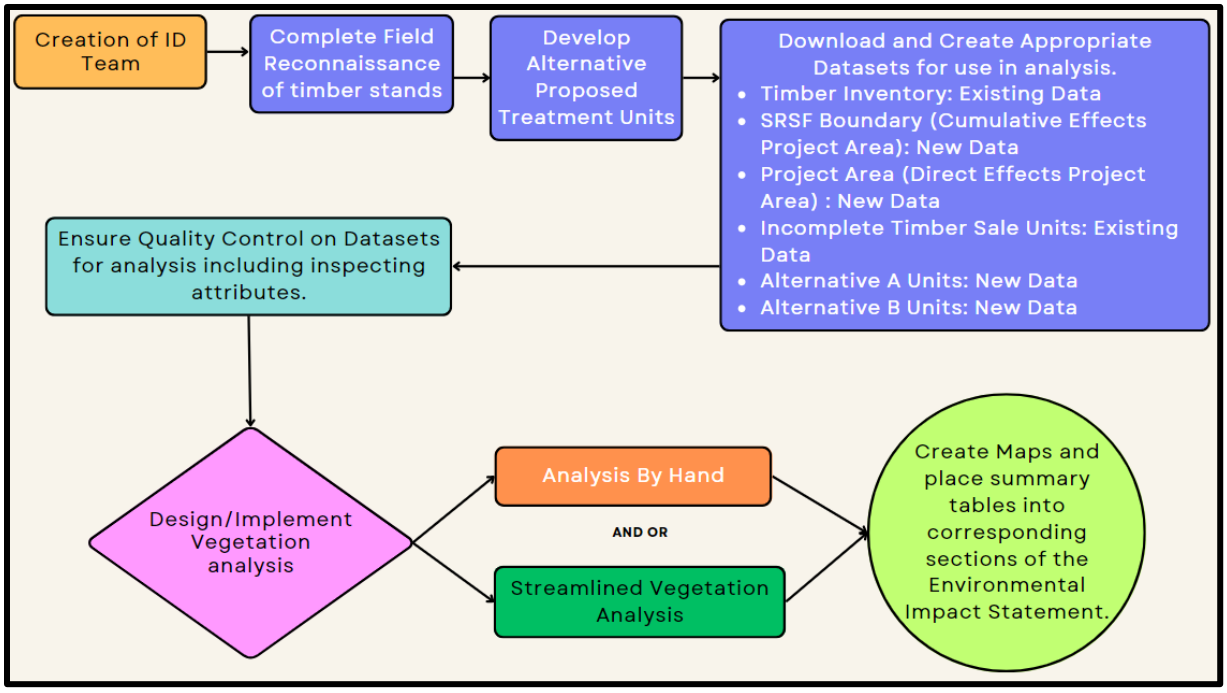
[4] Individual forest attributes are described in the methods section of this report, and a succinct list of attribute fields can be found in *Appendix A* and *Appendix B* at the end of this report. Data documentation and sources can be found in *Data Citation* section at the end of this report.

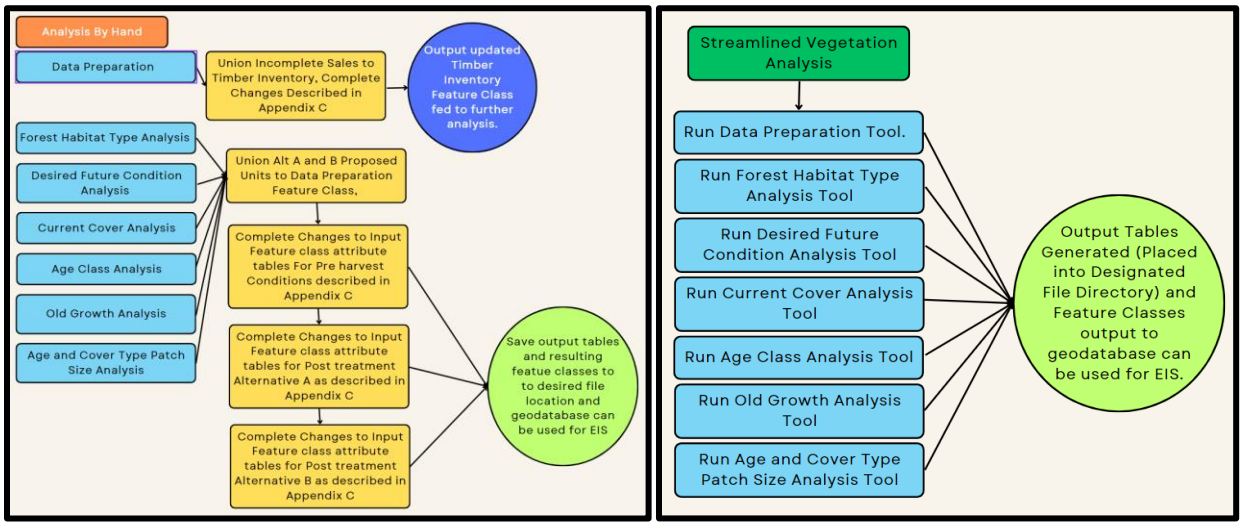
4. Forest age class
5. Forest old growth
6. Age and cover type patch size

As an added goal there is an expectation that when the toolbox is complete, ID team members will be able to run spatial analysis multiple times to show potential effects to the forest environment when determining management alternatives for the project. These scenarios will provide an opportunity for foresters to experiment, recognize patterns, and more closely adhere to objectives set by the decision maker as they alter forest management practices within the scope of the environmental impact statement. Inconsistencies will be minimized by reducing the process to defining input datasets only. This will give a more methodical nature to vegetation analysis and will allow practitioners to spend more time out on the landscape where they can build a more holistic understanding of the ground they are managing. The result is expected to be a more informed decision for forest management treatment on the ground.

Overall workflow

Before describing the overall methods used to complete this project, a basic workflow for environmental impact statement vegetation analysis was created. This workflow serves as a roadmap for completing analysis both by hand, and when utilizing custom geoprocessing tools created in this project. The second workflow shows the difference in completing analysis by hand and when utilizing the streamlined vegetation analysis. This is the intended workflow for completing analysis. Detailed workflows for how the tools are shown in the methods section.





Figures above show basic workflow to follow for environmental impact statement. Streamlined vegetation analysis follows the same workflow as analysis by hand but will do so with Python scripting and only require end users to input cumulative and direct effects updated timber inventories and alternative treatment units.

Methods:

To complete this project, several geoprocessing tools were developed to streamline the vegetation analysis for environmental impact statements on the Swan River State Forest. The tools were created in the ArcGIS Pro environment and their uses were intended as follows:

1. **Vegetation analysis data prep:** Prepared data for use in further geoprocessing tools
2. **Forest habitat type group to Excel:** Summarized the change in acreage based on management type. Outputs to an Excel file.
3. **Desired future forest condition to Excel:** Summarized the change in acreage based on management type. Outputs to an Excel file.
4. **Current forest cover type to Excel:** Summarized the change in acreage based on management type. Outputs to an Excel file.
5. **Forest age class to Excel:** Summarized the change in acreage based on management type. Outputs to an Excel file.
6. **Forest old growth to Excel:** Summarized the change in acreage based on management type. Outputs to an Excel file.
7. **Age and cover type patch size to Excel:** Summarized the change in acreage based on management type. Outputs to an Excel file.

Each of these geoprocessing tools were created to run independently of each other and output the desired results into a user specified file location. This allowed end users to develop and run unique scenarios that affected changes in acreage of forest attributes. The intended result in a real analysis process would allow foresters and ID team members to change recommendation of treatment based on the outcome of the geoprocessing tools. Each of these tools were created and described in this methods section of the report.

Project area development & hypothetical unit creation

Forested landscapes of northwestern Montana are generally dynamic, and data for individual project areas show snapshots of what the landscape looked like at the time of collection. In this research, historical datasets like the state of Montana's forest inventory were utilized to complete the analysis. To begin, two areas of interest were identified and used in this project for analysis. This closely follows Swan River State Forest protocol where a larger area is examined for potential effects to a

cumulative environment, and a finer look at potential effects at a smaller project area level. To complete this, a cumulative effects area (CumulativeEffects_Boundary_20230221[5]), was created that included all acreage (both State owned and adjacent ownerships) within the confines of the SRSF boundary. A direct effects area (DirectEffects_Boundary_20230221[5]) was then created that was roughly centered within the SRSF and followed major road systems traditionally utilized by logging traffic and the north and south bounds of the SRSF boundary. The direct effects project area was chosen to cover a variety of forested stands, that contained all attributes available within the timber inventory dataset. Creation of the direct effects project area also considered normal operations of forest management projects by following the bounds of existing road systems and is shown in the map below (Map 1). Both project areas were selected to mimic standard preparation practices for project development that foresters and ID team members would carry out prior to vegetation analysis on an EIS.

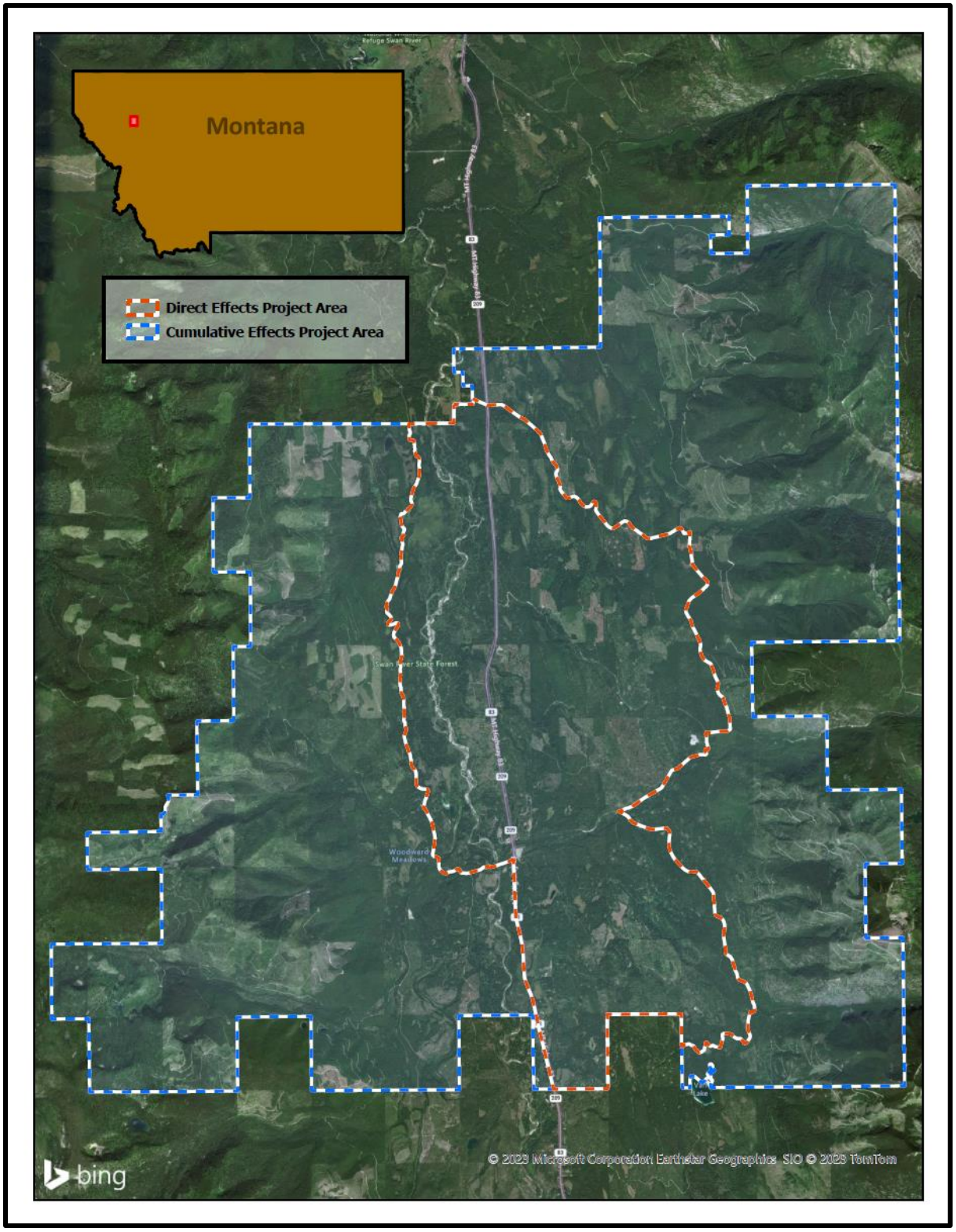
To easily compare the results of a traditional analysis workflow and the streamlined workflow created in this project, hypothetical incomplete timber sale units (Incomplete_Units_20230221[5]) were created. Proposed alternative Alpha and Beta units (PTU_AltA20_230221, PTU_AltB_20230221[5]) needed to be created as well to test the resulting geoprocessing tools. The incomplete timber sale units represent open or incomplete timber sales that would be currently under contract that might have a potential effect on current condition of forest structure once the sales are finished but before the life of the current EIS would expire. The incomplete sale units were selected by creating a point in the ArcGIS Pro and using the buffer tool to set the circle's radius to 7,398.6 feet. By using the buffer tool and setting the circles radius as the buffer distance, a perfectly circular incomplete timber sale unit was created with an area of 100 acres. It should be noted that any polygon shape could be utilized when running the tools and it is expected that foresters would create irregular treatment areas to follow stand boundaries and features on the ground. The size, shape, and orientation of the incomplete timber sale units do not affect the performance of the geoprocessing tools. Ten, 100-acre circular units were created this way and then distributed across the landscape to include only forested areas. Incomplete timber sale locations were also selected to be completely within the cumulative effects analysis area, but outside of the direct effects analysis area. Once the incomplete timber sale units were positioned, key attributes were added including a silvicultural prescription, cutting unit number, and insect and disease risk rating. These fields coincide with state proposed timber sale units (PTU's) and utilize the same domain as the

[5] Feature Class used as Geoprocessing Input, See *Appendix A* for details on data and *Appendix B* for details on attribute fields used in Python scripts.

PTU's to ensure that the geoprocessing tools could be utilized for live state data later. Incomplete timber sale boundaries are always included when preparing data for vegetation analysis in traditional workflows.

The alternative Alpha and Beta units were then created and represent forest treatment projects that would be traditionally proposed by foresters. The same methodology of creating the 100 acre circle units in the incomplete timber sale units for the alternative Alpha and Beta treatment areas was followed, and each of these units fell completely within the direct effects analysis area. Locating these proposed treatment units this way follows a real-world example of how units would be located, apart from the circular units not generally falling along stand boundaries. For this report, these circular areas give a quick visual representation of how the geoprocessing tools behave after geoprocessing is complete. These areas provided a baseline for vegetation analysis and are the source for potential effects on the environment within the cumulative effects and direct effects areas. The incomplete units, alternative Alpha units, and alternative Beta units were created with the intention of being input datasets for the Initial data freeze, as well as the suite of vegetation analysis geoprocessing tools from this project.

The creation of hypothetical units was intended to follow a similar process that foresters and ID team members would follow to complete the very initial stages of project development. During a non-hypothetical project, the units can vary greatly from project inception and might not fall within the parameters set by the project leader, or decision maker. The ID team would then need to go through the analysis to ensure that the total acres that were proposed match the total acres being treated within the project area. There is opportunity to increase the efficiency and robustness of the geoprocessing tools to account for these errors, but for this project and to immediately utilize the proposed tools, ID team members using the veg analysis geoprocessing toolbox will need to ensure that their units fall completely within the direct effects and cumulative effects project areas and that their units do not overlap. This could cause discrepancies in reported acres when adding up acres of treatment units, but the geoprocessing tools would not account for the overlapped acreage.



Map 1 shows the cumulative effects and direct effects project area for the streamlining vegetation analysis project. Areas within the cumulative effects area consist of state ownership and adjacent landowners including private, United States Forest Service, and Montana Fish Wildlife and Parks. Analysis was only conducted on state ownership areas.

Data preparation tool

The Swan River State Forest is unique among state managed forest land in that it is one of three blocked up management "State Forests" that is managed exclusively for the sustained economic benefit to state owned trusts. The forest is under constant change due to forest management and timber harvest activities, and generally those activities are planned, and environmental analysis is completed on a three-year cycle before the next round of projects are analyzed for. Before analysis for the next forest management cycle can begin, some preparation of the input dataset (Timberinventory_20220223) must be prepared so that forested stands on the landscape represent the most up-to-date potential attributes. These forest attributes would be changed due to these previous forest management activities. This preparation accounts for current forest management projects that are under way but have not been completed where the effects of this harvest would then be input into the timber inventory dataset.

The baseline timber inventory dataset[6] was downloaded from a state server to a local file geodatabase that contains all input datasets for the geoprocessing tools. Notes were added to the metadata detailing the date of download and set use limitations to ensure that future users would know the proper use of the data as shown in figure 1. It should be noted that the described metadata process was followed for all input datasets, as well as all created datasets from custom geoprocessing tools. Not every metadata alteration was noted in each methods section from here after. Additionally, the Python script in the tools does not update the metadata automatically, and it is up to the user to identify when it is appropriate to complete the final metadata update. In order to continue to implement a FAIR guideline for data management, future iterations of these geoprocessing tools can incorporate metadata information that is coded into the embedded scripts that will automatically update created feature classes metadata to describe the process that was carried out in the tool itself.

[6] A vector based multi polygon dataset that contains forest attributes contained within each of the polygons. Detailed information on the dataset can be found in *Appendix A*.

TimberInventory_20230221

Type File Geodatabase Feature Class

Tags FMB, DNRC, TLMD, Timber

Summary

Forest Management Bureau's Timber Inventory. Also called the Stand Level Inventory (SLI).

Description

The Timber Inventory holds data about each timber stand on DNRC Trust Lands. There are about 170 fields describing the stands.

**Coal Creek Forest: in April 2020 the entire Coal Creek Forest was replaced with the April 2018 geometry to better fit the GPS'd monuments on the ground. The attributes from 2020 were joined to the geometry of 2018.

G. Mazza 4/29/2020

-Downloaded from K:\TLMD\TLMD_Data\FMB_Timberinventory on 1/03/2023

-Placed into Local File .GDB for use in Streamlining Vegetation Analysis Practicum

C. Stephenson 1/03/2023

Credits

Photo: http://dnrc.mt.gov/divisions/forestry/forestry-assistance/forest-in-focus/collaboration-and-partnerships

MT DNRC

Use limitations

To be used for the Streamlining Vegetation Analysis Geoprocessing tools only. This featureclass should remain in this state for the rest of the project. Any edits to this must be do a copy of the dataset.

Extent

West	-116.160912	East	-103.731335
North	49.117780	South	44.382253

Scale Range

Maximum (zoomed in)	1:5,000
Minimum (zoomed out)	1:150,000,000

Figure 1 shows a sample metadata for input datasets. As datasets are downloaded, manipulated metadata must be edited to reflect the changes being made to the dataset and the intended use of the resulting changes.

After downloading the dataset, a Python script was created in a central custom toolbox called Veg_Analysis.atbx using the ArcGIS Pro script creation tool. The basic workflow of the Python script is shown in the conceptual model in figure 2 below.

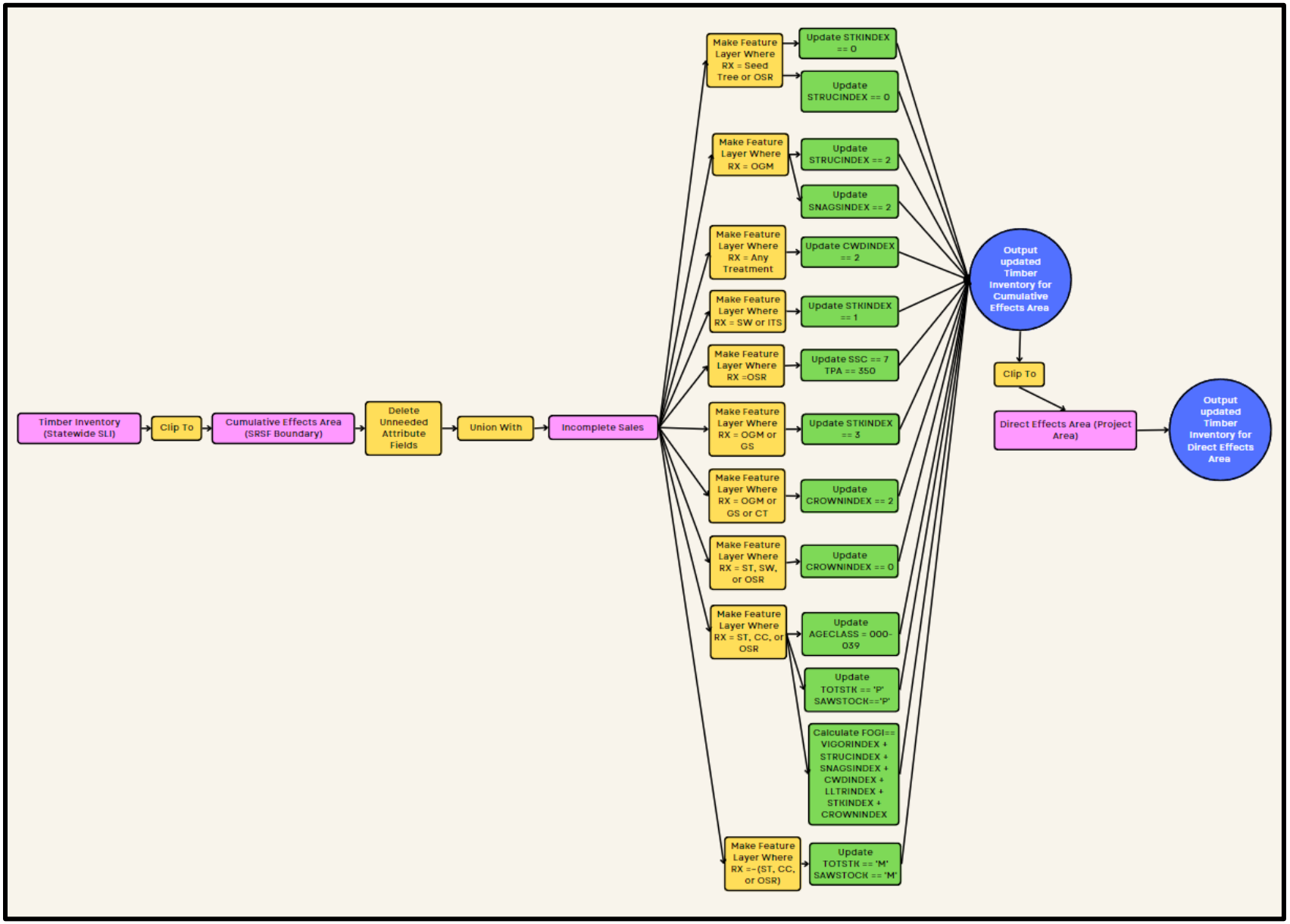


Figure 2 shows how timber inventory and proposed harvest units are manipulated before using the data in vegetation analysis. Raw data outputs (Shown by the blue circles) will be the basis for inputs in all vegetation analysis tools.

A general name, label name, and toolbox description were created which were then stored in the metadata for the tool itself. Five user input parameters were defined for the tool and were detailed as follows:

- **Direct effects project area (project area):** User input, data type: feature layer. Intended to be the (DirectEffects_Boundary_20230221) input dataset.
- **Cumulative effects project area (analysis area):** User Input, data type: feature layer. Intended to be the (CumulativeEffects_Boundary_20230221) input dataset.
- **Timber inventory:** User input, data type: feature Layer. Intended to be the (Timberinventory_20230103) input dataset.
- **Incomplete sales:** User input, data type: feature layer. Intended to be the (Incomplete_Units_20230221) input dataset.
- **Output location (enter project name):** User input, datatype is feature class. Allows users to input descriptive text showing script run number or other. For this project input followed level 1 of naming convention[7]. Also allows user to select file geodatabase location for output datasets of the tool.
- **Input date:** User input, data type is date. Allows users to select date of run for the geoprocessing tool. User selects date from calendar feature.

Once user inputs were defined, a Python script was written in the execution window of the ArcGIS Pro scripting tool properties as shown in *Appendix D: Geoprocessing Tool Python Script1*. The Python script functions by first taking the user input parameters for date and restructuring the characters of the resulting string to create a variable that implements the level 3 naming convention[8]. The tool clips the timber inventory datasets to the cumulative effects project area and producing the (ResultsRun_CEtimberinvnetory_20230221) feature class. By using this resultant feature class in further portions of the script it was ensured that further geoprocessing tool runs only were looking at timber inventory data within the desired project areas.

The resulting cumulative effects timber inventory was then passed to a delete field function where attribute fields that would not be utilized in further analysis were deleted. This process made it easier to inspect the resulting feature classes attribute table and minimized the potential for incorrect

[7] Level 1 of naming convention requires user's initials. See naming convention found in *Appendix A*.
[8] Level 3 of naming convention requires user's run date. See naming convention found in *Appendix A*.

fields to be summarized later. A union function was then completed between the incomplete timber sales input and the cumulative effects area timber inventory which maintains the extent, boundaries, and attributes of both feature classes, and a resulting feature class called (ResultsRun_UpdatedCETimberinventory_20230221) was created.

Once the updated cumulative effects timber inventory feature class was created, it was then utilized to make a temporary feature layer. A where clause was implemented when creating the feature layer to select specific incomplete timber sale silvicultural treatments that would affect post-harvest condition. The calculate field function was then used to calculate the appropriate attribute field to a new value that would represent the appropriate post-harvest conditions. Domains were enforced to ensure that coded value fields would return easy to understand text strings in the attribute table after the tool was completed. The silvicultural prescription descriptions and the forest attribute alterations are described in detail in *Appendix C: Vegetation Analysis Workbook Reference*. Because the make feature layer functions were passed the ResultsRun_UpdatedCETimberinventoryory_20230221 dataset every time a new calculate field method was required, the updated cumulative effects timber inventory dataset attribute fields were updated, and the final feature class is overwritten in the user selected file geodatabase. Temporary feature layers are not written to the disk and are not saved after script is ran. The feature class shows all the potential harvest effects on timber inventory within the cumulative effects analysis area based on the silvicultural prescriptions of the incomplete timber sale units.

To finish off the vegetation analysis data prep geoprocessing tool, the final updated cumulative effects timber inventory dataset needed to be passed to another clip function that clips the feature class to the direct effects project area polygon. The direct effects area should always be smaller than the cumulative effects project area and will always fall within the confines of the Swan River State Forest boundary. At a state level, the same methods should be utilized, where the cumulative effects area is always larger and completely contains the direct effects project area. Traditionally this is checked at the ID team level before analysis begins. Any incomplete timber sale units will generally not be under the new direct effects area. By waiting until after all harvest updates are complete, the clip function for the direct effects area represents the most up to date timber inventory available. The vegetation analysis data prep tool results in two unique feature classes that were input into further geoprocessing tools as described below:

- **Cumulative effects updated timber inventory:** Tool output, data type: feature layer. Intended to be used for inputs into further summary table geoprocessing tools. (ResultsRun_UpdatedCETimberInventory_20230221) input dataset.
- **Direct effects updated timber inventory:** Tool output, data type: feature layer. Intended to be used for inputs into further summary table geoprocessing tools. (ResultsRun_UpdatedDETimberinventory_20230221) input dataset.

From here on, users will now be able to run any combination of the following geoprocessing tools created by the project. For the resulting feature layers to be compatible with further processing tools, the silvicultural treatment field (Rx) was deleted from the feature classes so that new treatment units can be added later with an Rx field and change in forest attributes can be quantified. This is the result of the input incomplete timber sale units utilizing a common silvicultural prescription name as state PTU's. Results of running this tool are described in the results/discussion section of this report. Once the data prep phase is complete the newly created feature classes above can be utilized as input parameters for the six vegetation analysis geoprocessing tools.

Forest habitat group type tool

For environmental impact statements, foresters and ID team members are required to complete analysis on potential effects forest management projects may have to the existing environment. In this project, the effect of forest management on six aspects of forested stands were completed using Python scripting. To complete this analysis more efficiently, the Python scripts were implemented by custom geoprocessing tools where current forest stand condition and proposed post-harvest condition were summarized for forest habitat type groups, desired future conditions, current cover types, age class, old-growth status, and age/cover type patch sizes. The Python script developed for this tool can be found in *Appendix D: Geoprocessing Tool Python Script 2: Habitat Type Group Analysis (Current Condition).* The resulting geoprocessing tools require user input feature classes from the vegetation analysis data prep tool. The inputs are then processed, and summary tables exported as Microsoft Excel spreadsheets into a designated folder. A supplemental support document was also created as shown in *Appendix C.* This appendix describes how inputs were manipulated in the analysis, why the analysis was being completed, and the functional product of each geoprocessing tool for the project. This workbook is crucial for ID team members to understand how the scripting manipulates their input data and will aid in eventual troubleshooting and adaptation of the scripts for future projects.

The first tool that was created was the forest habitat type group tool. Forest habitat type groups can be described as the potential vegetation community, patterns of succession, and potential for productivity at a given site (Pfister et al., 1977). The forest habitat types were split in the confines of the cumulative effects area, and direct effects area into nine separate habitat type groups. In the context of this project, management of forest stands would not change the habitat type group because the measurement of this forest attribute only gives foresters an idea of what vegetation and productivity might be. The created tool simply summarized the total acreage for each group type for the cumulative and direct effects area. Basic workflow and process for the Python script is described by conceptual model in figure 3.

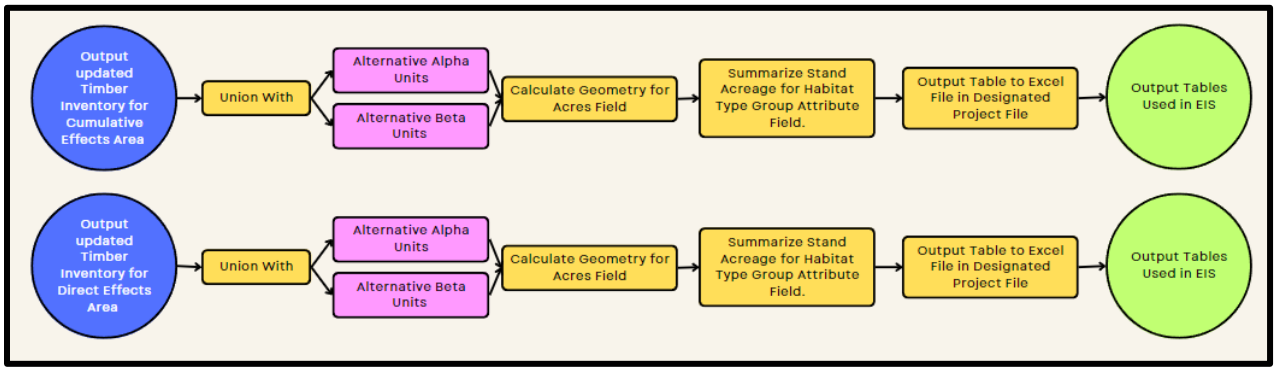


Figure 3 workflow shows a basic outline for the scripting process. Blue circles represent key feature class inputs and green circles represent outputs that will be used in the environmental impact statement. The individual workflows are one process that can take either input.

The forest habitat type group tool consists of four user input parameters including the direct effects updated timber inventory, the cumulative effects updated timber inventory, and one output excel file locations as described below[9].

- **Cumulative effects updated timber inventory:** data type: feature layer. (ResultsRun_UpdatedCETimberInventory_20230221).
- **Direct effects updated timber inventory:** data type: feature layer. (ResultsRun_UpdatedDETimberinventory_20230221).

[9] Descriptions of the input parameters are included in the metadata for the geoprocessing tools where users can "hover" over the input information to see how to input appropriate data.

- **Output location and file name:** data type: file. This parameter allows the user to select a save folder in their home directory and assign a unique name for the summary table. The parameter also has a file filter on it to only allow the creation of Excel (.xlsx) file type. File set at a desired location as an output dataset.

Once the datasets were defined, areas for both the direct effects updated timber inventory area, and cumulative effects updated timber inventory area were updated utilizing the calculate geometry attributes function from the arcpy site package. This process needs to be completed at the beginning of each of the tools to ensure that the acreage being summarized is accurate and that inadvertent changes to stand polygon boundaries are quickly identified. To determine the accuracy of the acres within the updated timber inventory areas, control acreage from the original downloaded datasets can be compared to the output summary tables from this tool. An example of this would be that total acres of both project areas need to match that of the original acres.

After the acres are calculated, the updated direct effects and cumulative effects area were passed to the statistics function from the arcpy site package. This function summed the acres based on the habitat type group field (HAB_GRP) and the non-spatial tables were saved to the GIS's default geodatabase set in the environments tab. The non-spatial tables were then sent to an Excel file through use of the arcpy conversion function, table to Excel, where the output location and file name parameters set the location of where the newly created Excel files were written to the disk. To keep the home environment geodatabase workspace clean, the non-spatial tables were passed to the delete function from arcpy. Once all scripting was completed, the metadata for the habitat type group tool was updated to include descriptions of the tool's function, parameter definitions, and use limitations of the tool. Results from running this tool are described in the discussion section of this report.

Desired future condition tool

The next step in the environmental impact statement vegetation analysis is to complete an existing condition analysis for the desired future conditions of forested stands within the cumulative effects, and direct effects analysis areas. The desired future condition of a forested stand represents the historic dominant tree cover based on observations from the 1930's (Losensky, 1997). The desired future condition does not change based on silvicultural treatment selected by foresters and ID team members, but some prescriptions will alter current stand cover which will move the stand towards its desired future condition faster than others. Because the desired future condition will never change for

the stand, a Python script, like the habitat type group analysis was created. The change in the current stand cover as it relates to desired future conditions is implemented in the following section.

To complete current desired future condition analysis, the same procedure was followed to create summary Excel tables based on the acreage of the desired future condition forest types as shown in figure 4. The script for this process is shown in *Appendix D: Geoprocessing Tool Python Script 3.* There are eight different forest types that are classified across the forest that denote the desired predominant forest cover in desired tree species. To begin, user input parameters were set to collect the following features, as well as set the output file locations for the resulting Excel summary tables as described below.

- **Cumulative Effects Updated Timber Inventory:** data type: feature layer. (ResultsRun_UpdatedCETimberInventory_20230221).
- **Direct Effects Updated Timber Inventory:** data type: feature layer. (ResultsRun_UpdatedDETimberinventory_20230221).
- **Direct Effects Output Location and File Name:** data type: file. This parameter allows the user to select a save folder in their home directory and assign a unique name for the summary table. The parameter also has a file filter on it to only allow the creation of Excel (.xlsx) file type. File set at a desired location as an output dataset.

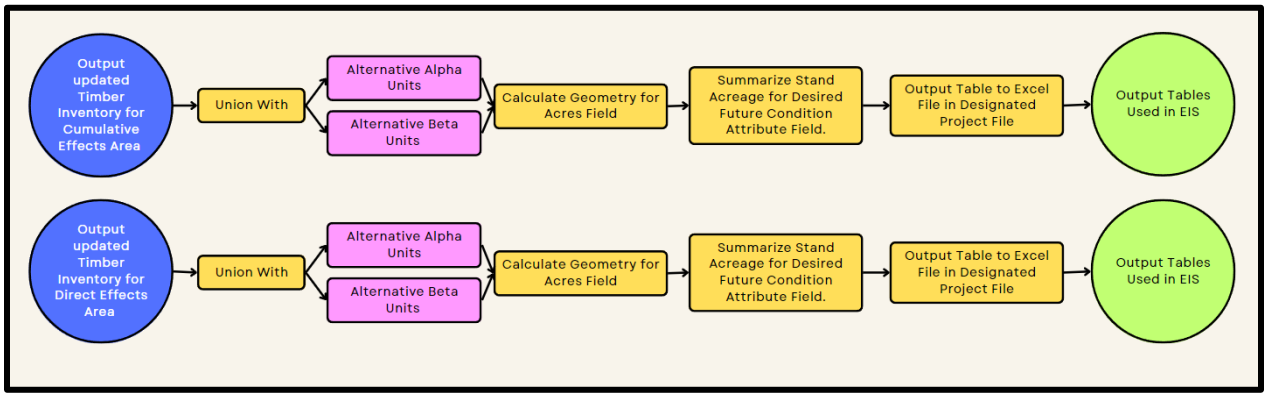
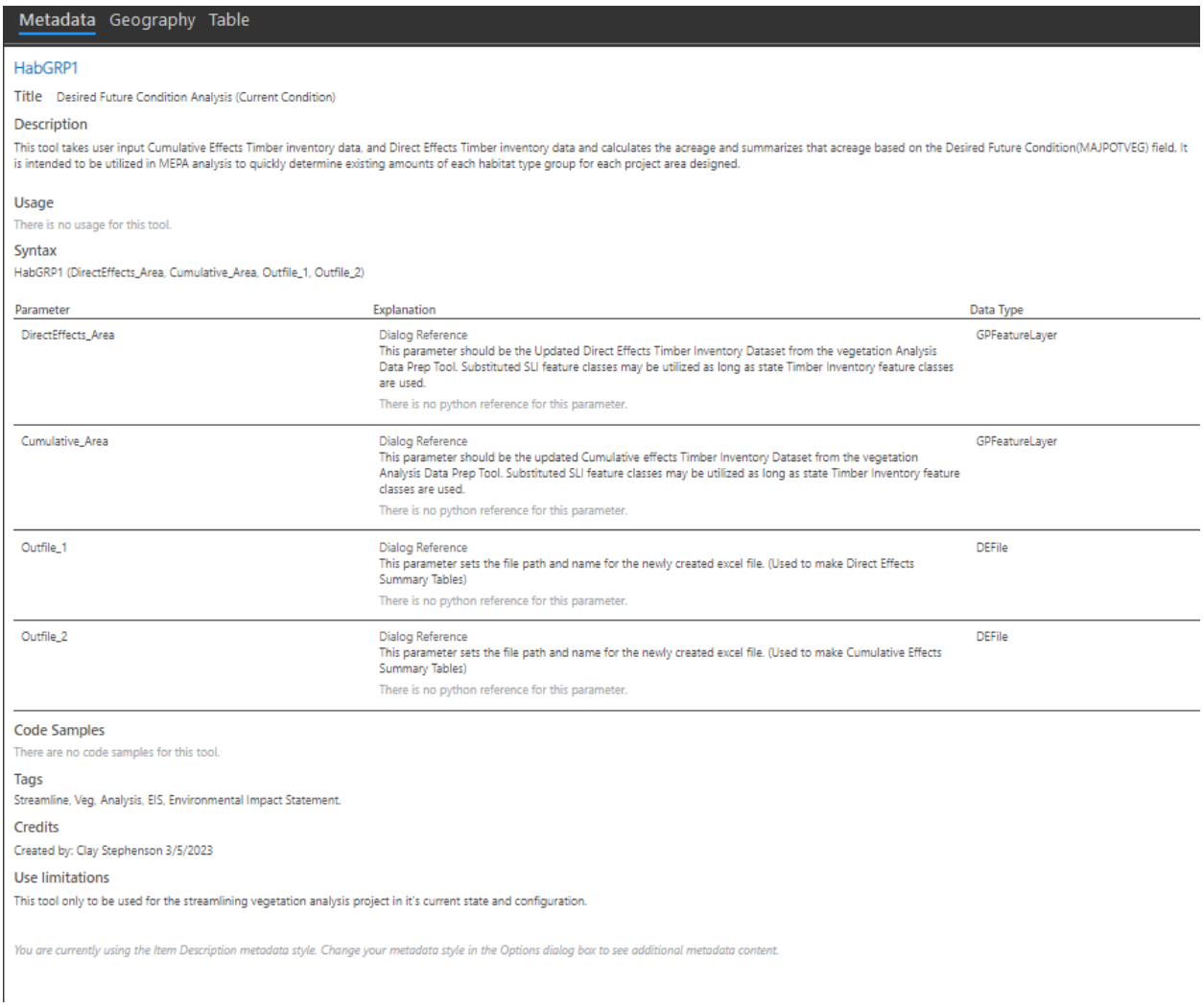


Figure 4 workflow shows a basic outline for the scripting process. Blue circles represent key feature class inputs and green circles represent outputs that will be used in the environmental impact statement. The individual workflows are one process that can take either input.

After user inputs were designated, geometry was then calculated for the acres field of the input datasets, ensuring that any intentional alterations to the source dataset are captured. Once the acreage had been calculated, the resulting feature classes were passed to a non-spatial summary statistics table where the acreage was summed based on the desired future condition (MAJPOTVEG) field. Those tables

are then passed to the table to Excel function from the arcpy site package, and the user input file path locations were utilized to set where the final tables were written to the local computer's disk. To keep the working environment clean, the non-spatial summary tables were then deleted from the home geodatabase, leaving only the desired Excel files on the computer.

Because multiple tools are being created at this point in the project, the metadata was not only updated for the resulting feature classes but was also filled out for the scripting tools themselves. Metadata was updated for each tool in this project, but an example from the desired future condition analysis tool is shown in figure 5 below.



Metadata Geography Table

HabGRP1

Title Desired Future Condition Analysis (Current Condition)

Description

This tool takes user input Cumulative Effects Timber inventory data, and Direct Effects Timber inventory data and calculates the acreage and summarizes that acreage based on the Desired Future Condition(MAJPOTVEG) field. It is intended to be utilized in MEPA analysis to quickly determine existing amounts of each habitat type group for each project area designed.

Usage

There is no usage for this tool.

Syntax

HabGRP1 (DirectEffects_Area, Cumulative_Area, Outfile_1, Outfile_2)

Parameter	Explanation	Data Type
DirectEffects_Area	Dialog Reference This parameter should be the Updated Direct Effects Timber Inventory Dataset from the vegetation Analysis Data Prep Tool. Substituted SLI feature classes may be utilized as long as state Timber Inventory feature classes are used. There is no python reference for this parameter.	GPFeatureLayer
Cumulative_Area	Dialog Reference This parameter should be the updated Cumulative effects Timber Inventory Dataset from the vegetation Analysis Data Prep Tool. Substituted SLI feature classes may be utilized as long as state Timber Inventory feature classes are used. There is no python reference for this parameter.	GPFeatureLayer
Outfile_1	Dialog Reference This parameter sets the file path and name for the newly created excel file. (Used to make Direct Effects Summary Tables) There is no python reference for this parameter.	DEFile
Outfile_2	Dialog Reference This parameter sets the file path and name for the newly created excel file. (Used to make Cumulative Effects Summary Tables) There is no python reference for this parameter.	DEFile

Code Samples

There are no code samples for this tool.

Tags

Streamline, Veg, Analysis, EIS, Environmental Impact Statement.

Credits

Created by: Clay Stephenson 3/5/2023

Use limitations

This tool only to be used for the streamlining vegetation analysis project in it's current state and configuration.

You are currently using the Item Description metadata style. Change your metadata style in the Options dialog box to see additional metadata content.

Figure 5 shows a sample metadata for custom geoprocessing tools. It details the intended usage, as well as documentation for user input parameters. As geoprocessing tools are created, metadata must be edited to reflect the intended use of the tool, document the input datasets and set limitations for the use of the tool.

Current cover type tool

Once the geoprocessing tool for identifying and exporting the desired future condition of forested stands was complete, the current cover analysis was completed. The created script is shown in *Appendix D: Geoprocessing Tool Python Script 4*. The current cover of forested stands is described as the current stand condition in terms of species composition, and these cover types were broken up into 9 distinct classes that detail the predominant cover type in each of the forest stands. These classes consist of Douglas-fir, hardwood, lodgepole pine, mixed conifer, non-stocked, ponderosa pine, subalpine fir, western larch/Douglas-fir, and western white pine cover types. Current cover is expected to change between current condition and post-harvest conditions in even aged management treatment because these treatments remove most of the large standing trees and canopy cover.

To capture this change in stand condition this script added two new user inputs over the previous geoprocessing tools. The overall workflow of this script is shown in figure 6.

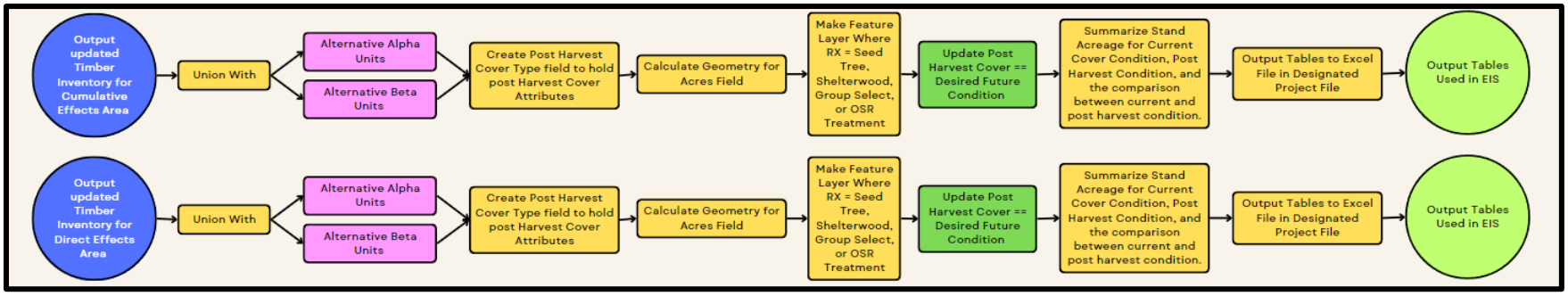


Figure 6 workflow shows a basic outline for the scripting process. Blue circles represent key feature class inputs and green circles represent outputs that will be used in the environmental impact statement. The individual workflows are one process that can take either input. **Update fields for other vegetation analysis portions will differ based on the proposed harvest treatment. I.e., age class will change based on less intense and more intense treatment.**

These new parameters, alternative proposed forest management treatments, serve as contrasting alternatives that ID team members will usually design during the project planning phase of the environmental impact statement. These alternatives generally must be viable options for forest management treatment, and foresters will use alternative development to home in on certain forest issues that could have long term impacts on forest stand structure and forest health. Generally, the ID team expects to see ample change between post treatment effects to forest stand condition. Because of the nature of an EIS's planning process, real world alternative treatment feature classes could not be used so hypothetical units were created that mimic the attributes of the server hosted feature layer PTU that is currently available to the foresters during the project implementation process. The user input parameters consist of two feature layers called alternative Alpha treatment units (PTU_AltA_20230221), and alternative Beta treatment units (PTU_AltB_20230221) that have key attributes of silvicultural prescription (RX), area, and cutting unit number. In a real analysis setting, there are more attribute fields available, but these attributes were unneeded for the scope of this project. The list of user inputs for this geoprocessing tools were:

- **Cumulative effects updated timber inventory:** data type: feature layer. (ResultsRun_UpdatedCETimberInventory_20230221).
- **Direct effects updated timber inventory:** data type: feature layer. (ResultsRun_UpdatedDETimberinventory_20230221).
- **Summary table output location and file name:** data type: file. This parameter allows the user to select a save folder in their home directory and assign a unique name for the summary table. The parameter also has a file filter on it to only allow the creation of excel (.xlsx) file type. File set at a desired location as an output dataset.
- **Alternative alpha treatment units:** data type: feature layer. This parameter is a user defined feature layer detailing polygons showing intended areas of forest management. Key attribute fields are silvicultural prescription (Rx) and cutting unit number (Cutting_Unit). (PTU_AltA_20230221).
- **Alternative beta treatment units:** data type: feature layer. This parameter is a user defined feature layer detailing polygons showing intended areas of forest management. Key attribute fields are silvicultural prescription (Rx) and cutting unit number (Cutting_Unit). (PTU_AltB_20230221).

A pivotal concept that needed to be coded into the Python scripts and output to the ID team/end users is the description of change in acreage between the current forest stand condition, and the intended post-harvest condition based on alternative selection. In short, the program needed to tell the end user what acres of cover type they currently had in their respective project areas, and what those acres would change to once the forest treatment was completed. To complete this task, the script first uses the union function from the arcpy site package to incorporate the user input cumulative effects updated timber inventory and direct effects updated timber inventory with both the Alpha and Beta treatment units. The union function will take input polygons and "overlay" them onto one another and all features and their attributes are written to a new feature class as shown in figure 7. This process gave the input stand level inventory the attributes of the alternative's silvicultural prescription with which the tool makes field calculations to quantify the change in acreage for the desired forest attribute[10].

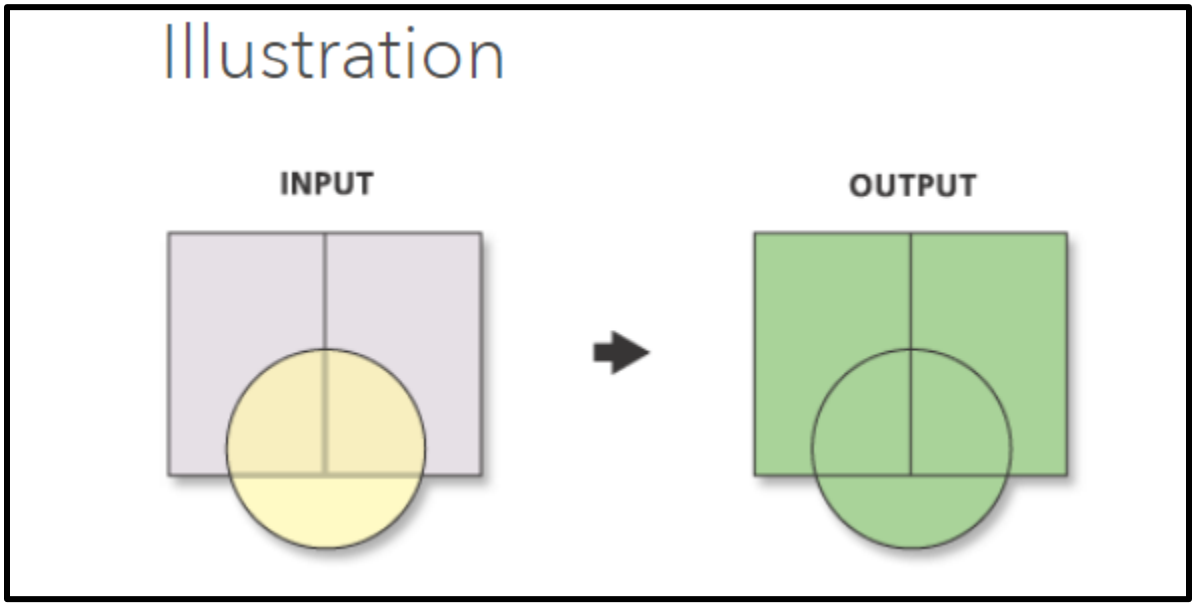


Figure 7 shows how the union works within the script. Input proposed alternative units and the updated timber inventory layer are created and all boundaries and edges are maintained. In addition to boundaries being maintained, the attributes from the attribute table are maintained. This will allow further manipulation of attribute fields that are created later in the Python script. This image was produced by ESRI and accessed by C. Stephenson on 3/26/2023. The illustration can be found on the ArcGIS Pro summary website found at (https://pro.arcgis.com/en/pro-app/latest/tool-reference/analysis/union.htm)

Once these union datasets were created, they were passed to arcpy's calculate field function where a post treatment cover field (CVR_CURR_POST) was created and populated with the same values that

[10] This and all processes described in this report had to be completed twice, once for each alternative. This action is reflected in the Python scripts shown in *Appendix D*

the current cover field (CVR_CURR) holds. To ensure that acreages are correct after geometry was changed due to the union process, area in acres was then calculated for the acres (ACRES) field. Once acres were updated, union features were passed to the arcpy make feature layer function. This allowed where clauses to be implemented so that even age management treatments could be selected. After the desired Rx treatment was selected[11] the post treatment cover field (CVR_CURR_POST) was updated to match the preferred future vegetation field (MAJPOTVEG) if the Rx treatment fell into an even aged management regime. This shows that the current forest stand cover will change to the preferred future vegetation cover with an even aged management treatment.

Finally, summary tables were built and exported to the user defined file output for current cover stand condition, and post treatment cover stand condition, as well as summarizing the change between current and post treatment condition by incorporating the summary statistics for acres by both fields and reporting them to the same Excel worksheet. All summary statistics tables were exported to the same Excel file and the individual sheets were clearly labeled by the script and could clearly be accessed within the Excel environment. To keep the GIS working environment clean, all intermediate and final feature classes and non-spatial tables were deleted at the end of the script. Results from this geoprocessing tool are displayed and presented in the discussion portion of this report.

Forest age class tool

Like forest current and post-harvest cover type, forest age class is another important metric that foresters and ID team members pay attention to because forest age can imminently be affected by common treatments used to achieve overall project objectives. The age class of forested stands is described as the current average age of all trees within the stand, and these ages were broken up into 6 classes that represent the average age of the dominant vegetation cover across the stand. These classes consist of 000-039, 040-099, 100-149, 150-199, 200+, and old growth age. Age class is expected to change between current condition and post-harvest conditions in more intense management treatment because these treatments remove most of the large standing trees/cover which significantly reduces the average age of the stand.

[11] The resulting feature layer from this process make a temporary feature layer that only contain attributes of the Rx field designated by the script. Essentially further edits to this feature layer will only affect the desired polygons that have that Rx treatment.

The geoprocessing tool and Python script that was created (*Appendix D: Geoprocessing Tool Python Script 5*) for this part of the project followed the same basic principles as described above in the current cover type workflow. A workflow for the script was developed before coding as shown in figure 8.

The user defined parameters for the script include the following:

- **Cumulative Effects Updated Timber Inventory:** data type: feature layer. (ResultsRun_UpdatedCETimberInventory_20230221).
- **Direct Effects Updated Timber Inventory:** data type: feature layer. (ResultsRun_UpdatedDETimberinventory_20230221).
- **Summary Table Output Location and File Name:** data type: file. This parameter allows the user to select a save folder in their home directory and assign a unique name for the summary table. The parameter also has a file filter on it to only allow the creation of Excel (.xlsx) file type. File set at a desired location as an output dataset.
- **Alternative Alpha Treatment Units:** data type: feature layer. This parameter is a user defined feature layer detailing polygons showing intended areas of forest management. Key attribute fields are silvicultural prescription (Rx), and cutting unit number (Cutting_Unit). (PTU_AltA_20230221).
- **Alternative Beta Treatment Units:** data type: feature layer. This parameter is a user defined feature layer detailing polygons showing intended areas of forest management. Key attribute fields are silvicultural prescription (Rx) and cutting unit number (Cutting_Unit). (PTU_AltB_20230221).

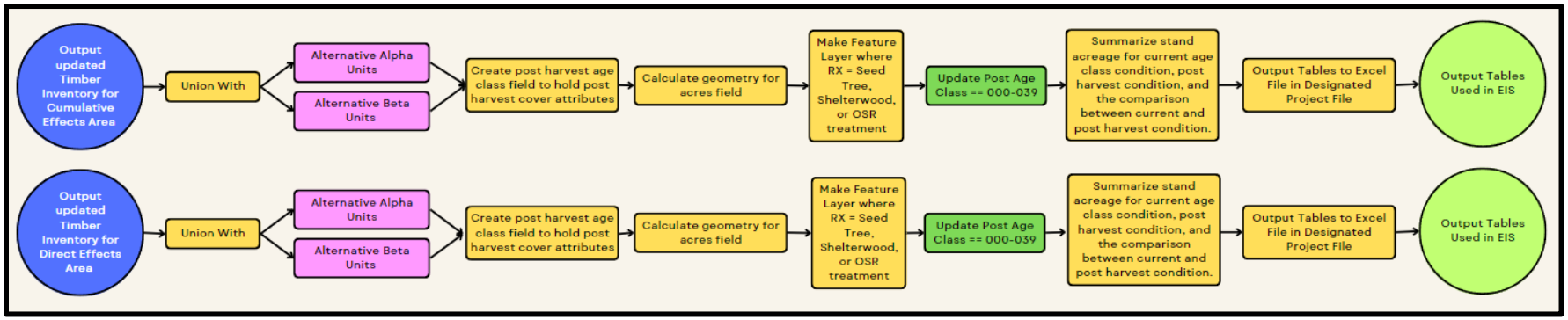


Figure 8 workflow shows a basic outline for the scripting process. Blue circles represent key feature class inputs and green circles represent outputs that will be used in the environmental impact statement. **Update fields for other vegetation analysis portions will differ based on the proposed harvest treatment. I.e., age class will change based on less intense and more intense treatment.**

The script took these variables and completed the union function between the updated timber inventory feature classes to the alternative Alpha and Beta treatment units. Once the union function was completed the features were passed to the calculate field function where a post treatment age class (AGECLASS_POST) field was created and populated with the values from the current age class field (AGECLASS). A series of feature layers were then created and where clauses implemented to show only stands that had the silvicultural prescription consistent with even aged management. These temporary feature layers' post treatment age class fields were then updated to show 000-039 years of average age. Because these changes are made to the selection only, but the process utilizes the original union feature as an input, the changes made to the selected feature layer are incorporated into the original union layer.

Once changes to the post treatment age class (AGECLASS_POST) field were made, these features were passed to the summary statistics function where acres were calculated. The resulting non spatial tables were then exported to Excel where the summary of acres based on current age class, post treatment age class, and the change in age class types were sent to workbooks in the user defined Excel location. Once the Excel tables were exported all temporary and final feature classes, layers, and non-spatial summary statistic tables were deleted. Results and outputs from this geoprocessing tool are discussed in the discussion section of this report.

Forest old growth tool

Like age class, old growth must be considered by ID team members and foresters. The amount and distribution of old growth on the forested landscape is often one of the largest defining factors that foresters use to distinguish between alternatives. The state of Montana utilizes the Green et al (1992) definition of old growth where number of large diameter trees per acre, basal area, and average age of the stand are the defining factors in determining old growth. These attributes are described in many places in the state's timber inventory (Timberinventory_20230103), but the age class (AGECLASS) field combines field verified stands into the old growth designation and this process is completed just before analysis begins for the project. For an environmental impact statement this represents the most concise representation of old growth in the project area and is what has traditionally been utilized for vegetation analysis. It should also be noted that further description of forest stands old growth attributes and indexes are considered for environmental impact statements but are outside the scope of the current geoprocessing tools intended for the vegetation analysis. The old growth analysis geoprocessing tool follows a similar workflow as described in previous sections and only accounts for

acres of current old growth and potential acres that will not be old growth post-harvest. Future versions of the vegetation analysis geoprocessing tool could utilize and describe the old growth forest attributes, should ID team members need them.

To complete the old growth analysis, a Python script was written (*Appendix D: Geoprocessing Tool Python Script 6*) and created the following user defined parameters[12].

- **Cumulative Effects Updated Timber Inventory:** data type: feature layer. (ResultsRun_UpdatedCETimberInventory_20230221).
- **Direct Effects Updated Timber Inventory:** data type: feature layer. (ResultsRun_UpdatedDETimberinventory_20230221).
- **Summary Table Output Location and File Name:** data type: file. This parameter allows the user to select a save folder in their home directory and assign a unique name for the summary table. The parameter also has a file filter on it to only allow the creation of Excel (.xlsx) file type. File set at a desired location as an output dataset.
- **Alternative Alpha Treatment Units:** data type: feature layer. This parameter is a user defined feature layer detailing polygons showing intended areas of forest management. Key attribute fields are silvicultural prescription (Rx) and cutting unit number (Cutting_Unit). (PTU_AltA_20230221).
- **Alternative Beta Treatment Units:** data type: feature layer. This parameter is a user defined feature layer detailing polygons showing intended areas of forest management. Key attribute fields are silvicultural prescription (Rx), and cutting unit number (Cutting_Unit). (PTU_AltB_20230221).

The script follows the workflow described in figure 9.

[12] User defined parameters for all geoprocessing tools after data prep are the same. This portion of the report is intended to thoroughly detail the inputs and describe the process of the scripts. Results will be covered in the discussion section.

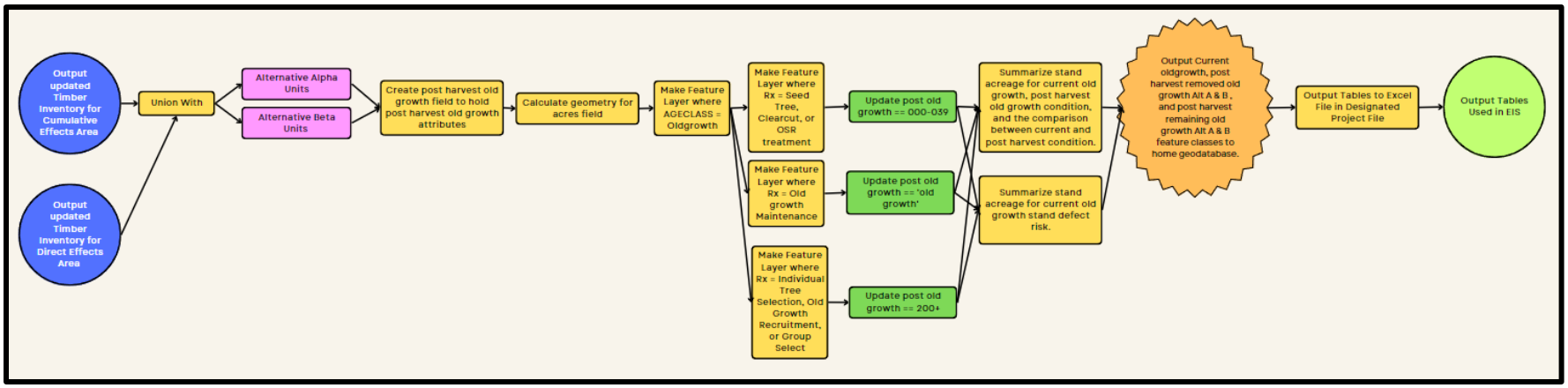


Figure 9 workflow shows a basic outline for the scripting process. Blue circles represent key feature class inputs and green circles represent outputs that will be used in the environmental impact statement. The orange star figure represents output feature classes that will be added to the home geodatabase. Process is the result of *not* deleting the features in the script. Update fields for other vegetation analysis portions will differ based on the proposed harvest treatment. I.e., age class will change based on less intense and more intense treatment.

These parameters were then utilized by the script to union the updated timber inventory feature classes and union them to the alternative Alpha and Beta treatments. A post treatment old growth field (OLDGROWTH_POST) was then created in the union dataset and calculated to contain the same values as the current age class field (AGECLASS). The make feature layer function from the arcpy site package was then utilized to select areas that were old growth only. This allows the rest of the script to make changes to only the affected old growth acres and disregards all other age classes. The old growth only feature layer was then passed to another make feature layer function where silvicultural prescriptions for even aged management were selected and the old growth post treatment field updated to show the corresponding age class. This process follows the same logic as the age class geoprocessing tool where even aged management changes to the 000-039 age class.

In addition to removal of old growth into the lowest age class this also takes into consideration how uneven aged management, and low impact treatments could affect the age class of a current old growth stand. To do this, a feature layer was created by inputting the old growth only feature layer into the make feature layer function. A where clause only identified treatments in old growth recruitment (OGR), group select (GS), and individual tree select (ITS) and changed the post treatment old growth (OLDGROWTH_POST) to be in the 200+ year age class. This portion of the script says that the treatment in these old growth stands was not enough to change the overarching age of the stand, but that generally, the stand would no longer meet the rest of the requirements described in the Green et al (1992) definition. Summary tables were then created and exported to Excel showing the current amount of acreage for each age class[13] and alternative Alpha and alternative Beta post treatment old growth amounts were exported. Like before, all temporary and final feature layers and non-spatial summary statistics tables were deleted from the workspace environment.

A key concept that is discussed at the project implementation and planning level is the level of insect and disease pressure within old growth stands. ID team members will often report the number of high-risk, medium-risk, and low-risk old growth stands that are being treated and these numbers often will inform the selection of a preferred alternative. For example, an alternative treating more acres of high-risk old growth than an old growth maintenance treatment might be selected over one that does not treat the high-risk acres. For this project, a hypothetical insect and disease risk rating field

[13] The first summary table would be identical to the first table from the age class analysis but was exported for ease of interpretation for ID team members when taking the tables from Excel into a final environmental document. This also provides an opportunity to quality control numbers across non-spatial tables.

(Defect_Risk) was applied to both the alternative Alpha and alternative Beta proposed treatment units. This field mimics the insect and disease risk rating that is generally collected in the field during the reconnaissance phase of project development and is expected to vary within the project area from forested stand to forested stand. To quantify the number and current risk of acres of old growth that will potentially be treated, the old growth only feature layer was passed to the summary statistics function and the acres were summed and output to an Excel table.

Because foresters often include maps of old growth in the vegetation analysis section of environmental impact statements, the copy features function from arcpy was used at the end of this script to save the following temporary feature layers to the disk/the home geodatabase that the current ArcGIS Pro project file is using[14]. The following temporary feature layers were saved to home geodatabase. They are intended to serve as a visual representation of where old growth exists and is being removed within the cumulative effects and direct effects project area.

- **Current_OG_SRSF:** data type: feature class. This feature class shows the user all current old growth stands within the cumulative effects project area. *End users might alter this output to fit the thematic needs of the maps they are creating.
- **Removed_OG_AltA:** data type: feature class. This feature class shows all areas of old growth that will be removed with alternative Alpha treatments.
- **Removed_OG_AltB:** data type: feature class. This feature class shows all areas of old growth that will be removed with alternative Beta treatments.
- **Remaining_OG_AltA:** data type: feature class. This feature class shows all areas of old growth remaining after implementing alternative Alpha treatments.
- **Remaining_OG_AltB:** data type: feature class. This feature class shows all areas of old growth remaining after implementation of alternative Beta treatments.

Age and cover type patch size tool

The final tool created to support and streamline vegetation analysis for foresters and ID team members was the age and cover type patch size tool. The Python script for this tool is shown in *Appendix D: Geoprocessing Tool Python Script 8*. Age class distribution and cover type distribution across the landscape informs ID team members of where the acreage occurs. Larger patches of contiguous older forest age classes and areas that meet the desired future and historical conditions of the forest are

[14] The home environment is set outside of the geoprocessing tools. This process is described in *Appendix C.*

more desirable. These forest attributes are taken into consideration during the planning phase of the environmental impact statement process. The script developed for this project follows the outline described in the conceptual model in figure 10. Though there is only one model shown for the process, two distinct geoprocessing tools were created to give flexibility to ID team members as they run the analysis. This was also completed to make coding easier where each individual script completes a singular analysis process.

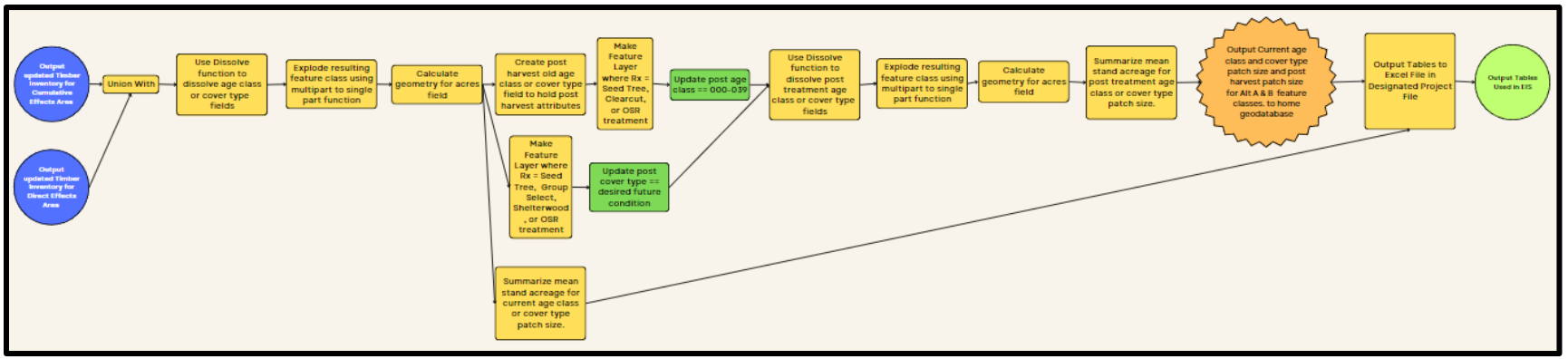


Figure 10 workflow shows a basic outline for the scripting process. Blue circles represent key feature class inputs and green circles represent outputs that will be used in the environmental impact statement. The orange star figure represents output feature classes that will be added to the home geodatabase. Process is the result of *not* deleting the features in the script. **Update fields for other vegetation analysis portions will differ based on the proposed harvest treatment. I.e., age class will change based on less intense and more intense treatment.**

Once again, user defined parameters were first created in the script. These parameters are the same as the geoprocessing tools created before. To find the current patch size however, a new process was completed at the beginning of this script. To begin, a new feature class was created that showed where the current adjacent age class boundaries were dissolved. The dissolve functions process is illustrated below in figure 11. After the dissolve function was completed for the age class field, the resulting features were multiple multi-part polygons based on the age class. This would be inadequate to calculate the mean acres for these polygons, so the features were passed to the multipart to single part function from the arcpy site package. This tool "explodes" the features so that unconnected islands of individual age classes would be represented as their own, single part polygon. The multipart to single part function's process is described in figure 12.

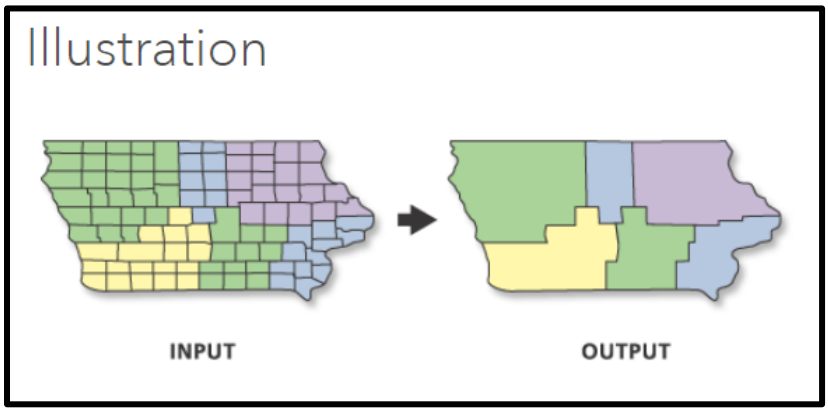


Figure 11 shows the basic use of the dissolve tool from the arcpy site package. Like attributes from a designated field are dissolved to show total acreage of the desired class. This image was produced by ESRI and accessed by C. Stephenson on 3/26/2023. The illustration can be found on the ArcGIS Pro summary website found at (https://pro.arcgis.com/en/pro-app/latest/tool-reference/data-management/dissolve.htm)

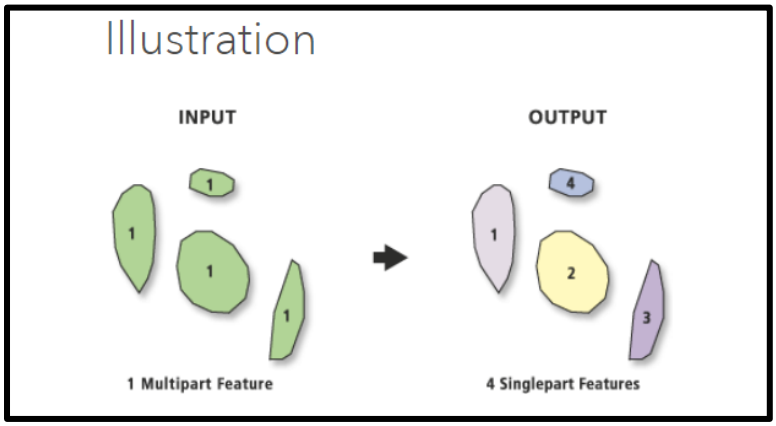


Figure 12 shows the basic use of the multipart to single part tool from the arcpy site package. Like attributed polygons that are not connected or are " Islands" will be separated. This allows a true mean patch size to be calculated later in the script. This image was produced by ESRI and

accessed by C. Stephenson on 3/26/2023. The illustration can be found on the ArcGIS Pro summary website found at (https://pro.arcgis.com/en/pro-app/latest/tool-reference/data-management/multipart-to-singlepart.htm)

Once the current age classes were dissolved and "exploded", a summary table was created that summarized using the 'mean' parameter on the acres (ACRES) field based on the current age class (AGECLASS) field. The resulting non spatial table was then passed to the table to Excel function used at the end of the script.

After the first summary table was created, a similar process to the previous geoprocessing scripts was implemented to change the current age class conditions to the intended post treatment age class conditions. This was done by creating a new post age class field (AGECLASS_POST) and updating the field by selecting the desired silvicultural prescriptions using the make feature layer function. This process resulted in identical results to the age class analysis script, but further manipulation of the features was required to result in the correct patch size analysis. To do this, the age class post (AGECLASS_POST) field was dissolved and exploded using the dissolve function and the multipart to single part function from the arcpy site package. To ensure that the acres were accurate for the final summary statistics table, the exploded feature class was then passed to the calculate geometry attribute function. This calculated the geometry for the newly minted age class patch size polygons. Finally, the post treatment age class (AGECLASS_POST) field from the post age exploded patch feature class was summarized using the 'mean' parameter and the resulting non spatial summary table was passed to the table to Excel function. All temporary and final feature classes and non-spatial tables were then deleted at the end of the script to ensure that the home workspace environment remained uncluttered.

Because foresters generally will need to visually represent the current and updated patch sizes for the environmental impact statement, a few key resulting patch size feature classes were omitted from the delete function. These feature classes can be incorporated into maps for the environmental impact statement and are as follows:

- **Current_Ageclass_Patchsize_CE:** datatype: feature class. This feature class shows the average patch size of all age classes across the cumulative effects project area. * End users might alter this output to fit the thematic needs of the maps they are creating.
- **AltA_Post_age_Patchsize:** datatype: feature class. This feature class shows the average patch size of all age classes after harvest for alternative A.
- **AltB_Post_age_Patchsize:** datatype: feature class. This feature class shows the average patch size of all age classes after harvest for alternative B.

- **Current_Cover_Patchsize_CE:** datatype: feature class. This feature class shows the average patch size of all cover types across the cumulative effects project area. * End users might alter this output to fit the thematic needs of the maps they are creating.
- **AltA_Post_Cover_Patchsize:** datatype: feature class. This feature class shows the average patch size of all cover types after harvest for alternative A.
- **AltB_Post_Cover_Patchsize:** datatype: feature class. This feature class shows the average patch size of all cover types after harvest for alternative B.

After the age class patch size was completed, the same process was carried out for forest cover types. The Python scripting looks exactly the same as the age class but the variables and field names were changed from the current and post age class fields (AGECLASS, AGELCASS_POST) to the current and post cover type fields (CVR_CURR, CVR_CURR_POST). The resulting tables and feature classes are labeled accordingly. Details on the resulting feature classes described above can be found in *Appendix A.*

Methods summary

Two project areas were defined called the cumulative effects and direct effects project areas. Once that was completed hypothetical incomplete harvest units, alternative Alpha units, alternative Beta units, were then created to serve as inputs for the streamlining vegetation analysis geoprocessing tools. A data preparation tool was then developed to take the state's timber inventory layer, the project area boundaries, and the incomplete harvest units to incorporate potential harvest effects to the current timber inventory using union, make feature layer, calculate field, and summarize arcpy functions. Forest habitat type and desired future condition summary table tools were created by calculating total acreage for the desired forest attributes and exporting them to Excel using the table to Excel functions from the arcpy site package.

To quantify post-harvest treatments, a current and post treatment cover type, forest age class, old growth, and age and cover patch size tools were created. These tools followed similar processes to the data preparation tool where the union, make feature layer, calculate field, and summarize functions were used. Outputs for these tools include table to Excel generated Excel summary tables, in addition to select feature classes that were output to the workspace environment. This end use of these outputs is intended to both describe the changes on the landscape as well as show the changes to old growth spatially via the resulting feature classes.

To finish up the project, a final two geoprocessing tools were scripted to show the effects of alternative forest management on the patch size for age classes and cover types across the cumulative effects and direct effects project areas. This was completed by inputting the updated timber inventory layers from the data prep method and the alternative treatment units. These features were then manipulated using the union, make feature layer, calculate field, dissolve, multipart to single part, and summarize functions. These tools produced a series of non-spatial summary tables describing changes in the forest age class and cover across the project areas, as well as a visual representation of those changes in the form of updated feature classes.

The methods for this project are intended to be dynamic and should be manipulated and changed as projects and state processes change. Though the processes for how the scripting works for streamlining vegetation analysis, the results of the geoprocessing tools will be described and discussed in the following section.

Tool outputs and discussion

Upon completion of the methods section of the project, a complete "run" of the geoprocessing tools was completed to ensure proper function. This process also served to investigate how the end user would experience the vegetation analysis toolbox and further tweaks and alterations were made to the Python scripts and functionality to improve the user experience. The following discussion presents the findings and outputs of the specific geoprocessing tools that were created and describes the use of the tools as it relates to an environmental impact statement. The analysis that was completed by running through the same process an ID team member might complete during vegetation analysis and findings are presented. Because state environmental analysis processes are completed over long time scales and must go through extensive internal review before release for public review, live data was not utilized in the geoprocessing tools. This suite of tools may be utilized to complete vegetation analysis at any area in the state, once it is vetted through an internal review process at the Forest Management Bureau within the Department of Natural Resources and Conservation. This portion of the project also discusses how the outputs for this tool can be utilized for further spatial analysis that can be completed utilizing other Python scripting site packages and libraries.

Data preparation results

The first step in completing the streamlined version of the vegetation analysis, using the geoprocessing tools created in the methods section was to run the vegetation analysis data prep tool. The inputs and starting environment in the ArcGIS Pro GIS are shown in figure 13.

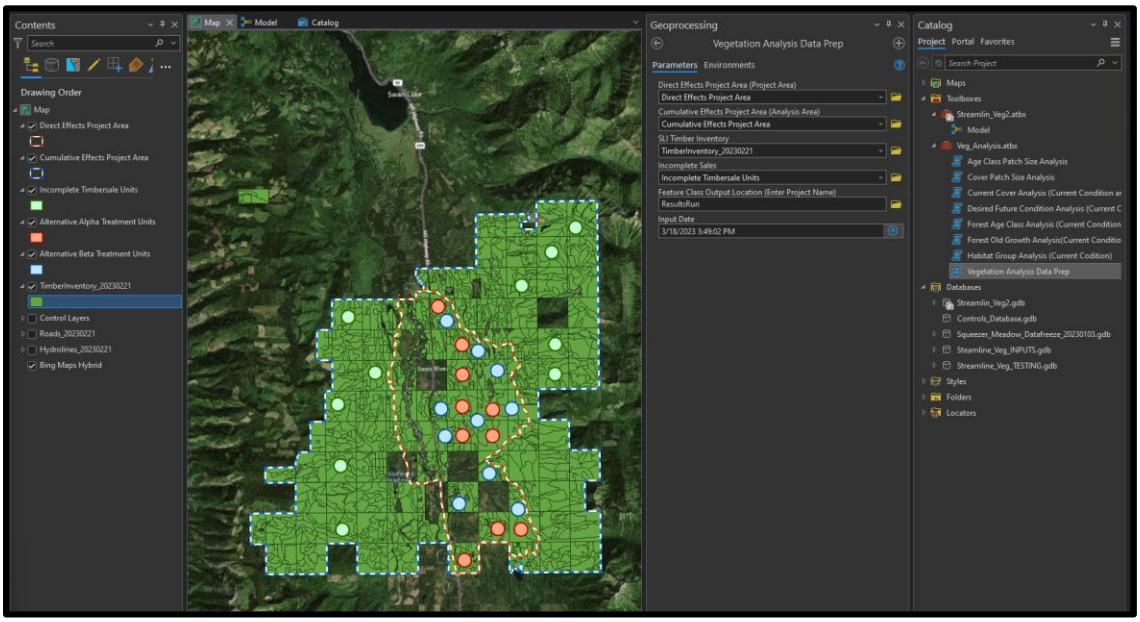
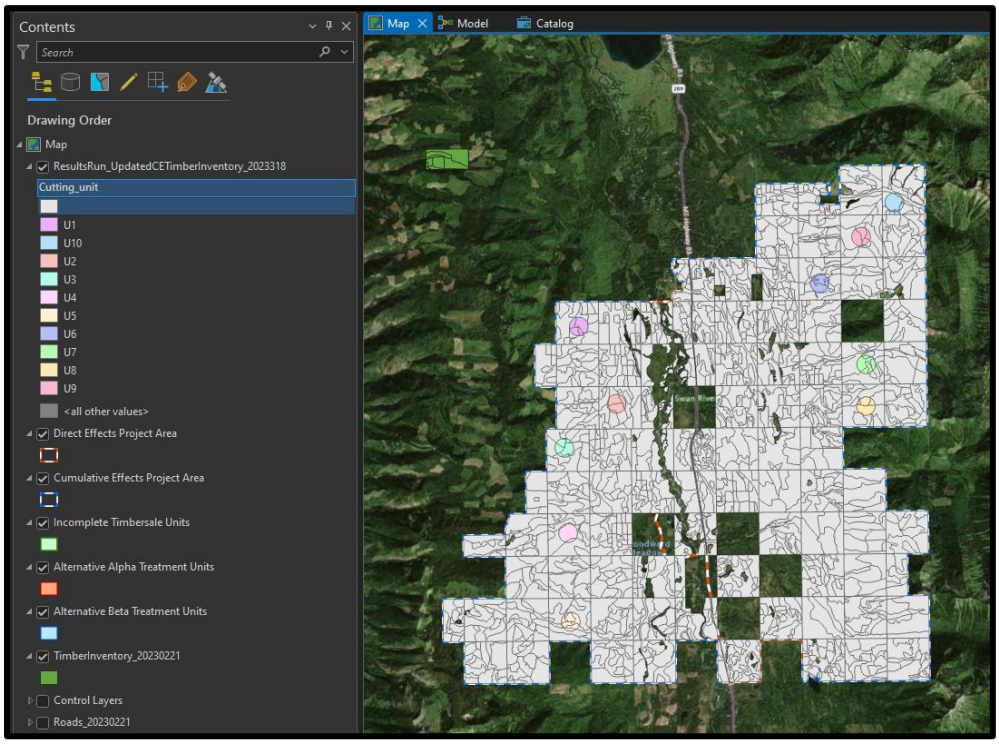


Figure 13 shows the GIS environment before implementing the data prep tool. Datasets are detailed in the contents pane and inputs are shown in the geoprocessing tool in the middle right of the window. Green circles in this map represent incomplete timber sales, blue and red circles represent alternative Alpha and Alternative Beta proposed treatment units.

The expected result of this tool was to output three distinct feature classes within the Streamline_Veg_INPUTS geodatabase that follow the naming convention described at the beginning of *Appendix A*. Visually the expected result was to show a feature layer of the Stand Level Inventory (SLI) in which you could symbolize the layer by cutting unit. This would show the 10 theoretical inputs as cutting units that maintain the timber inventory boundaries, cutting unit boundaries, while maintaining all attributes of both features as shown in figure 14.



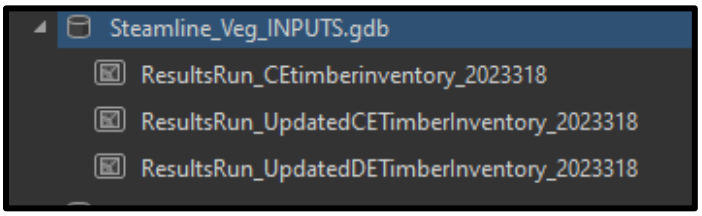


Figure 14 shows the updated timber inventory layer for use in later geoprocessing tools and the outputs of the feature class. The Streamline_Veg_INPUTS.gdb was designated as the output geodatabase location in the user parameters of the data prep tool. Individual cutting units are highlighted in pastel colors (U1-U10)

To further investigate the effectiveness of this tool, one of the 16 attribute fields that have been altered was selected to investigate the expected change based on Rx from the incomplete timber sale units. For ease of symbolizing and clarity of the breaks, the age class (AGECLASS) field was chosen. Based on the methodology described in the methods section, the tool itself, and *Appendix C*; areas that were treated with seed tree, clear cut, or overstory removal (OSR) treatments would be represented in the 0 to 39 years at model run age class. When symbolized on the age class field and units with this Rx designation show that they have been updated to reflect their potential post-harvest conditions (figure 15). There were 5 units treated with adequate silvicultural Rx that would cause change in age class. It was possible to investigate the rest of the incomplete harvest units where age class remained unaffected by running the tool as these areas would show up falling under a treatment area, but current age class would not change.

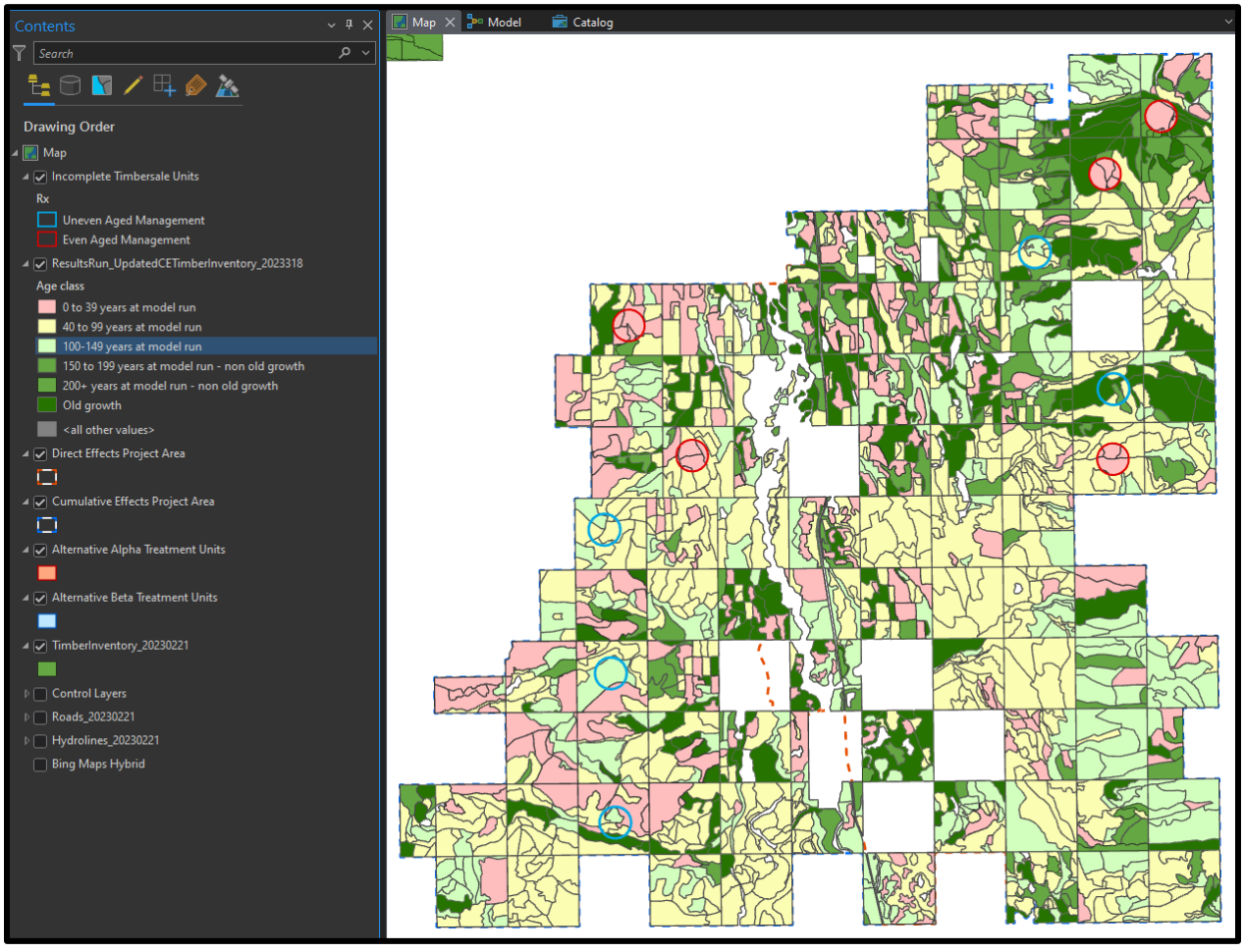


Figure 15 shows updated cumulative effects timber inventory dataset symbolized based on the age class field. The red circles represent incomplete units that are being treated with clear cut, seed tree, or overstory removal (OSR) treatments. The blue circles represent incomplete units that are being treated with all other "uneven aged management" treatments.

The data prep tool has shown to significantly reduce the time required to implement a data freeze on the local datasets in the initial stages of planning for the vegetation analysis. By reducing the amount of individual geoprocessing tools that need to be ran, and coding the post-harvest change criteria into the Python script that is manipulating the timber inventory layer, the potential for inconsistencies in how data is prepared for the analysis is minimized. By using this geoprocessing tool for data prep, it is easier to have a reproducible result between members of an ID team. In later iterations of testing, replicability can be examined by looking at the trend of acres across varying cumulative effects project areas at the state. Overall, the tool behaves as expected and the outputs of this tool, ResultsRun_UpdatedCETimberInventory_20230221 and ResultsRun_UpdatedDETimberInventory_20230221 can be used in the following vegetation analysis processes.

Forest habitat group results

After completing the baseline data prep tool, the updated cumulative effects, and direct effects timber inventory layers can be input into the forest habitat group geoprocessing tool. To check the expected result of this tool, the original timber inventory dataset was clipped to both the cumulative effects and direct effects project area. This process allowed the calculation of total forested acres within those areas. The cumulative effects area calculated out to ~54,260 acres and the direct effects area calculated out to ~13,021 acres[15]. These numbers will serve as a baseline for which summary tables will be compared for the rest of the project. The expectation is that the total number of acres will not change regardless of the forest attribute that is being summarized. The key input parameters for this run were the updated timber inventories for the cumulative effects and direct effects areas, detailed in figure 16. Upon running the tool an output Excel table was created in a folder on the disk that was designated in the tool parameters. The resulting Excel table found that the acreage does match up with the expected result for both the cumulative effects and direct effects project areas (table 1), and the habitat type parameters return all westside codes. This is a telltale sign in the dataset that is expected due to the geographic location of the project areas that shows no incorrect areas were included in the geoprocessing tool.

[15] Acres for the cumulative effects and direct effects project areas are rounded to the nearest acre for ease of understanding. Actual acres may minutely change based on presence of sliver acres and geoprocessing tool execution.

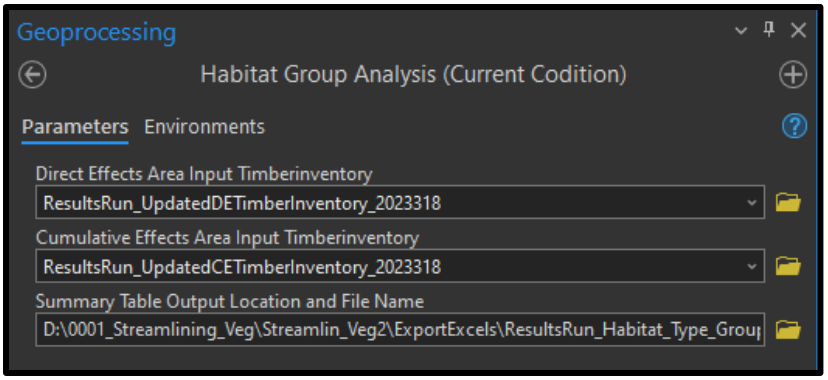


Figure 16 shows the inputs and file path for the output Excel that the summary table will be sent to upon completion.

Table 1: Habitat Type group summary table

Direct Effects Analysis Area				Cumulative Effects Analysis Area			
OBJECTID	Habitat type group	FREQUEN	SUM_ACRES	OBJECTID	Habitat type group	FREQUENCY	SUM_ACRES
1	No habitat type/class	1	21	1	No habitat type/class	1	21
2	Moderately warm and dry (westside)	29	594	2	Cold (westside)	2	90
3	Moderately cool and dry (westside)	1	80	3	Moderately warm and dry (westside)	86	3013
4	Warm and moist (westside)	363	9601	4	Moderately cool and dry (westside)	11	433
5	Cool and moist (westside)	66	1549	5	Warm and moist (westside)	1134	34344
6	Wet (westside)	32	842	6	Cool and moist (westside)	356	11628
7	Moderately cool and moist (westside)	9	328	7	Wet (westside)	48	1191
8	Cool and moderately dry (westside)	1	6	8	Moderately cool and moist (westside)	20	676
		Total	13021	9	Cool and moderately dry (westside)	49	2104
				10	Cold and moderately dry (westside)	22	760
						Total	54260

Table 1 shows the results of running the geoprocessing tool. Acreages were rounded to the nearest whole acre to produce even results. *Some discrepancy in acres is expected due to processing of the base dataset.

Of all the analysis processes that were completed during the vegetation analysis, the current condition summary tables are the least intensive and require the least amount of time to complete but the tool does reduce the number of steps to produce both tables as well as serving as a baseline for the rest of the analysis moving forward, and the output Excel file path parameter allows end users to quickly designate an area in their home file directory. This will allow for greater organization while allowing the end user to develop a final naming convention that will keep the project file uncluttered. This is because mistakes in creating the summary tables by hand can be minimized and intermediate and draft summary tables will not need to be created. The tool behaves as expected and the summary acres created by the tool match those created by hand.

Desired future condition results

Continuing, the desired future condition, current condition tool was ran by inputting the same feature classes as the habitat group analysis tool (figure 17). The expected results of this tool are largely the same as the previous run in that there should be ~13,021 and ~54,260 acres within the cumulative effects and direct effects project areas respectively. The run completed without issue and output the Excel files into the designated folder. The output Excels tables showed that the totals did match the expected result as shown in table 2.

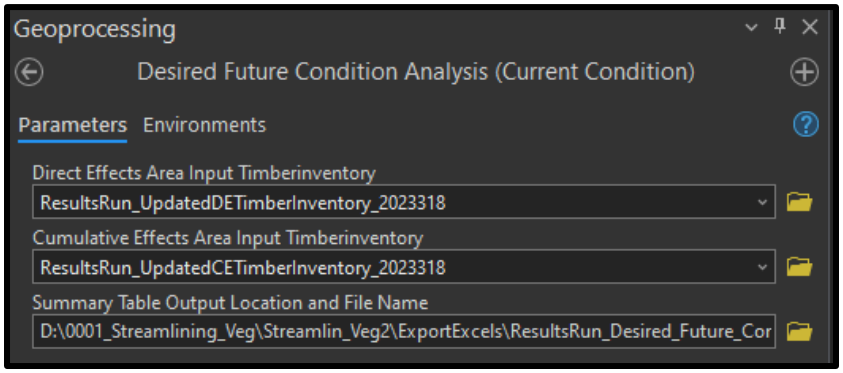


Figure 17 shows the inputs and file path for the output Excel that the summary table will be sent to upon completion.

Table 2: desired future conditions summary table

Direct Effects Analysis Area			
OBJECTID	Preferred future veg type	FREQUENCY	SUM_ACRES
1	Douglas fir	9	169
2	Hardwood	1	54
3	Lodgepole pine	16	221
4	Mixed conifer	81	2338
5	Ponderosa pine	83	2036
6	Western larch/Douglas fir	142	4286
7	Western white pine	170	3916
		Total	13021

Cumulative Effects Analysis Area			
OBJECTID	Preferred future veg type	FREQUENCY	SUM_ACRES
1	Douglas fir	33	1110
2	Hardwood	2	134
3	Lodgepole pine	73	1832
4	Mixed conifer	275	8907
5	Ponderosa pine	136	3808
6	Subalpine fir	109	4226
7	Western larch/Douglas fir	608	20792
8	Western white pine	493	13452
		Total	54260

Table 2 shows the results of running the geoprocessing tool. Acreages were rounded to the nearest whole acre to produce even results. *Some discrepancy in acres is expected due to processing of the base dataset.

The resulting table shows the total acres expected as well as some differences in the preferred future vegetation type between the cumulative effects and direct effects project areas. This change is also expected with the addition of the subalpine fir cover type because the direct effects area covers most of the low-lying acres across the Swan River State Forest and this cover type usually occurs at

higher and cooler elevations. The “correct” answers to each step are easily described and can be verified by adding the total acres of the preferred vegetation types to equal the total forested acreage in the cumulative effects and direct effects project area. Like the previous tool, this process would give an ID team member the ability to quickly run the summary tool and both summary tables will be output, leading to a more efficient and streamlined analysis.

Current cover type results

The next geoprocessing tool had some different requirements in that this is the first forest attribute that could be affected by prescribed silvicultural treatment. The overall process for determining the initial current condition was like the first two workflows, but an additional two parameters were included in the results run as shown in figure 18. These parameters give the script the key attributes to make changes to newly created fields that will quantify the post-harvest treatment conditions of the affected forested stands. The resulting output summary tables showed the expected results but were a bit harder to understand because of the complexity of how the acres changed from current conditions to post harvest treatment. This is shown in tables 3 through 7.

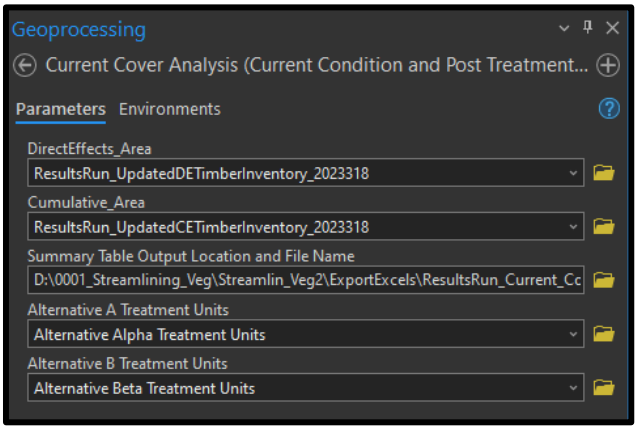


Figure 18 shows the inputs and file path for the output Excel that the summary table will be sent to upon completion.

Table 3: Current cover summary table

Direct Effects Analysis Area Current Cover			
OB	Lozensky type	FREQUENCY	SUM_ACRES
1		1	0
2	Douglas fir	15	536
3	Hardwood	4	93
4	Lodgepole pine	22	449
5	Mixed conifer	296	7136
6	Nonstocked	5	56
7	Ponderosa pine	89	1494
8	Subalpine fir	1	6
9	Western larch/Douglas fir	91	2272
10	Western white pine	46	979
		Total	13021

Cumulative Effects Analysis Area Current Cover			
OBJECTID	Lozensky type	FREQUENCY	SUM_ACRES
1		1	0
2	Douglas fir	99	3503
3	Hardwood	6	195
4	Lodgepole pine	74	2222
5	Mixed conifer	880	25556
6	Nonstocked	23	847
7	Ponderosa pine	119	2370
8	Subalpine fir	139	5610
9	Western larch/Douglas fir	332	10165
10	Western white pine	124	3792
		Total	54260

Table 4: Post treatment cover type summary table alternative Alpha

Direct Effects Analysis Area Post Treatment Cover Alternative Alpha

OB	Cover Type Post Treatment	FREQUENCY	SUM_ACRES
1		1	0
2	Douglas fir	15	536
3	Hardwood	4	93
4	Lodgepole pine	22	407
5	Mixed conifer	283	6924
6	Nonstocked	5	56
7	Ponderosa pine	82	1449
8	Subalpine fir	1	6
9	Western larch/Douglas fir	104	2513
10	Western white pine	53	1037
		Total	13021

Cumulative Effects Analysis Area Post Treatment Cover Alternative Alpha

OBJECTID	Cover Type Post Treatment	FREQUENCY	SUM_ACRES
1		1	0
2	Douglas fir	99	3503
3	Hardwood	6	195
4	Lodgepole pine	74	2180
5	Mixed conifer	867	25344
6	Nonstocked	23	847
7	Ponderosa pine	112	2325
8	Subalpine fir	139	5610
9	Western larch/Douglas fir	345	10406
10	Western white pine	131	3850
		Total	54260

Table 5: Post treatment cover type summary table alternative Beta

Direct Effects Analysis Area Post Treatment Cover Alternative Beta

OB	Cover Type Post Treatment	FREQUENCY	SUM_ACRES
1	Douglas fir	18	526
2	Hardwood	4	93
3	Lodgepole pine	23	432
4	Mixed conifer	287	6898
5	Nonstocked	5	56
6	Ponderosa pine	85	1554
7	Subalpine fir	1	6
8	Western larch/Douglas fir	96	2377
9	Western white pine	56	1078
		Total	13021

Cumulative Effects Analysis Area Post Treatment Cover Alternative Beta

OBJECTID	Cover Type Post Treatment	FREQUENCY	SUM_ACRES
1	Douglas fir	102	3493
2	Hardwood	6	195
3	Lodgepole pine	75	2205
4	Mixed conifer	871	25318
5	Nonstocked	23	847
6	Ponderosa pine	115	2430
7	Subalpine fir	139	5610
8	Western larch/Douglas fir	337	10270
9	Western white pine	134	3891
		Total	54260

Table 6: Current and post treatment change summary table alternative Alpha

Direct Effects Analysis Area Post Treatment Change in Cover Alternative Alpha

OB	Lozensky type	Cover Type Post Treatment	FREQUENCY	SUM_ACRES
1			1	0
2	Douglas fir	Douglas fir	15	536
3	Hardwood	Hardwood	4	93
4	Lodgepole pine	Lodgepole pine	21	404
5	Lodgepole pine	Western larch/Douglas fir	1	45
6	Mixed conifer	Mixed conifer	283	6924
7	Mixed conifer	Western larch/Douglas fir	9	170
8	Mixed conifer	Western white pine	4	42
9	Nonstocked	Nonstocked	5	56
10	Ponderosa pine	Ponderosa pine	81	1434
11	Ponderosa pine	Western larch/Douglas fir	6	43
12	Ponderosa pine	Western white pine	2	17
13	Subalpine fir	Subalpine fir	1	6
14	Western larch/Douglas fir	Lodgepole pine	1	3
15	Western larch/Douglas fir	Ponderosa pine	1	15
16	Western larch/Douglas fir	Western larch/Douglas fir	86	2218
17	Western larch/Douglas fir	Western white pine	3	36
18	Western white pine	Western larch/Douglas fir	2	37
19	Western white pine	Western white pine	44	941
			Total	13021

Cumulative Effects Analysis Area Post Treatment Change in Cover Alternative Alpha

OBJECTID	Lozensky type	Cover Type Post Treatment	FREQUENCY	SUM_ACRES
1			1	0
2	Douglas fir	Douglas fir	99	3503
3	Hardwood	Hardwood	6	195
4	Lodgepole pine	Lodgepole pine	73	2178
5	Lodgepole pine	Western larch/Douglas fir	1	45
6	Mixed conifer	Mixed conifer	867	25344
7	Mixed conifer	Western larch/Douglas fir	9	170
8	Mixed conifer	Western white pine	4	42
9	Nonstocked	Nonstocked	23	847
10	Ponderosa pine	Ponderosa pine	111	2310
11	Ponderosa pine	Western larch/Douglas fir	6	43
12	Ponderosa pine	Western white pine	2	17
13	Subalpine fir	Subalpine fir	139	5610
14	Western larch/Douglas fir	Lodgepole pine	1	3
15	Western larch/Douglas fir	Ponderosa pine	1	15
16	Western larch/Douglas fir	Western larch/Douglas fir	327	10111
17	Western larch/Douglas fir	Western white pine	3	36
18	Western white pine	Western larch/Douglas fir	2	37
19	Western white pine	Western white pine	122	3754
			Total	54260

Table 7: Current and post treatment change summary table alternative Beta

Direct Effects Analysis Area Post Treatment Change in Cover Alternative Beta

OB	Lozensky type	Cover Type Post Treatment	FREQUENCY	SUM_ACRES
1	Douglas fir	Douglas fir	18	526
2	Douglas fir	Mixed conifer	1	9
3	Douglas fir	Western larch/Douglas fir	1	1
4	Hardwood	Hardwood	4	93
5	Lodgepole pine	Lodgepole pine	21	415
6	Lodgepole pine	Mixed conifer	2	34
7	Mixed conifer	Lodgepole pine	2	17
8	Mixed conifer	Mixed conifer	283	6830
9	Mixed conifer	Ponderosa pine	5	98
10	Mixed conifer	Western larch/Douglas fir	6	142
11	Mixed conifer	Western white pine	4	49
12	Nonstocked	Nonstocked	5	56
13	Ponderosa pine	Ponderosa pine	78	1427
14	Ponderosa pine	Western larch/Douglas fir	2	21
15	Ponderosa pine	Western white pine	4	46
16	Subalpine fir	Subalpine fir	1	6
17	Western larch/Douglas fir	Mixed conifer	1	25
18	Western larch/Douglas fir	Ponderosa pine	2	29
19	Western larch/Douglas fir	Western larch/Douglas fir	87	2213
20	Western larch/Douglas fir	Western white pine	1	5
21	Western white pine	Western white pine	47	979
			Total	13021

Cumulative Effects Analysis Area Post Treatment Change in Cover Alternative Beta

OBJECTID	Lozensky type	Cover Type Post Treatment	FREQUENCY	SUM_ACRES
1	Douglas fir	Douglas fir	102	3493
2	Douglas fir	Mixed conifer	1	9
3	Douglas fir	Western larch/Douglas fir	1	1
4	Hardwood	Hardwood	6	195
5	Lodgepole pine	Lodgepole pine	73	2188
6	Lodgepole pine	Mixed conifer	2	34
7	Mixed conifer	Lodgepole pine	2	17
8	Mixed conifer	Mixed conifer	867	25250
9	Mixed conifer	Ponderosa pine	5	98
10	Mixed conifer	Western larch/Douglas fir	6	142
11	Mixed conifer	Western white pine	4	49
12	Nonstocked	Nonstocked	23	847
13	Ponderosa pine	Ponderosa pine	108	2303
14	Ponderosa pine	Western larch/Douglas fir	2	21
15	Ponderosa pine	Western white pine	4	46
16	Subalpine fir	Subalpine fir	139	5610
17	Western larch/Douglas fir	Mixed conifer	1	25
18	Western larch/Douglas fir	Ponderosa pine	2	29
19	Western larch/Douglas fir	Western larch/Douglas fir	328	10106
20	Western larch/Douglas fir	Western white pine	1	5
21	Western white pine	Western white pine	125	3792
			Total	54260

Table 5-7 shows the results of running the geoprocessing tool. Acreages were rounded to the nearest whole acre to produce even results. *Some discrepancy in acres is expected due to processing of the base dataset.

The resulting tables all represent the correct acreage for the cumulative effects and direct effects areas as shown by the total fields above. The difficulty in interpretation lies in looking at the last 4 tables in the series where the sum_acres column shows the total acres and the Losensky type (current cover type). The cover type post treatment columns show the first column is the current condition and the second column is the condition the treatment will cause. This means that a record with the first column showing Douglas-fir and the second column showing mixed conifer, the acres in the sum acres field are the number of acres of Douglas-fir being changed to mixed conifer. The results of this shown above show that total acreage change is zero and the acres of cover type are merely moved around depending on the treatments implemented with each alternative.

For an environmental impact statement, these cover type changes are important because of rules and regulations set by the State Forest Land Management Plan (SFLMP) and Administrative Rules that require sustainable treatment of biodiverse forests while actively moving stands towards their desired future conditions. This tool drastically reduces the amount of trial and error that was historically used to complete this same analysis by hand by automatically placing the summary tables into the output Excel. Traditionally acreages were summarized within the Excel environment using pivot tables after exporting the raw data. When analysis was completed before, the summary of acres and the change in those acres might be off by several acres depending on how the data was selected before exporting. Though the resulting tables are difficult to understand without explanation, foresters and ID

team members will be able to interpret the results and present the information in the final environmental impact statement once the tool is run. Overall, the tool behaves as expected and future tweaks to the code could allow for a more instantly understood summary table.

Forest age class results

After current cover type analysis was completed the cumulative effects and direct effects updated timber inventory feature classes were passed to the forest age class analysis tool as input parameters. The starting environment before running the tool is shown in figure 19. Like the cover type tool, the input parameters are the same except for the final summary table output file name. To keep the final summary tables organized, the results run name was used to lead the file name and Age_Class_Analysis was added afterwards. All output Excels followed this same naming convention. The program run completed without issue and summary tables were output into the designated file directory. Upon initial inspection, the results from the table were as expected, with the cumulative effects area showing ~54,260 acres and ~13,021 acres respectively as shown in tables 8 through 12.

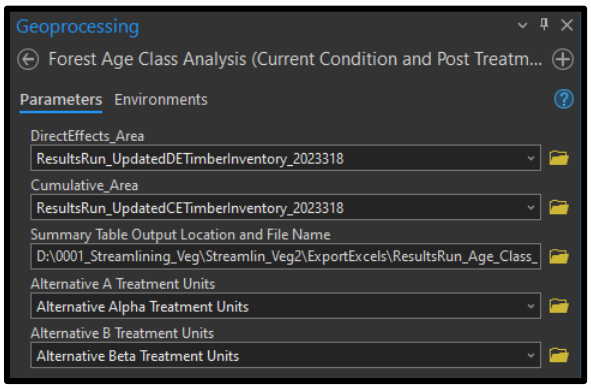


Figure 19 shows the inputs and file path for the output Excel that the summary table will be sent to upon completion.

Table 8: Current age class summary table

Direct Effects Analysis Area Current Age Class			
OBJECTID	Age class	FREQUENCY	SUM_ACRES
1		1	0
2	0 to 39 years at model run	115	1845
3	40 to 99 years at model run	220	5712
4	100-149 years at model run	100	2292
5	150 to 199 years at model run - non old growth	43	789
6	200+ years at model run - non old growth	7	153
7	Old growth	84	2231
		Total	13021

Cumulative Effects Analysis Area Current Age Class			
OBJECTID	Age class	FREQUENCY	SUM_ACRES
1		1	0
2	0 to 39 years at model run	296	8403
3	40 to 99 years at model run	740	22177
4	100-149 years at model run	340	11126
5	150 to 199 years at model run - non old growth	140	3734
6	200+ years at model run - non old growth	46	1436
7	Old growth	234	7383
		Total	54260

Table 9: Post treatment age class summary table alternative Alpha

Direct Effects Analysis Area Post Treatment Age Class Alternative Alpha			
OBJECTID	Age Class Post Treatment	FREQUENCY	SUM_ACRES
1		1	0
2	0 to 39 years at model run	139	2154
3	100-149 years at model run	90	2189
4	150 to 199 years at model run - non old growth	41	721
5	200+ years at model run - non old growth	7	153
6	40 to 99 years at model run	212	5587
7	Old growth	80	2216
		Total	13021

Cumulative Effects Analysis Area Post Age Class Alternative Alpha			
OBJECTID	Age Class Post Treatment	FREQUENCY	SUM_ACRES
1		1	0
2	0 to 39 years at model run	320	8713
3	100-149 years at model run	330	11023
4	150 to 199 years at model run - non old growth	138	3667
5	200+ years at model run - non old growth	46	1436
6	40 to 99 years at model run	732	22053
7	Old growth	230	7368
		Total	54260

Table 10: Post treatment age class summary table alternative Beta

Direct Effects Analysis Area Post Treatment Age Class Alternative Beta			
OBJECTID	Age Class Post Treatment	FREQUENCY	SUM_ACRES
1	0 to 39 years at model run	131	2222
2	100-149 years at model run	90	2172
3	150 to 199 years at model run - non old growth	43	767
4	200+ years at model run - non old growth	6	153
5	40 to 99 years at model run	227	5541
6	Old growth	78	2164
		Total	13021

Cumulative Effects Analysis Area Post Treatment Age Class Alternative Beta			
OBJECTID	Age Class Post Treatment	FREQUENCY	SUM_ACRES
1	0 to 39 years at model run	312	8781
2	100-149 years at model run	330	11007
3	150 to 199 years at model run - non old growth	140	3713
4	200+ years at model run - non old growth	45	1436
5	40 to 99 years at model run	747	22007
6	Old growth	228	7316
		Total	54260

Table 11: Current and post treatment age class change summary table alternative Alpha

Direct Effects Analysis Area Post Treatment Age Class Alternative Alpha				
OBJECTID	Age class	Age Class Post Treatment	FREQUENCY	SUM_ACRES
1			1	0
2	0 to 39 years at model run	0 to 39 years at model run	115	1845
3	40 to 99 years at model run	0 to 39 years at model run	8	124
4	40 to 99 years at model run	40 to 99 years at model run	212	5587
5	100-149 years at model run	0 to 39 years at model run	10	103
6	100-149 years at model run	100-149 years at model run	90	2189
7	150 to 199 years at model run - non old growth	0 to 39 years at model run	2	68
8	150 to 199 years at model run - non old growth	150 to 199 years at model run - non old growth	41	721
9	200+ years at model run - non old growth	200+ years at model run - non old growth	7	153
10	Old growth	0 to 39 years at model run	4	15
11	Old growth	Old growth	80	2216
			Total	13021

Cumulative Effects Analysis Area Post Treatment Age Class Alternative Alpha				
OBJECTID	Age class	Age Class Post Treatment	FREQUENCY	SUM_ACRES
1			1	0
2	0 to 39 years at model run	0 to 39 years at model run	296	8403
3	40 to 99 years at model run	0 to 39 years at model run	8	124
4	40 to 99 years at model run	40 to 99 years at model run	732	22053
5	100-149 years at model run	0 to 39 years at model run	10	103
6	100-149 years at model run	100-149 years at model run	330	11023
7	150 to 199 years at model run - non old growth	0 to 39 years at model run	2	68
8	150 to 199 years at model run - non old growth	150 to 199 years at model run - non old growth	138	3667
9	200+ years at model run - non old growth	200+ years at model run - non old growth	46	1436
10	Old growth	0 to 39 years at model run	4	15
11	Old growth	Old growth	230	7368
			Total	54260

Table 12: Current and post treatment age class change summary table alternative Beta

Direct Effects Analysis Area Post Treatment Age Class Alternative Beta				
OBJECTID	Age class	Age Class Post Treatment	FREQUENCY	SUM_ACRES
1	0 to 39 years at model run	0 to 39 years at model run	107	1845
2	40 to 99 years at model run	0 to 39 years at model run	14	170
3	40 to 99 years at model run	40 to 99 years at model run	227	5541
4	100-149 years at model run	0 to 39 years at model run	6	119
5	100-149 years at model run	100-149 years at model run	90	2172
6	150 to 199 years at model run - non old growth	0 to 39 years at model run	1	21
7	150 to 199 years at model run - non old growth	150 to 199 years at model run - non old growth	43	767
8	200+ years at model run - non old growth	200+ years at model run - non old growth	6	153
9	Old growth	0 to 39 years at model run	3	67
10	Old growth	Old growth	78	2164
			Total	13021

Cumulative Effects Analysis Area Post Treatment Age Class Alternative Beta				
OBJECTID	Age class	Age Class Post Treatment	FREQUENCY	SUM_ACRES
1	0 to 39 years at model run	0 to 39 years at model run	288	8403
2	40 to 99 years at model run	0 to 39 years at model run	14	170
3	40 to 99 years at model run	40 to 99 years at model run	747	22007
4	100-149 years at model run	0 to 39 years at model run	6	119
5	100-149 years at model run	100-149 years at model run	330	11007
6	150 to 199 years at model run - non old growth	0 to 39 years at model run	1	21
7	150 to 199 years at model run - non old growth	150 to 199 years at model run - non old growth	140	3713
8	200+ years at model run - non old growth	200+ years at model run - non old growth	45	1436
9	Old growth	0 to 39 years at model run	3	67
10	Old growth	Old growth	228	7316
			Total	54260

Tables 8-12: shows the results of running the geoprocessing tool. Acreages were rounded to the nearest whole acre to produce even results.

*Some discrepancy in acres is expected due to processing of the base dataset.

Like the cover type analysis tables, the resulting 10 tables from the age class analysis show the current stand conditions, and the post-treatment conditions for the cumulative effects area and direct effects area based on the alternative Alpha and alternative Beta treatments. The last 4 tables follow the same logic described in the cover type analysis section. An example of this from one age class to another would be that the Sum_Acres column for an age class of 40 to 99 years in the age class column and 0 to 39 years in the age class post treatment column would represent the number of acres changing from the 40-year class to the 0-year class. This type of analysis was often difficult to complete with little training, and inconsistencies in acres would come up when completing the process by hand. These inconsistencies were often due to individual team members not following a consistent approach to selecting acres. Age class units, and slivers might have been missed due to misunderstanding of members on how to utilize the definition query tools. This geoprocessing tool seems to minimize the chance of inconsistencies and reduces the time spent on completing the analysis. The results from running the tool were as expected.

Forest old growth results

Another important part of the environmental impact statement vegetation analysis process is to review and describe how old growth might be affected by alternatives within the project. The expected output for this tool is slightly altered from previous runs, in that the tool does not look at all the acres across the cumulative effects and direct effects project areas, but only those that are currently old growth. To test the expected result a definition query had to be placed on the original cumulative effects area updated timber inventory layer as well as the direct effects area updated timber inventory layer. The acres were then summarized. The direct effects area currently has ~2,231 acres of old growth and the cumulative effects area currently has ~7,383 acres of old growth. Upon completion of the geoprocessing tools run, the expected result for all treated and altered acres would be these two values described above.

The inputs for the old growth analysis tool are the same as the previous tools as shown in figure 20. The expected outputs for the tool are a series of tables showing the current condition of old growth within the cumulative effects and direct effects project area and post treatment condition of old growth within those same areas. The tables are summarized based on the age class and old growth post treatment fields as shown in tables 13-17. The summary tables also consider the current cover type of

the old growth stands in question. The age class and post age class field were added and summarized to make the resulting tables easier to understand.[16]

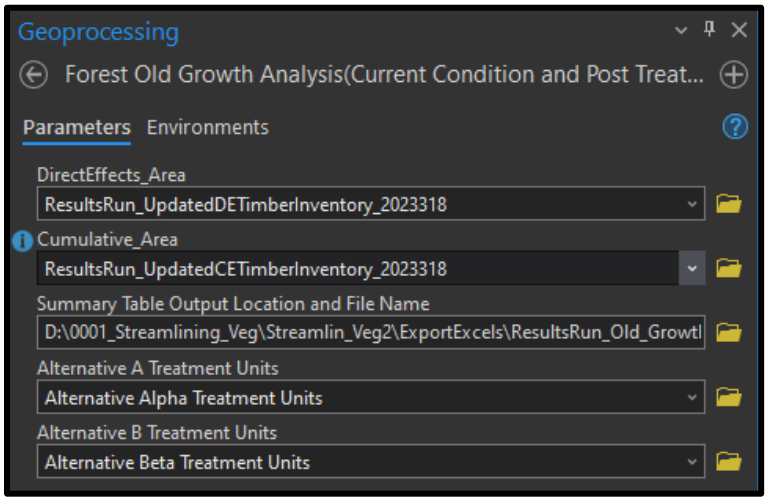


Figure 20 shows the inputs and file path for the output Excel that the summary table will be sent to upon completion.

Table 13: Current old growth summary table

Direct Effects Area Current Old Growth based on Current Cover				
OBJECTID	Age class	Lozensky type	FREQUENCY	SUM_ACRES
1	Old growth	Mixed conifer	57	1417
2	Old growth	Ponderosa pine	1	89
3	Old growth	Western larch/Douglas fir	19	406
4	Old growth	Western white pine	7	319
			Total	2231

Cumulative Effects Area Current Old Growth based on Current Cover				
OBJECTID	Age class	Lozensky type	FREQUENCY	SUM_ACRES
1	Old growth	Douglas fir	3	55
2	Old growth	Mixed conifer	175	5384
3	Old growth	Ponderosa pine	1	109
4	Old growth	Subalpine fir	11	494
5	Old growth	Western larch/Douglas fir	33	952
6	Old growth	Western white pine	11	390
			Total	7383

Table 14: Post treatment old growth by current cover type summary alternative Alpha

Direct Effects area Post Treatment Old Growth by Cover Type Alternative Alpha				
OBJECTID	Old Growth Post Treatment	Lozensky type	FREQUENCY	SUM_ACRES
1	0 to 39 years at model run	Mixed conifer	4	15
2	200+ years at model run - non oldgrowth	Mixed conifer	2	34
3	200+ years at model run - non oldgrowth	Western larch/Douglas fir	2	22
4	Old growth	Mixed conifer	51	1369
5	Old growth	Ponderosa pine	1	89
6	Old growth	Western larch/Douglas fir	17	383
7	Old growth	Western white pine	7	319
			Total	2231

Cumulative Effects area Post Treatment Old Growth by Cover Type Alternative Alpha				
OBJECTID	Old Growth Post Treatment	Lozensky type	FREQUENCY	SUM_ACRES
1	0 to 39 years at model run	Mixed conifer	4	15
2	Old growth	Douglas fir	3	55
3	Old growth	Mixed conifer	171	5369
4	Old growth	Ponderosa pine	1	109
5	Old growth	Subalpine fir	11	494
6	Old growth	Western larch/Douglas fir	33	952
7	Old growth	Western white pine	11	390
			Total	7383

[16] All tables 13-17 are considered old growth prior to treatment.

Table 15: Post treatment old growth by current cover type summary alternative Beta

Direct Effects area Post Treatment Old Growth by Cover Type Alternative Beta				
OBJECTID	Old Growth Post Treatment	Lozensky type	FREQUENCY	SUM_ACRES
1	0 to 39 years at model run	Mixed conifer	3	67
2	Old growth	Mixed conifer	53	1350
3	Old growth	Ponderosa pine	1	89
4	Old growth	Western larch/Douglas fir	17	406
5	Old growth	Western white pine	7	319
			Total	2231

Cumulative Effects area Post Treatment Old Growth by Cover Type Alternative Beta				
OBJECTID	Old Growth Post Treatment	Lozensky type	FREQUENCY	SUM_ACRES
1	0 to 39 years at model run	Mixed conifer	3	67
2	Old growth	Douglas fir	3	55
3	Old growth	Mixed conifer	171	5317
4	Old growth	Ponderosa pine	1	109
5	Old growth	Subalpine fir	11	494
6	Old growth	Western larch/Douglas fir	31	952
7	Old growth	Western white pine	11	390
			Total	7383

Table 16: Post treatment old growth by defect risk summary alternative Alpha

Direct Effects area Post Treatment Old Growth Treated Risk Alternative Alpha			
OBJECTID	Defect Risk	FREQUENCY	SUM_ACRES
1		76	2160
2	High	2	14
3	Low	4	56
4	Medium	2	1
		Total	2231

Cumulative Effects area Post Treatment Old Growth Treated Risk Alternative Alpha			
OBJECTID	Defect Risk	FREQUENCY	SUM_ACRES
1		226	7312
2	High	2	14
3	Low	4	56
4	Medium	2	1
		Total	7383

Table 17: Post treatment old growth by defect risk summary alternative Beta

Direct Effects area Post Treatment Old Growth Treated Risk Alternative Beta			
OBJECTID	Defect Risk	FREQUENCY	SUM_ACRES
1		76	2136
2	High	1	24
3	Low	3	62
4	Medium	1	9
		Total	2231

Cumulative Effects area Post Treatment Old Growth Treated Risk Alternative Beta			
OBJECTID	Defect Risk	FREQUENCY	SUM_ACRES
1		226	7288
2	High	1	24
3	Low	3	62
4	Medium	1	9
		Total	7383

Tables 13-17 shows the results of running the geoprocessing tool. Acreages were rounded to the nearest whole acre to produce even results.

*Some discrepancy in acres is expected due to processing of the base dataset.

The geoprocessing tool created 10 non-spatial tables from the old growth analysis that show the current stand conditions, and the post-treatment conditions for the cumulative effects area and direct effects area based on the alternative Alpha and alternative Beta treatments. All the acres in the summary tables are considered old growth currently and those acres that are labeled with a different age class (other than old growth) in the old growth post treatment field are considered not old growth

post treatment. The Losensky type field shows the current cover type of that acreage of old growth. There is still room for improvement on this process as the tool does not describe the change of Losensky old growth acres to the desired future condition based on the proposed silvicultural treatment type. Moving forward, this script will need to be updated and to include all expected changes. Though it does include these modifications in its current state, the results from the tool can be incorporated into the basic summary tables for old growth. This type of analysis was the most difficult to accomplish while completing the vegetation analysis by hand. There are numerous variables to keep track of during the analysis and it is extremely easy to get "lost" in the process and inconsistent numbers to be produced for the environmental impact statement. This tool does a good job at the first steps for old growth analysis, but there is much improvement that could be made to further increase the utility of the tool. This geoprocessing tool seems to minimize the chance of inconsistencies and reduces the time spent on completing the analysis. The results from running the tool were as expected.

An added benefit of the old growth analysis geoprocessing tool is that several output feature classes are added to the home environment in the GIS. Current old growth in the Swan River State Forest, remaining old growth post treatment for alternatives Alpha and Beta, as well as remove old growth post treatment for alternatives Alpha and Beta are produced. These feature layers combined can show areas of high, medium, and low risk old growth, and whether these areas will be removed from the timber inventory post-harvest (figure 21 A & B). This added functionality will also allow foresters and ID team members to run this tool several times during the planning process of the environmental impact statement and physically see how the landscape might change under a preferred alternative. This will potentially lead to more informed decisions on the ground as silvicultural prescriptions are designated for the project.

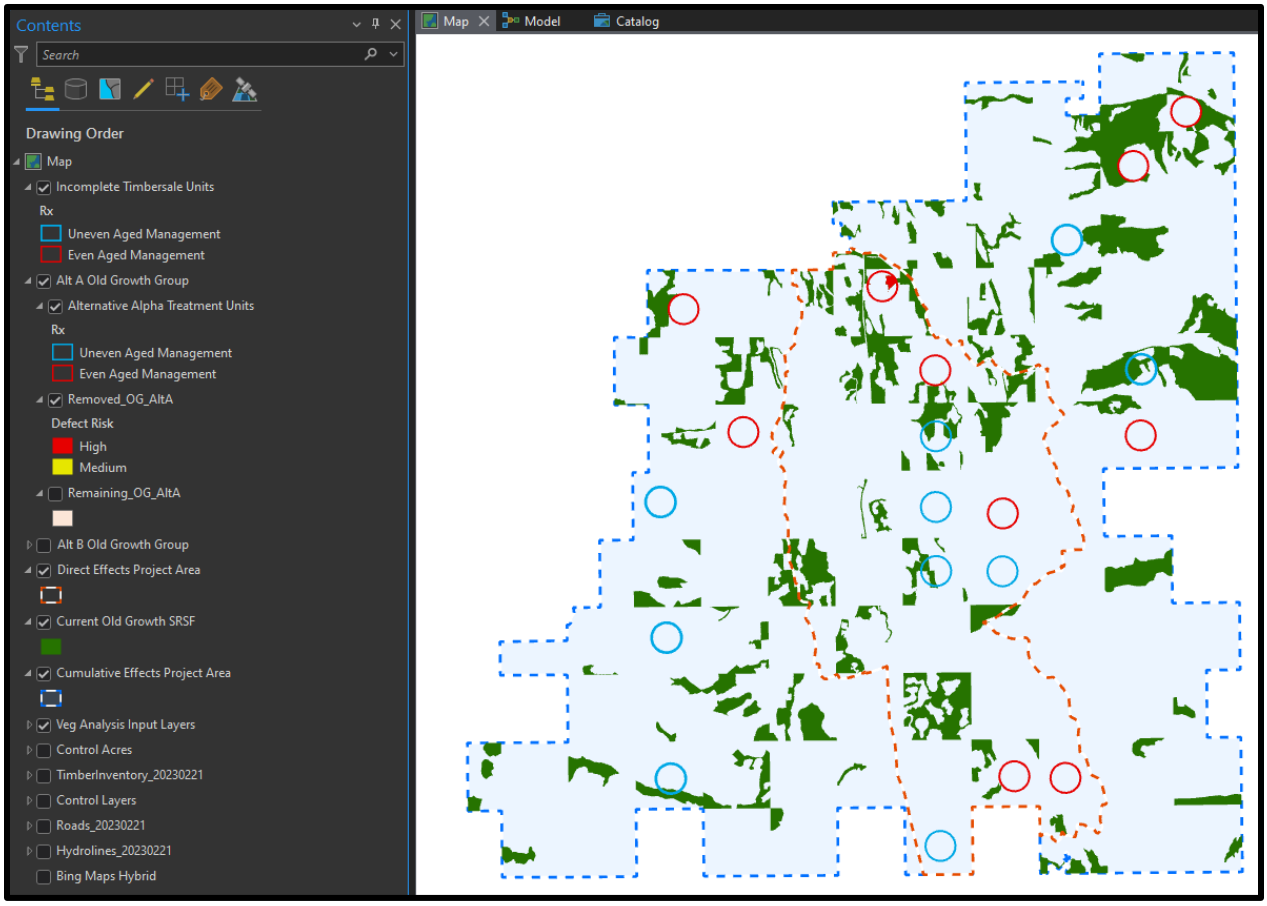


Figure 21A shows current old growth (dark green areas) on the SRSF feature layer that was output from the streamlining vegetation analysis old growth analysis tool. Bule and red circles represent incomplete and alternative Alpha treatment units that would either remove old growth (red circles) or maintain old growth (blue circles). High and medium risk old growth stands that have been removed are highlighted in the direct effects project area by bright red and yellow symbology. *A remaining old growth feature class is also produced but not symbolized in this figure.

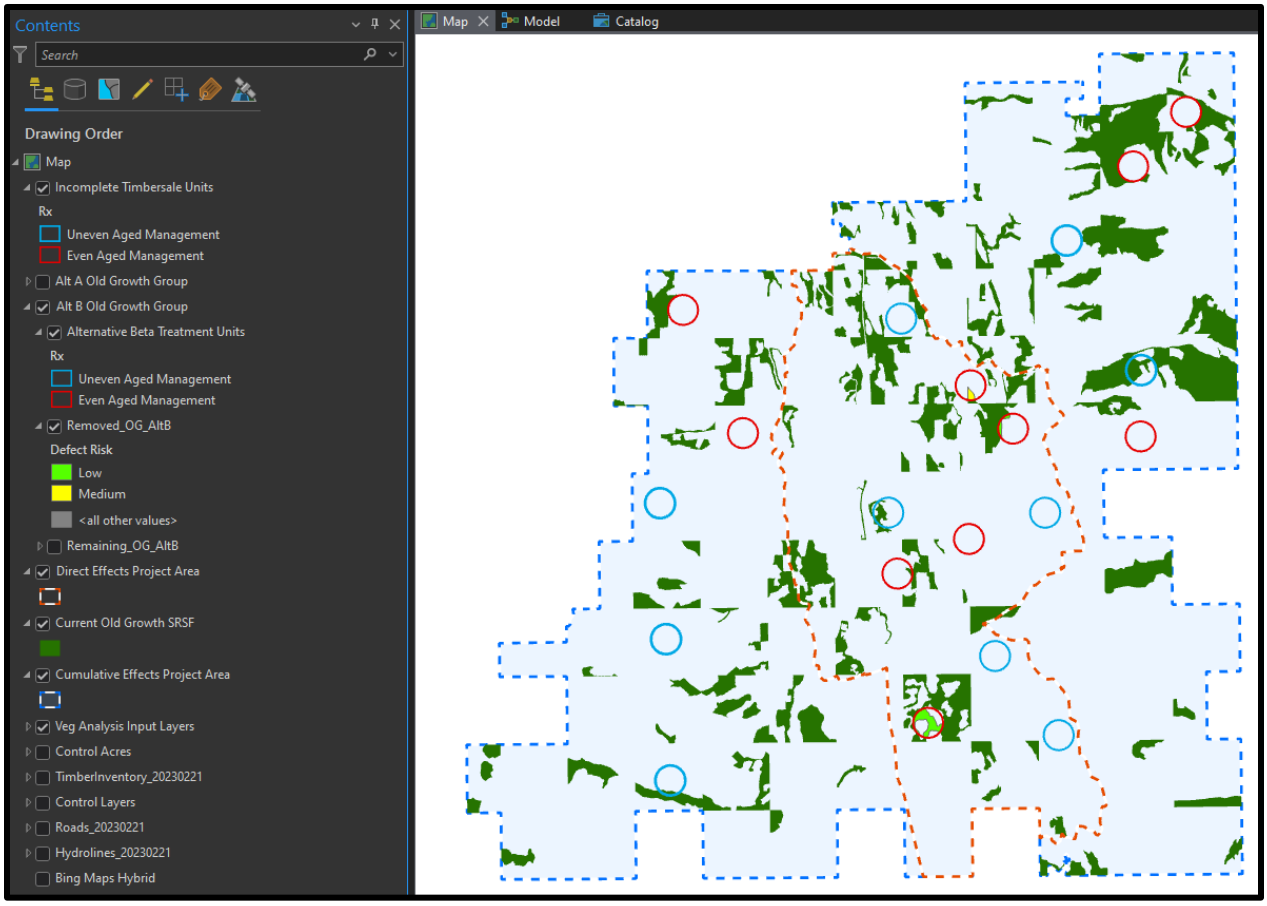


Figure 21B shows current old growth (dark green areas) on the SRSF feature layer that was output from the streamlining vegetation analysis old growth analysis tool. Bule and red circles represent incomplete and alternative Beta treatment units that would either remove old growth (red circles) or maintain old growth (blue circles). High and medium risk old growth stands that have been removed are highlighted in the direct effects project area by bright green and yellow symbology. *A remaining old growth feature class is also produced but not symbolized in this figure.

Age and cover type patch size results

The final geoprocessing tool that was created takes the cumulative effects area and direct effects area updated timber inventory layers as input parameters (figure 22) to determine the current and post-harvest effects to the average patch size for both the age class (AGECLASS) and current cover type (CVR_CURR) fields. The expected results of this tool were several output Excel tables and three distinct feature layers that show the current average patch size across age classes. Most treatments that alter age class and cover types within the cumulative effects and direct effects project area overall reduce the mean patch size for older age classes and will slightly increase patch size for younger age groups. This is because stands are more actively moved towards younger age groups with treatments, breaking up the landscape and potentially adding acres to adjacent stands with a younger age class that

were previously treated. This trend is shown in tables 18 through 20, where younger age class mean patch size slightly increases across both alternative treatments and the older age classes slightly decrease. Though this rule does not happen every time, the trend in how patch size changes show how the forest cover moves as treatment is carried out.

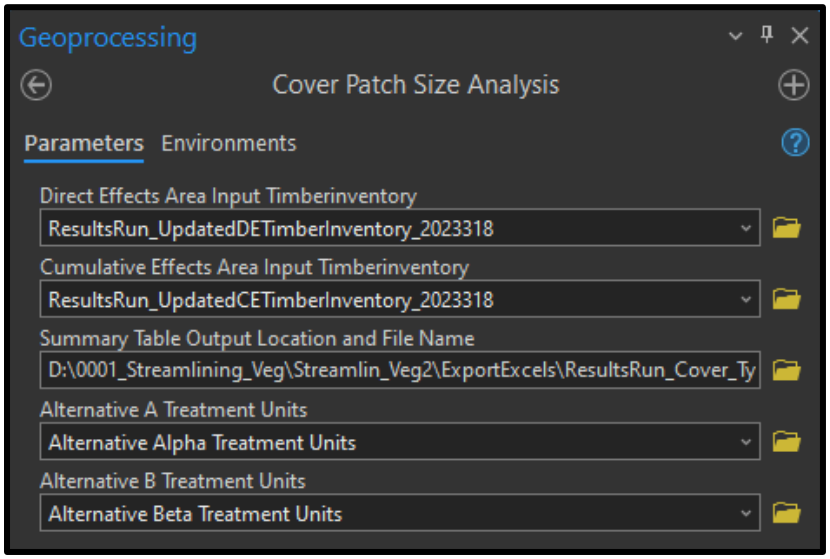


Figure 22 shows the inputs and file path for the output Excel that the summary table will be sent to upon completion. *Inputs for the age class patch analysis are exactly the same with the exception of the title of the tool and the output file name.

Table 18: Current age class patch size

Direct Effects Current Age Class Patch Size Analysis			
OBJECTID	Age class	FREQUENCY	MEAN_ACRES
1	0 to 39 years at model run	82	22
2	40 to 99 years at model run	73	78
3	100-149 years at model run	54	42
4	150 to 199 years at model run - non	32	25
5	200+ years at model run - non old gr	6	26
6	Old growth	45	50

Cumulative Effects Current Age Class Patch Size Analysis			
OBJECTID	Age class	FREQUENCY	MEAN_ACRES
1	0 to 39 years at model run	169	50
2	40 to 99 years at model run	152	146
3	100-149 years at model run	142	78
4	150 to 199 years at model run - non old growth	83	45
5	200+ years at model run - non old growth	35	41
6	Old growth	119	62

Table 19: Post treatment age class patch size alternative Alpha

Direct Effects Post Age Class Patch Size Analysis Alternative Alpha			
OBJECTID	Age Class Post Treatment	FREQUENCY	MEAN_ACRES
1		1	0
2	0 to 39 years at model run	82	26
3	100-149 years at model run	54	41
4	150 to 199 years at model run - non	33	22
5	200+ years at model run - non old gr	6	26
6	40 to 99 years at model run	73	77
7	Old growth	46	48

Cumulative Effects Post Age Class Patch Size Analysis Alternative Alpha			
OBJECTID	Age Class Post Treatment	FREQUENCY	MEAN_ACRES
1		1	0
2	0 to 39 years at model run	169	52
3	100-149 years at model run	142	78
4	150 to 199 years at model run - non old growth	84	44
5	200+ years at model run - non old growth	35	41
6	40 to 99 years at model run	152	145
7	Old growth	120	61

Table 20: Post treatment age class patch size alternative Beta

Direct Effects Post Age Class Patch Size Analysis Alternative Beta			
OBJECTID	Age Class Post Treatment	FREQUENCY	MEAN_ACRES
1	0 to 39 years at model run	84	26
2	100-149 years at model run	55	39
3	150 to 199 years at model run - non	33	23
4	200+ years at model run - non old gr	6	26
5	40 to 99 years at model run	73	76
6	Old growth	45	48

Cumulative Effects Post Age Class Patch Size Analysis Alternative Beta			
OBJECTID	Age Class Post Treatment	FREQUENCY	MEAN_ACRES
1	0 to 39 years at model run	171	51
2	100-149 years at model run	143	77
3	150 to 199 years at model run - non old growth	84	44
4	200+ years at model run - non old growth	35	41
5	40 to 99 years at model run	152	145
6	Old growth	119	61

Tables 18-20 show the results of running the geoprocessing tool. Acreages were rounded to the nearest whole acre to produce even results.

*Some discrepancy in acres is expected due to processing of the base dataset.

In contrast to the age class patch size analysis, the cover type fields will not follow the same logic of moving acres from one cover type to another. Because cover type is varied across both the cumulative effects and direct effects areas, it is hard to determine which cover types are going to be more represented post-harvest than others. This is largely due to the use of the major potential vegetation field to calculate the post-harvest effects in the current cover post-harvest field. Overall, the expected result would be a slight increase in more commonly desired future cover types. In the Swan River State Forest, this is generally the western larch/Douglas-fir and western white pine cover types. This trend is shown in the alternative B treatments in tables 21-23, where there is a slight increase in patch size for the described cover types above. For alternative A, it is also expected that the mean patch size will be larger at the cumulative effects level than at the direct effects level as shown in table 22.

Table 21: Current cover mean patch size

Direct Effects Current Cover Mean Patch Size			
OBJECTID	Lozensky type	FREQUENCY	MEAN_ACRES
1	Douglas fir	10	54
2	Hardwood	4	23
3	Lodgepole pine	16	28
4	Mixed conifer	60	119
5	Nonstocked	5	11
6	Ponderosa pine	55	27
7	Subalpine fir	1	6
8	Western larch/Douglas fir	45	50
9	Western white pine	27	36

Cumulative Effects Current Cover Mean Patch Size			
OBJECTID	Lozensky type	FREQUENCY	MEAN_ACRES
1	Douglas fir	56	63
2	Hardwood	6	32
3	Lodgepole pine	39	57
4	Mixed conifer	89	287
5	Nonstocked	22	38
6	Ponderosa pine	66	36
7	Subalpine fir	25	224
8	Western larch/Douglas fir	131	78
9	Western white pine	65	58

Table 22: Post treatment cover mean patch size alternative Alpha

Direct Effects Post Cover Mean Patch Size Alternative Alpha			
OBJECTID	Cover Type Post Treatme	FREQUENCY	MEAN_ACRES
1		1	0
2	Douglas fir	10	54
3	Hardwood	4	23
4	Lodgepole pine	19	21
5	Mixed conifer	64	108
6	Nonstocked	5	11
7	Ponderosa pine	59	25
8	Subalpine fir	1	6
9	Western larch/Douglas fir	49	51
10	Western white pine	33	31

Cumulative Effects Post Cover Mean Patch Size Alternative Alpha			
OBJECTID	Cover Type Post Treatme	FREQUENCY	MEAN_ACRES
1		1	0
2	Douglas fir	56	63
3	Hardwood	6	32
4	Lodgepole pine	42	52
5	Mixed conifer	92	275
6	Nonstocked	22	38
7	Ponderosa pine	70	33
8	Subalpine fir	25	224
9	Western larch/Douglas fir	135	77
10	Western white pine	71	54

Table 23: Post treatment cover mean patch size alternative Beta

Direct Effects Post Cover Mean Patch Size Alternative Beta			
OBJECTID	Cover Type Post Treatme	FREQUENCY	MEAN_ACRES
1	Douglas fir	10	53
2	Hardwood	4	23
3	Lodgepole pine	17	25
4	Mixed conifer	59	117
5	Nonstocked	5	11
6	Ponderosa pine	59	26
7	Subalpine fir	1	6
8	Western larch/Douglas fir	50	48
9	Western white pine	29	37

Cumulative Effects Post Cover Mean Patch Size Alternative Beta			
OBJECTID	Cover Type Post Treatme	FREQUENCY	MEAN_ACRES
1	Douglas fir	56	62
2	Hardwood	6	32
3	Lodgepole pine	40	55
4	Mixed conifer	88	288
5	Nonstocked	22	38
6	Ponderosa pine	70	35
7	Subalpine fir	25	224
8	Western larch/Douglas fir	136	76
9	Western white pine	67	58

Tables 21-23 show the results of running the geoprocessing tool. Acreages were rounded to the nearest whole acre to produce even results.

*Some discrepancy in acres is expected due to processing of the base dataset.

Foresters and ID team members often include visual representation of the summary tables above. The geoprocessing tool can output three different age class patch size feature classes; ltA_Post_Age_PatchSize, AltB_Post_Age_PatchSize, and Current_Ageclass_PatchSize_CE into the home workspace geodatabase. The tool also outputs three different cover type patch size feature classes; ltA_Post_Cover_PatchSize, AltB_Post_Cover_PatchSize, and Current_Cover_PatchSize_CE. These feature classes can be placed in the mapping window of the GIS environment as shown in figure 23 and figure 24. Maps shown in this report are in the ArcGIS Pro GIS environment and do not reflect final maps that would be placed in a final environmental impact statement. Not only does the analysis process of dissolving and exploding the features become easier and less time consuming, foresters and ID team members can be sure that the analysis is completed the same way each time they decide to run a hypothetical alternative or add/remove potential harvest units in their project development process.

Of all the geoprocessing tools developed to streamline the vegetation analysis for environmental impact statements, the age class and cover type patch size seems to achieve the objectives of reducing time spent completing analysis, and reducing inconsistencies between different runs the most consistently. The tool behaves as expected and utilizing the tool reduces the amount of time and effort to complete good data management.

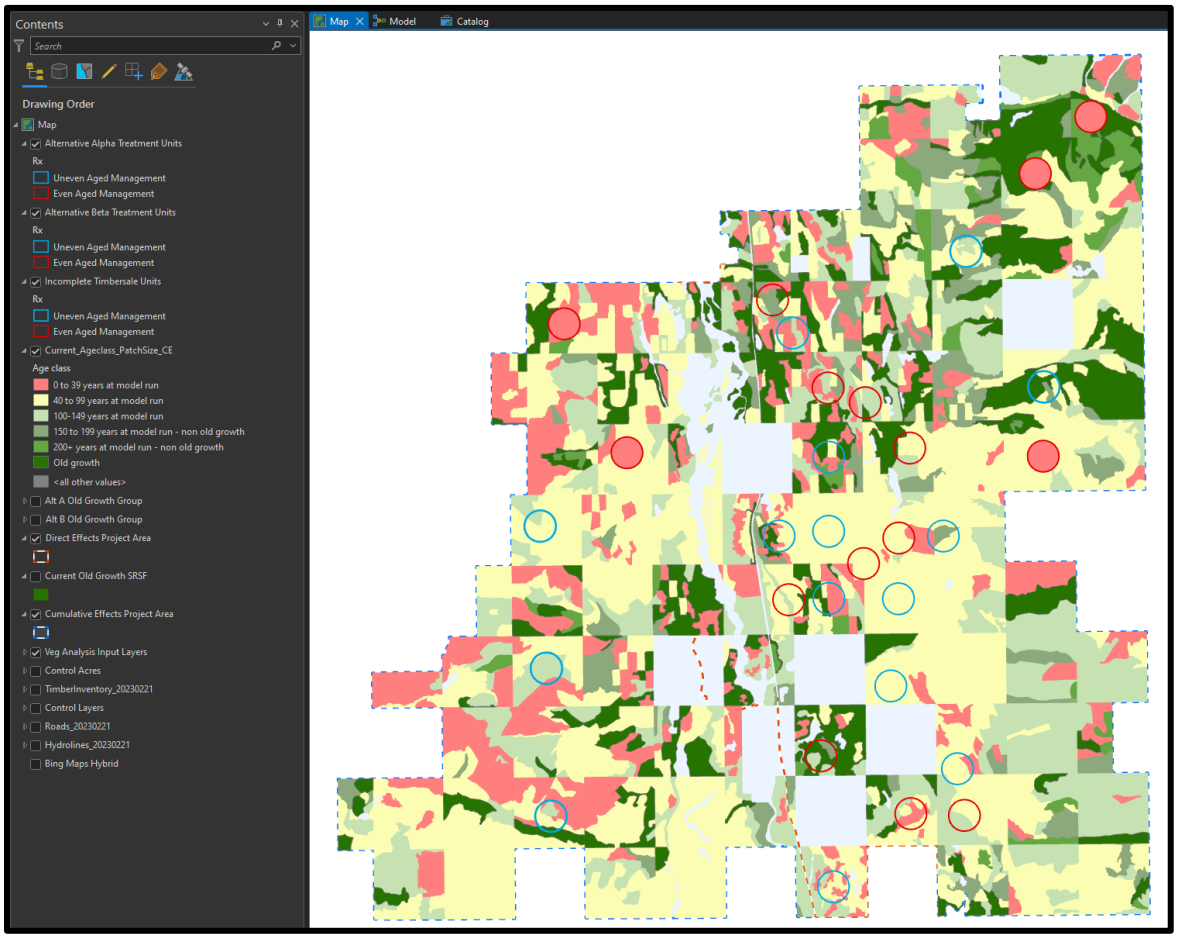


Figure 23 A shows the current age class patch size that was output from running the age class and cover type patch size analysis geoprocessing tool. Both alternative Alpha, alternative Beta, and incomplete timber sale units symbolized to show uneven aged and even aged treatments. Current conditions will take into consideration age class patch size outside of the direct effects project area.

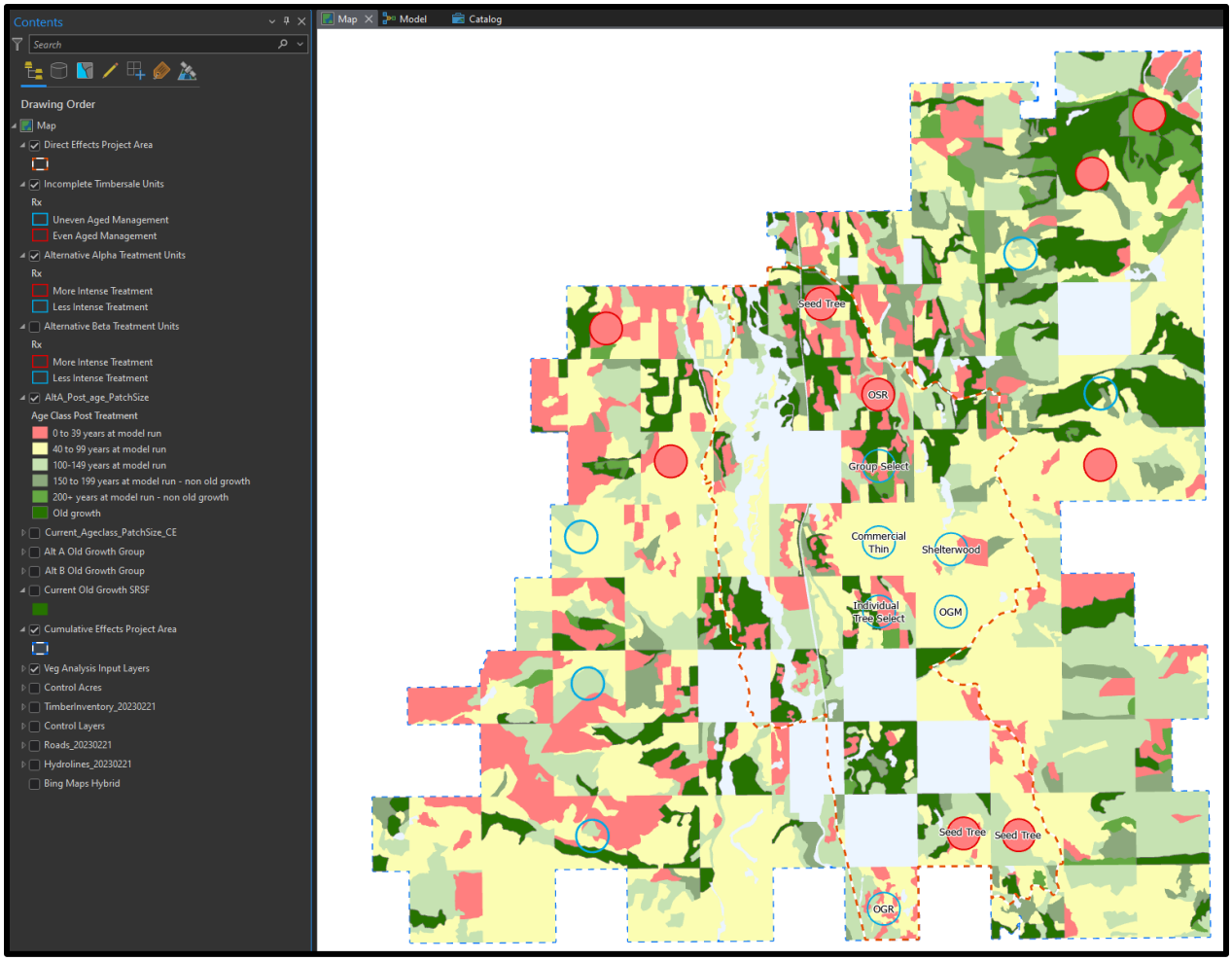


Figure 23 B shows the post alternative Alpha age class patch size that was output from running the age class and cover type patch size analysis geoprocessing tool. Alternative Alpha treatment units are symbolized to show more and less intense treatments that would alter the age class patch size post-harvest. Current conditions will take into consideration incomplete treatment effects on age class patch size outside of the direct effects project area.

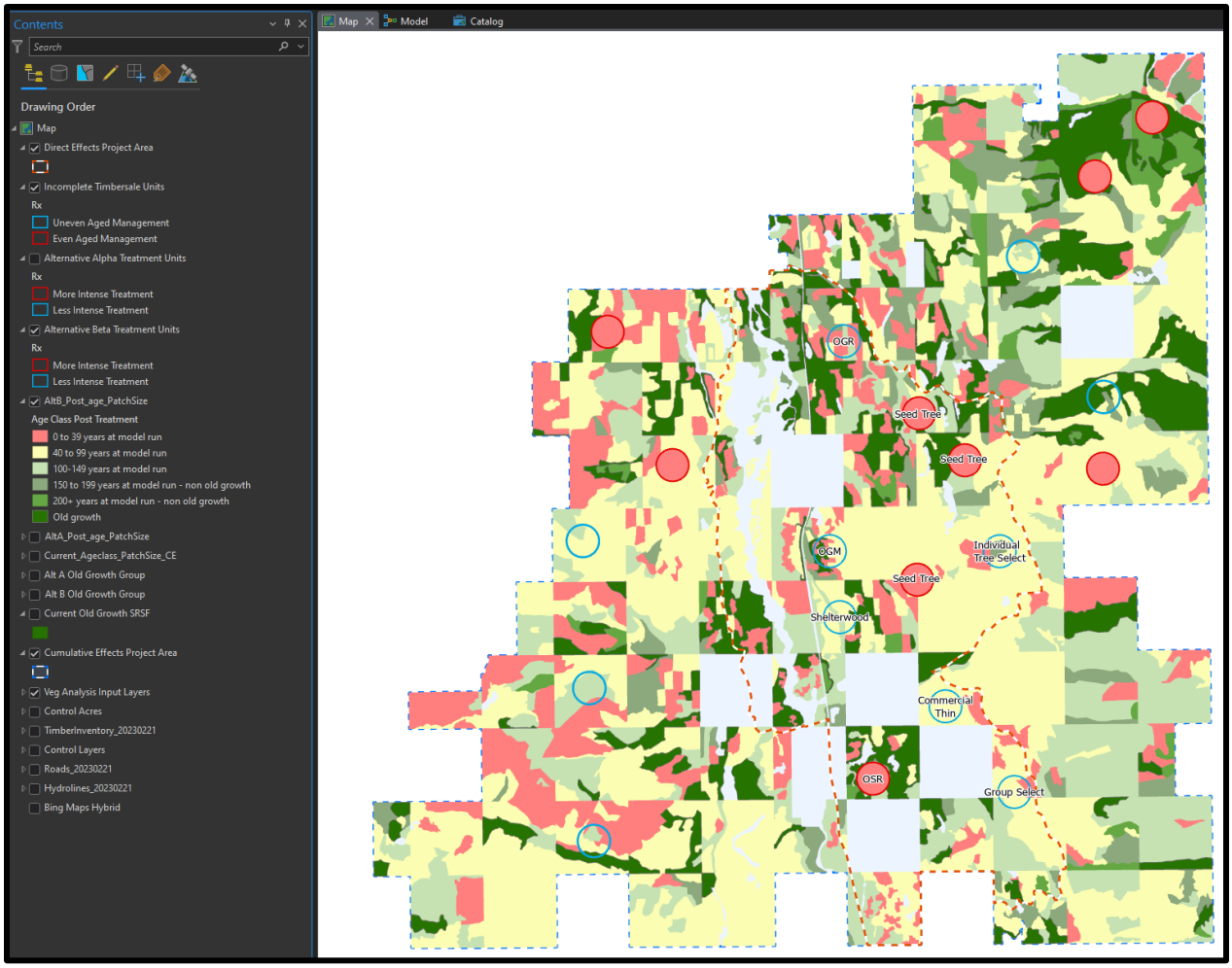


Figure 23 C shows the post alternative Beta age class patch size that was output from running the age class and cover type patch size analysis geoprocessing tool. Alternative Beta treatment units are symbolized to show more and less intense treatments that would alter the age class patch size post-harvest. Current conditions will take into consideration incomplete treatment effects on age class patch size outside of the direct effects project area.

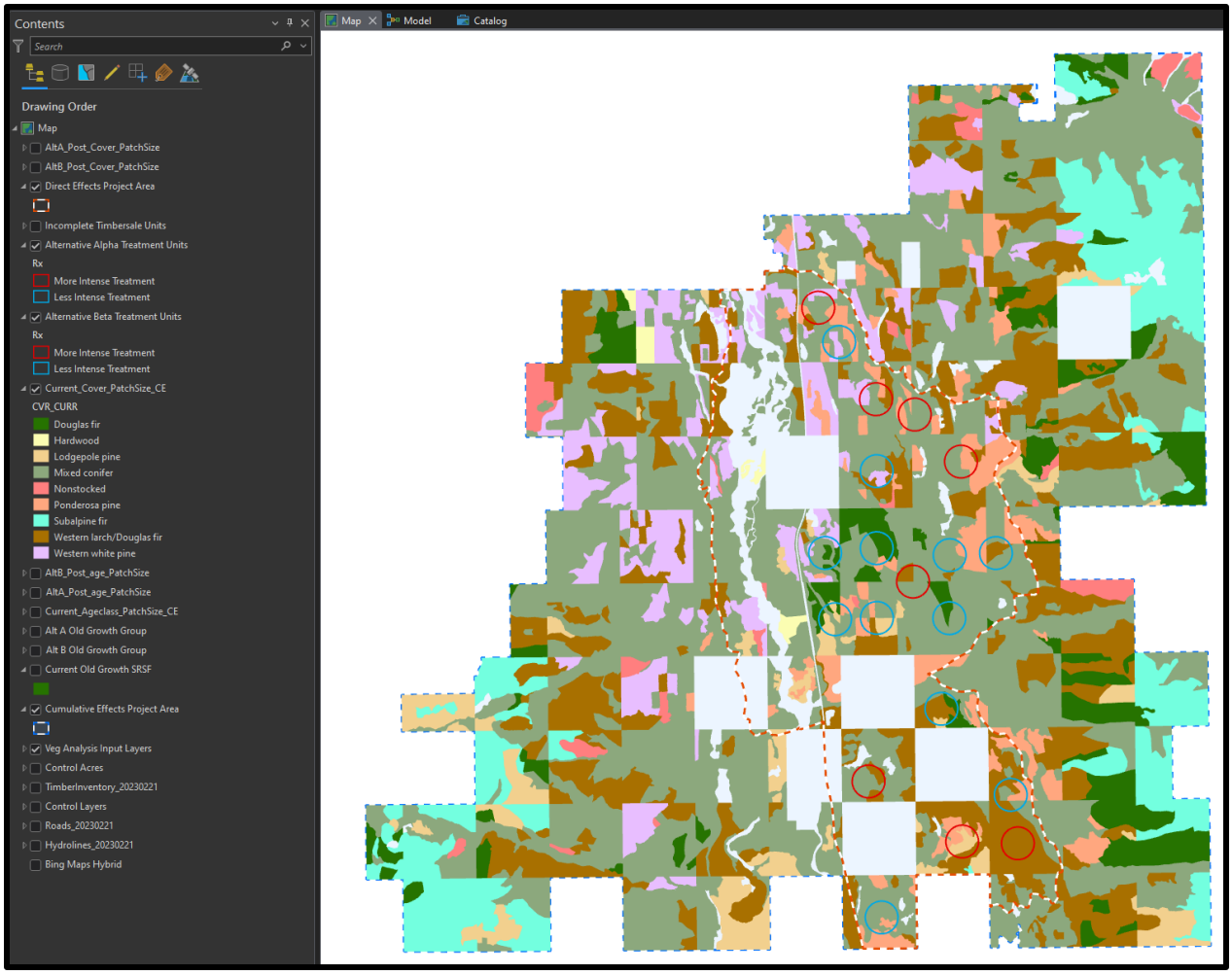


Figure 24 A shows the current cover class patch size that was output from running the age class and cover type patch size analysis geoprocessing tool. Both alternative Alpha, and alternative Beta treatment units are symbolized to more Intense and less intense treatments.

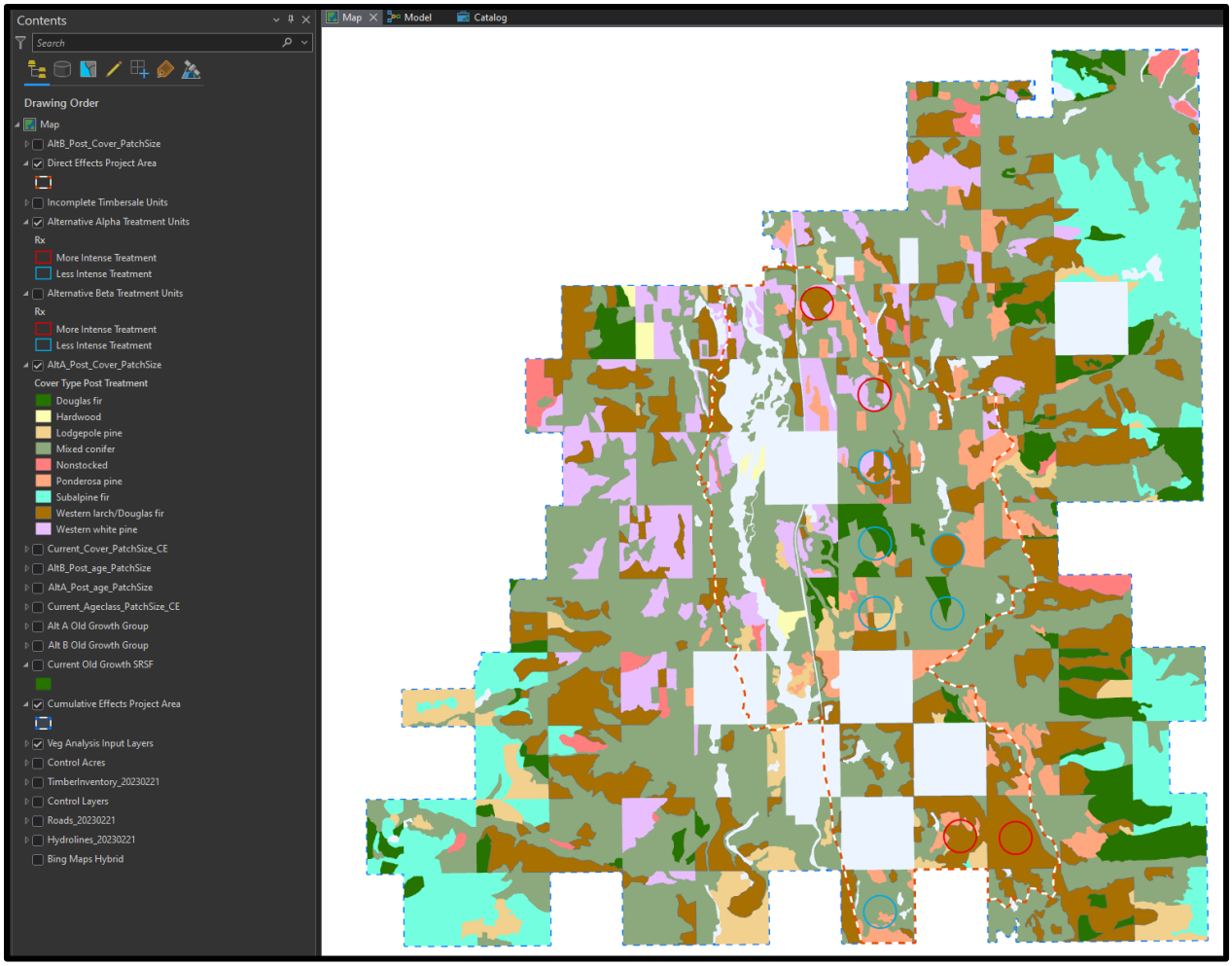


Figure 24 B shows the post-treatment alternative Alpha cover class patch size that was output from running the age class and cover type patch size analysis geoprocessing tool. alternative Alpha units highlighted with more intense treatment expect to see cover class change to move towards a desired future condition detailed in the MAJPOTVEG attribute field. *Some change might not be readily noticeable if current cover already meets desired future condition before treatment.

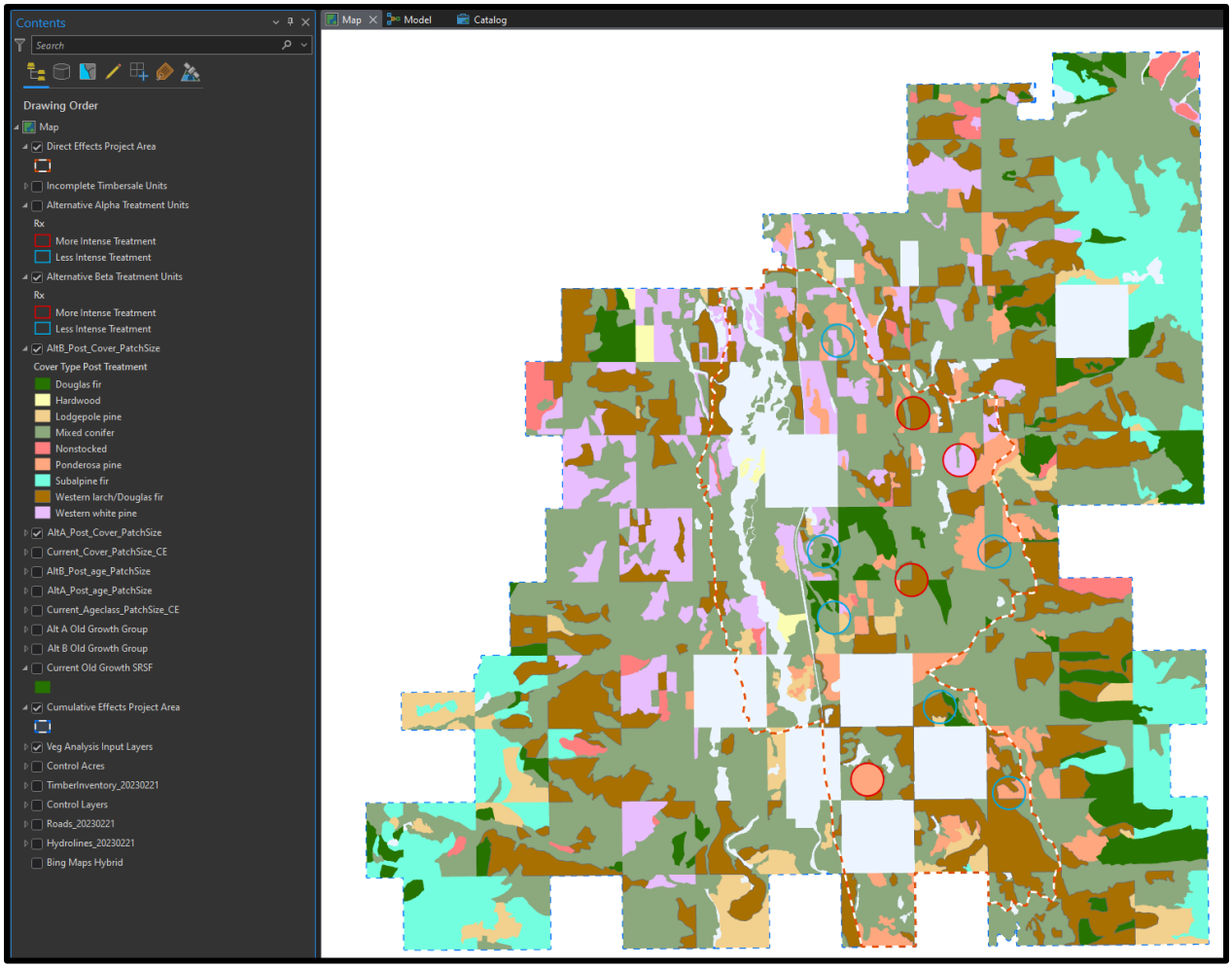


Figure 24 C shows the post-treatment alternative Beta cover class patch size that was output from running the age class and cover type patch size analysis geoprocessing tool. Alternative Beta Units highlighted with more intense treatment expect to see cover class change to move towards a desired future condition detailed in the MAJPOTVEG attribute field. *Some change might not be readily noticeable if current cover already meets desired future condition before treatment.

At the end of running each of the tools a few key takeaways could be reported. Because the tools required the same inputs, it was simple to run multiple tools back-to-back with no inconsistencies in the output tables. There is also a significant ability to utilize the program to provide a decision maker with many more alternatives. For example, the inputs for alternative Alpha and Beta are not “set” and as long as the ID team members are consistent with their naming practices and data management, multiple different scenarios and treatment units can be passed to the tool allowing users to try new ideas that they otherwise might not have tried due to time constraints. This means that foresters could design multiple different alternatives more quickly, and the potential effects of these will be more readily apparent when the ID team enters the alternative development phase of the project.

Spatial analysis issues and opportunities

During this phase of project completion, a few issues with the code were identified and the robustness of the tool could be improved in future iterations of the Python script. For starters, if incomplete harvest units do not fall completely within the cumulative effects analysis area, and completely outside the direct effects project area, there is potential for inconsistencies in actual acres being treated vs. acres that are being reported in the environmental impact statement. For example, if 10 acres of a 100-acre unit falls outside of the cumulative area, foresters would be apt to report that 100 acres are being treated, when only 90 acres would be accounted for in the analysis and the output summary tables from the geoprocessing tools. This can be mitigated by ensuring good data management practices when creating harvest units by ensuring that the units fall directly within the boundaries of the cumulative area. Moving forward, a topology tool script could be set for the map in the project that could validate the spatial relationship of the incomplete units leading to further warn the user when treatment areas are not within the correct project area. This issue is consistent across all the tools and care must be taken to ensure that forest management area inputs follow the methodology described in this report.

Furthermore, this project served as a good first step in spatial analysis, in that it easily quantifies the projected post-harvest impacts to forested stands on state trust lands. It does this by reclassifying key forest attributes based on expected results of forest management treatment. Because the tools themselves serve the purpose of making this analysis more efficient by creating simple to understand summary tables for use in environmental impact statements, more robust spatial analysis is not completed with this project. However, the outputs from the geoprocessing tools could be utilized to determine spatial relationships between attributes within the timber inventory, and further augment the decision-making process of future ID teams.

To do this, the categorical data of the updated alternative Alpha and Beta, or the data prep timber inventories could be exported into a csv file and read directly into a Jupyter notebook using the Pandas Python library and other libraries to identify areas of interest through means like hierarchical clustering. In this process, the summary table with attached polygon geometry could be exported using the current newly created geoprocessing tools. This file could then be read into a GeoPandas data frame, and the spatial relationship of the categorical data could be explored and conclusions drawn about where specific attributes exist on the landscape. This level of robustness in the analysis could lead

to further analysis where forested attributes can be clustered and the correlation between varying forest attributes can be discovered. This would lead to a better understanding of how forest management affects the landscape and lead to more informed decisions on the ground. Conceptually a question an ID team might want to understand is the location of adjacent age classes in relation to forest type, or old growth occurrence. By clustering these areas together and looking at where each of these attributes exist, and ID team might be able to predict where deficiencies exist in a certain age class or forest type and predict where they can be most successful at designing alternative treatments that maintain desirable attributes.

Finally, in order to make this project even more useful to ID team members, the following list are recommended improvements to the script and project that would increase the FAIR treatment of the data and make the vegetation analysis tool more useful in forest management project development.

- **Create a "complete run" version of the tool:** This tool would run every aspect of the vegetation analysis and allow the user to complete all steps with one button push leading to increased efficiency when completing the environmental analysis.
- **Add raw data output to each summary table:** This portion of the script would output the raw data from the manipulated timber inventory datasets into a final work sheet in the output Excel summary table. This would give the user the opportunity to run additional analysis to check summary statistics and develop new ways to investigate spatial relationships outside of the GIS environment.
- **Include automatic metadata for newly created feature classes:** This portion of the script would append metadata that would describe the analysis process on newly created feature classes. This would ensure that users would not need to remember to add in metadata at the end.
- **Include a topology check for newly created alternative units:** This portion would check for extreme examples where proposed treatment unit locations might lead to incorrect summary acres in the final environmental Impact statement report.
- **Design further ways to test results in a more controlled environment:** Units could be placed directly on the edges of a smaller hypothetical dataset that utilizes the same attributes and domains as state datasets. The acreages could then be calculated within the existing geoprocessing tools and results compared to a control dataset.

Conclusion

Environmental analysis for complex forest management projects takes time, and for a state agency like the Montana Department of Natural Resources and Conservation (DNRC), thorough and consistent completion of this analysis is key. This process is completed through the development of an interdisciplinary team (ID team), where foresters are usually tasked with completing the vegetation analysis for the project. In the past, analysis has been carried out over extended periods of time and personnel completing the analysis will sometimes change from the beginning of the project to the end. This can lead to inconsistencies within the analysis that requires increased review and quality control before environmental documents are released for public review purposes. This increased review can significantly increase the time spent completing analysis and the potential for inaccurate information to be produced is present. To comply with the analysis that is required by MEPA, this project attempted to streamline the vegetation analysis process by utilizing the ArcGIS Pro scripting tool and Python scripting to automatically complete a set of vegetation analysis. These scripts took user defined parameters such as a timber inventory layer, and proposed harvest units and combining them while changing key forest attribute fields to reflect potential post-harvest conditions.

The overall analysis required for completing this project was not overly complex, but time and consideration needed to be paid to each process and geoprocessing tool that was created. Some of the key concepts that ended up guiding this project were good project management, concise workflow development, and adequate documentation of resulting data tables and feature classes. By implementing these concepts, a quick and consistent way to complete vegetation analysis was developed. Each of the resulting geoprocessing tools behaved as expected and minimal input was required to complete adequate vegetation analysis. On average, it took roughly between 2 and 3 hours to complete the whole suite of vegetation analysis tools in one sitting. This timing was established when completing multiple testing runs during the results/discussion section of the project. This time to complete the analysis is far less than completing it by hand which was often completed over several days, and most of the time, in lengthy work sessions that spanned over multiple sittings. This project not only standardizes the analysis being completed by producing workflow (*Appendix C*) but allows foresters to complete multiple runs of analysis utilizing the geoprocessing tools. By quickly outputting the post-harvest effects of a treatment foresters can compare what the landscape might look like after a particular forest management project is completed. These hypothetical runs can be incorporated into an alternative development process at the ID team level which could lead to more informed decision

making on the ground as well as an increased realization of project objectives. In addition to making the vegetation analysis easier, the documentation of the vegetation analysis process and the resulting tables provided in this report leave an important and defensible line of reasoning as to how the analysis was completed and what the intended use of the products were for.

Though a good portion of the project did go according to plan, a few lessons were learned along the way that might make a project like this in the future better. Early in the planning process, it was identified that not every aspect of vegetation analysis that is sometimes utilized in an environmental impact statement could be covered by the newly created geoprocessing tools due to time constraints and the interaction of how silvicultural prescriptions might affect forest stand attributes. To mitigate these constraints, the most important forest attributes were chosen that had clear connections to pre and post treatment conditions. Difficulty also arose around producing summary tables that could easily be understood, and interpreting expected results from the geoprocessing tools was not always evident when first running the tools. Though a forester would be able to interpret the results with the produced workbook, there is room for improvement by incorporating table formatting into the code that could make it easier for non-forest professionals to instantly understand the outputs without formatting the output tables by hand. Though there is room for improvement, the intended audience of this report should be able to understand the results and future versions of the toolbox can address these issues.

In conclusion, this project's goal was to streamline the vegetation analysis for environmental impact statements that are completed for forest management projects on state trust lands. The project largely completed this task by developing a methodology for completing the analysis and implementing that methodology using geoprocessing tools and Python scripting. The expected results were achieved as described in the discussion section and though there is improvement to be made, this vegetation analysis toolbox can be recommended to an ID team to serve as a project development tool, data analysis tool, and quality control tool when compared to analysis completed by hand. Moving forward, the projects vegetation analysis geoprocessing toolbox needs to be vetted through the Montana DNRC's Forest Management Bureau to ensure that live data can be fed to through the tools and that the resulting summary tables match those of the expected results of analysis from current projects. Additionally, these scripts can be altered and modified to include specific analysis not covered in this report. These updates are expected and necessary to ensure that the scope of issues developed within EIS project planning are included in the analysis. There is still much room for improvement within the

current scope and scale of work when streamlining vegetation analysis, but this project has made headway on reducing time spent by foresters and ID team members to complete vegetation analysis.

References

Causton, D. R. (1988). *An introduction to vegetation analysis : principles, practice, and interpretation* [Book]. Allen & Unwin.

Esri. (2023a, February 20). *What is GIS?* Esri GIS Overview Website. https://www.esri.com/en-us/what-is-gis/overview

Esri. (2023b, April 30). *Python migration from 10.x to ArcGIS Pro*. Esri Python Migration Website. https://pro.arcgis.com/en/pro-app/latest/arcpy/get-started/Python-migration-for-arcgis-pro.htm

Green, P., Joy, J., Sirucek, D., Hann, W., Zack, A., & Naumann, B. (1992). Old-Growth Forest Types of the Northern Region. In *R-1 SES 4/92; USDA Forest Service, Northern Region, Missoula, MT.* (pp. 23–26). USDA Forest Service, Northern Region.

Kim, T. J. (1999). Metadata for geo-spatial data sharing: A comparative analysis. In *Ann Reg Sci* (Vol. 33).

Losensky, J. (1997). Historical Vegetation of Montana. In *Contract 970900, DNRC Technical Report* (pp. 1–3). Montana DNRC.

National Academy of Sciences. (2019). *Reproducibility and Replicability in Science*. https://doi.org/10.17226/25303

Pfister, R., Kovalchik, B., Arno, S., & Presby, R. (1977). Forest Habitat Types of Montana. In *USDA For. Serv. Gen. Tech. Rep. INT-34. Intermountain Forest & Range Experiment Station, Ogden, Utah 84401* (p. 139). USDA Forest Service.

Pimpler, E. (2015). *Programming ArcGIS with Python cookbook : over 85 hands-on recipes to teach you how to automate your ArcGIS for desktop geoprocessing tasks using Python* (2nd ed.) [Book]. Packt Publishing.

Rees, B. E. Van. (2014). Python Scripting and GIS Increasing Efficiency. *Geoinformatics*, *17*(7), 46–48.

Ricker, B. A., Rickles, P. R., Fagg, G. A., & Haklay, M. E. (2020). Tool, toolmaker, and scientist: case study experiences using GIS in interdisciplinary research. *Cartography and Geographic Information Science*, *47*(4), 350–366. https://doi.org/10.1080/15230406.2020.1748113

Saabith, A. L. S., Fareez, MMM., & Vinothraj, T. (2019). Python Current Trend Applications- An Overview. *Scientific Journal of Impact Factor*, *6*(10), 6–6.

Sonti, SH. (2015). Application of Geographic Information System (GIS) in Forest Management. *Journal of Geography and Natural Disasters*, *5*(3), 2–5. doi:10.4172/2167- 0587.1000145

Standovár, T., Szmorad, F., Kovács, B., Kelemen, K., Plattner, M., Roth, T., & Pataki, Z. (2016). A novel forest state assessment methodology to support conservation and forest management planning. *Community Ecology*, *17*(2), 167–177. https://doi.org/10.1556/168.2016.17.2.5

Stockwell, H. (2013). *A Guide to the MONTANA ENVIRONMENTAL POLICY ACT*. http://leg.mt.gov/eqc

Understanding Trust Lands. (2023, February 20). State of Montana Public Information Website. https://dnrc.mt.gov/TrustLand/about/understanding-trust-land

Wiles, J. (2017). Montana's State School Trust Lands. In *Public Land and Resources Law Review* (Vol. 38).

Wilkinson, M. D. (2016). The FAIR Guiding Principles for scientific data management and stewardship. *Scientific Data*, *3*(160018). doi: 10.1038/sdata.2016.18

Data citations

Timber Inventory

FMB_DNRC_TimberSLI_20201029 (2023) [Download]. Montana Department of Natural Resources and Conservation, 2705 Spurgin Rd, Missoula MT. 59804 [October 29, 2020]

SRSF Boundary

CumulativeEffects_Boundary_20230221 (2023) [User Generated]. User generated feature class to be utilized as the cumulative effects boundary area. [February 21, 2023]

SRSF Boundary

Direct_Effects_Boundary_20230221 (2023) [User Generated]. User generated feature class to be utilized as the direct effects boundary area. [February 21, 2023]

Alternative Alpha Forest Management Treatments

PTU_AltA_20230221 (2023) [User Generated]. User generated feature class to be utilized as the theoretical management unit alternative. [February 21, 2023]

Alternative Beta Forest Management Treatments

PTU_AltB_20230221 (2023) [User Generated]. User generated feature class to be utilized as the theoretical management unit alternative. [February 21, 2023]

State of Montana

States (2020) [downloaded file]. Price, Maribeth. Mastering ArcGIS Pro, McGraw Hill Education, New York, NY [November 21, 2020].

State of Montana Ownership

States (2023) [downloaded file]. Obtained from the State KDE internal drives.

Appendix A: Data dictionary reference

Appendix A serves as a repository of data that will be included as an input or is expected as an output from geoprocessing tools in this report. Sources of the data can be found in the data citation section of the report. Field alias/common name, field name, and attribute field descriptions are included in the table. In order to keep dataset names consistent in the beginning stages of the project, the following naming convention was utilized when running the data prep geoprocessing tool and when downloading datasets from their source.

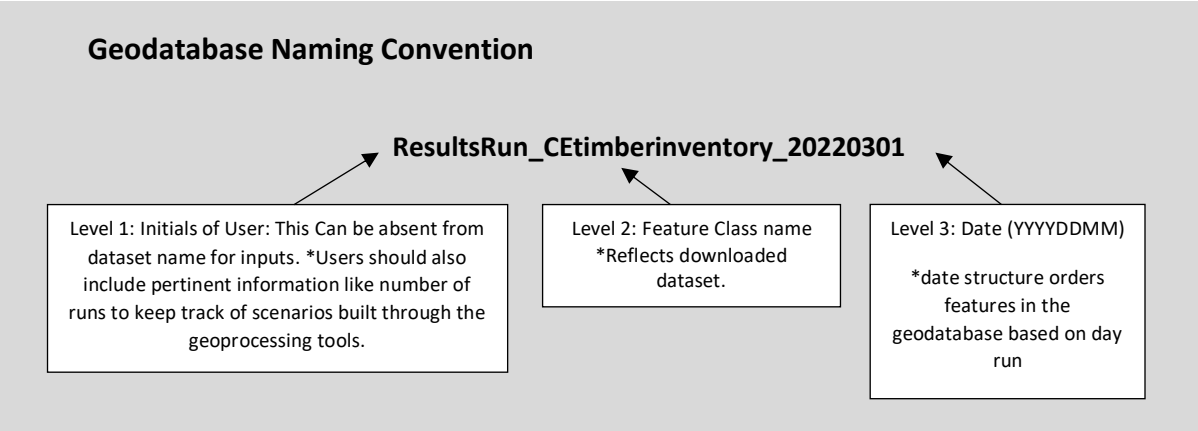


Figure 25: Shows the naming convention for input and output datasets used as parameters for vegetation analysis geoprocessing tools. Naming convetion is not mandatory for further analysis after this project.

Reference table 1: Data dictionary table[17]

Dataset Alias/ Common Name	***Dataset Name***	***Input/Output***	***Notes/Attributes***
Timber Inventory	Timberinventory_20230103	Input	Timber inventory consists of roughly 170 fields of forest attribute data that describes the current condition of individual polygons or stands. This project utilizes the following attribute fields: 1. FOGI CLASS 2. VIGORINDEX 3. STRUCINDEX 4. SNAGSINDEX 5. CWDINDEX 6. STKINDEX

[17] Reference table 1 is limited to datasets relevant to Python scripting only. Visual background layers like earth imagery and ownership used in maps within the document are not included.

Dataset Alias/ Common Name	*Dataset Name*	*Input/Output*	*Notes/Attributes*
			7. CROWNINDEX 8. TOTSTK 9. SAWSTK 10. SSC 11. TPA 12. AGECLASS 13. HAB_GRP 14. MAJPOTVEG 15. CVR_CURR 16. GIS_Acres
SRSF Boundary	CumulativeEffects_Boundary_20230221	Input	SRSF boundary consists of 1 polygon with two key attribute fields. Lands within the polygon boundaries include State owned lands and adjacent, non-state ownership. This project utilizes the following attribute fields: 1. Acres 2. Unit
Project Area	DirectEffects_Boundary_20230221	Input	Project area consists of 1 polygon with two key attribute fields. Lands within the polygon boundaries include land that covers haul routes and direct effects areas for forest management. This project utilizes the following attribute fields: 1. Acres 2. Project_Area_Name
Alternative Alpha Forest Management Treatments	PTU_AltA_20230221	Input	Alternative Alpha consists of 10 unique polygons within the project area detailing theoretical forest management units. The feature class contains 3 attribute fields that indicate stand location, proposed silvicultural treatment and other pertinent aspects of forest management activities.

Dataset Alias/ Common Name	*Dataset Name*	*Input/Output*	*Notes/Attributes*
			The project utilizes the following attribute fields: 1. Rx 2. Cutting_Unit 3. Defect_Risk
Alternative Beta Forest Management Treatments	PTU_AltB_20230221	Input	Alternative Beta consists of 10 unique polygons within the project area detailing theoretical forest management units. These units represent forest management units that might be recommended by foresters on the ID team. Their potential effects on the ground will alter forest attributes. The feature class contains 3 attribute fields that indicate stand location, proposed silvicultural treatment and other pertinent aspects of forest management activities. The project utilizes the following attribute fields: 1. Rx 2. Cutting_Unit 3. Defect_Risk
Incomplete Management Units	Incomplete_Units_20230221	Input	Incomplete Management Units consists of 5 unique polygons within the project area, detailing theoretical forest management units. These units represent forest management projects that would be “under” contract, but are currently not finished. Effects of these types of units must be considered as “Current Condition” before vegetation analysis can be completed. The feature class contains 2 attribute fields that indicate stand location, silvicultural treatment and other pertinent aspects of forest management activities.

Dataset Alias/ Common Name	***Dataset Name***	***Input/Output***	***Notes/Attributes***
Cumulative effects Timber Inventory	ResultsRun_CEtimberinventory_20230221	Output	Feature class output from the vegetation analysis data prep tool. The feature class contains the same attribute fields as the timber inventory dataset and is clipped to the cumulative effects project area. This can be utilized as an input for current stand condition summary table tools.
Updated Cumulative Effects Timber Inventory	ResultsRun_UpdatedCETimberInventory_20230221	Output	Feature class output from the vegetation analysis data prep tool. The feature class contains the same attributes fields as the timber inventory dataset *and* the incomplete management unit's data set. This can be utilized as an input for current stand conditions summary table tools.
Updated Direct Effects Timber Inventory	ResultsRun_UpdatedDETimberinventory_20230221	Output	Feature class output from the vegetation analysis data prep tool. The feature class contains the same attributes fields as the timber inventory dataset *and* the incomplete management unit's dataset. This can be utilized as an input for current stand conditions summary table tools.
	Current_OG_SRSF	Output	This feature class shows the user all current old growth stands within the cumulative effects project area. *End users might alter this output to fit the thematic needs of the maps they are creating.
	Removed_OG_AltA	Output	This feature class shows all areas of old growth that will be removed with alternative Alpha treatments.
	Removed_OG_AltB	Output	This feature class shows all areas of old growth that will be removed with alternative Beta treatments.

Dataset Alias/ Common Name	*Dataset Name*	*Input/Output*	*Notes/Attributes*
	Remaining_OG_AltA	Output	This feature class shows all areas of old growth remaining after implementing alternative Alpha treatments.
	Remaining_OG_AltB	Output	This feature class shows all areas of old growth remaining after implementation of Alternative Beta treatments.

Appendix B: Timber inventory attribute fields dictionary

Appendix A serves as a more detailed look into the attributes of the input datasets described in Appendix A. Timber inventory attributes were pulled from the Montana DNRC's SLI_Datadictionary accessed on 12/20/2022. Other attributes are either calculated in the ArcGIS Pro environment or created by foresters in the field reconnaissance portion of project planning.

Reference table 2: Attribute dictionary table

Dataset Alias/ Common Name and Source	***Notes/Attributes***
Timber Inventory DNRC Data Dictionary	FOGI CLASS – Full Old Growth Index * Calculated by adding VIGORINDEX + STRUCINDEX + SNAGSINDEX + CWDINDEX + LLTR_INDEX + STK_INDEX + CROWNINDEX VIGORINDEX – Record of vigor in the stand: 0 = Full vigor; open grown trees, 1 = Good to average vigor; clumpy grown trees 2 = Just below average to poor vigor; poor crown ratios 3 = Very poor vigor; stand is in decadent condition STRUCINDEX – Record of structure in the stand: 0 = single storied 1 = two-storied 2 = multi-storied SNAGSINDEX – Measure of amounts of snags in the stand 0 = No Snags 1 = Few snags (1-2 snags /acre) 2 = Some snags (3-10 snags/acre) 3 = Lots of snags (>= 11 snags/acre) CWDINDEX – Measure of amounts of Coarse Woody Debris in the stand. Calculated by number of pieces counted. 0 = No CWD (<1 ton/acre) 1 = Few CWD (1-9 tons/acre) 2 = Some CWD (10-20 tons/acre) 3 = Lots of CWD (>= 21 tons/acre)

Dataset Alias/ Common Name and Source	*Notes/Attributes*
	STKINDEX – Measure of gross stocking in thousand board feet in the stand. 0 = GMBFS < 4 1 = GMBFS >= 4 and GMBFS < 7 2 = GMBFS >= 7 and GMBFS < 10 3 = GMBFS >= 10 and GMBFS < 13 4 = GMBFS >= 13 and GMBFS < 16 5 = GMBFS >= 16 and GMBFS < 21 6 = GMBFS >= 21 and GMBFS < 26 7 = GMBFS >= 26 CROWNINDEX – Measure of crown density based on overall stocking of the stand 0 = Poorly stocked 10-39% of area 2 = Medium stocked 40-69 % of area 4 = Well stocked greater than 70% of area TOTSTK – Measure of total stocking in the stand of all size trees N = Non-stocked 0% of area S = Scattered stocking 0-9% of area P = Poorly stocked 10-39% of area M = Medium stocked 40-69% of area W = Well stocked greater than 70% of area SAWSTK – Measure of sawtimber total stocking in the stand N = Non-stocked 0% of area S = Scattered stocking 0-9% of area P = Poorly stocked 10-39% of area M = Medium stocked 40-69% of area W = Well stocked greater than 70% of area SSC – Forested Stand Size Class 6 = Non-stocked or deforested 7 = Seedling/sapling (> 50% of trees are less than 5” DBH) 8 = Pole timber (> 50% of trees are between 5” and 8.9” DBH) 9 = Saw timber (>10% crown density in trees ≥9”DBH or 11”DBH for hardwood, 40” max) TPA – Average number of trees per acre recorded to the nearest 50 TPA

Dataset Alias/ Common Name and Source	*Notes/Attributes*
	AGECLASS – Average age of the stand NONFOREST = Non-forest, road or water 000-039 = 0 to 39 years at model run 040-099 = 40 to 99 years at model run 100-149 = 100 to 149 years at model run 150-199 = 150 to 199 years at model run 200+ = 200+ years No Age Data= Those stands with INVSAM=12 and no age data OLDGROWTH = Those stands designated as old growth HAB_GRP – Habitat Type Group of the stand 2 = Warm and dry 3 = Cold 4 = Moderately warm and dry 5 = Moderately cool and dry 6 = Warm and moist 7 = Cool and moist 8 = Wet 9 = Moderately cool and moist 10 = Cool and moderately dry 11 = Cold and moderately dry MAJPOTVEG – Majority desired future condition cover within the stand. Based on Montana Administrative rules and the Losensky coding system. DF = Douglas-fir HW = Hardwood LP = Lodgepole pine MC = Mixed conifer NONCOMM = Noncommercial NONSTKD = Non-stocked PF-NC = Noncommercial Limber pine PP = Ponderosa pine SUBALP = Subalpine fir WL/DF = Western larch / Douglas-fir WWP = Western white pine CVR_CURR – Current cover within the stand. Based on Lozensky coding system. DF = Douglas-fir HW = Hardwood LP = Lodgepole pine MC = Mixed conifer NONCOMM = Noncommercial NONSTKD = Non-stocked

Dataset Alias/ Common Name and Source	*Notes/Attributes*
	PF-NC = Noncommercial limber pine PP = Ponderosa pine SUBALP = Subalpine fir WL/DF = Western larch / Douglas-fir WWP = Western white pine GIS_Acres – Acres calculated using ArcGIS Pro. Represents gross acres.
SRSF Boundary	This project utilizes the following attribute fields: Acres – Acres calculated using ArcGIS Pro. Unit – User defined text field: This project utilizes “Swan River State Forest”.
Project Area	This project utilizes the following attribute fields: Acres – Acres calculated using ArcGIS Pro Project_Area_Name – User defined text field: This project utilizes “StreamlineVeg”
Alternative Alpha Forest Management Treatments	The project utilizes the following attribute fields: Defect Risk – Measure of overall insect pressure in stand. Measured in the field using SRSF insect and disease risk rating system. High = Risk rating above 13 Medium = Risk rating 8-12 Low = Risk rating 4-7 Rx – Designated Silvicultural Prescription for the stand ST = Seed tree OSR = Overstory removal CC = Clearcut CT = Commercial thin GS = Group select ITS = Individual tree select OGM = Old growth maintenance OGR = Old growth recruitment SW = Shelterwood Cutting_Unit – User designated unit identifier: This project utilizes A1-A10

Dataset Alias/ Common Name and Source	***Notes/Attributes***
Alternative Beta Forest Management Treatments	The project utilizes the following attribute fields: Defect Risk – Measure of overall insect pressure in stand. Measured in the field using SRSF insect and disease risk rating system. High = Risk rating above 13 Medium = Risk rating 8-12 Low = Risk rating 4-7 Rx – Designated Silvicultural Prescription for the stand ST = Seed tree OSR = Overstory removal CC = Clearcut CT = Commercial thin GS = Group select ITS = Individual tree select OGM = Old growth maintenance OGR = Old growth recruitment SW = Shelterwood Cutting_Unit – User designated unit identifier: This project utilizes B1-B10
Incomplete Management Units	The project utilizes the following attribute fields: Rx – Designated silvicultural prescription for the stand ST = Seed tree OSR = Overstory removal CC = Clearcut CT = Commercial thin GS = Group select ITS = Individual tree select OGM = Old growth maintenance OGR = Old growth recruitment SW = Shelterwood Cutting_Unit – User designated unit identifier: This project utilizes A1-A10
Cumulative effects Timber Inventory	The feature class contains the same attribute fields as the timber inventory.
Updated Cumulative Effects Timber Inventory	The feature class contains the same attributes fields as the timber inventory dataset *and* the Incomplete management units dataset.

Dataset Alias/ Common Name and Source	***Notes/Attributes***
Updated Direct Effects Timber Inventory	The feature class contains the same attributes fields as the timber inventory dataset *and* the incomplete management unit's data set.

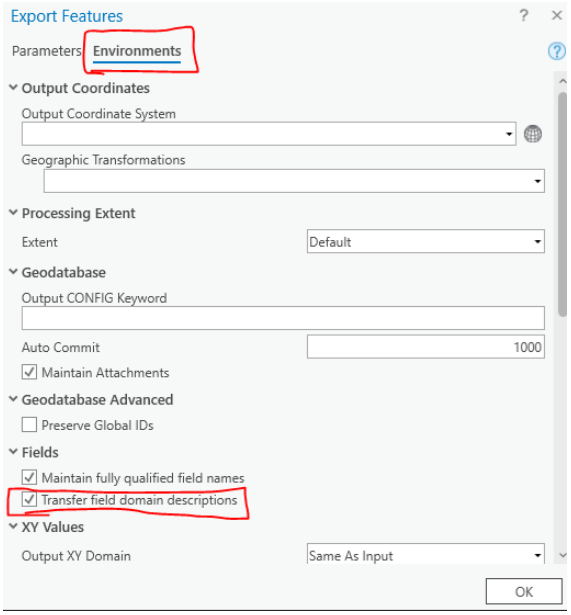
Appendix C: Vegetation analysis workbook reference

This appendix document is intended to be a record of updates that need to be made to state timber inventory data prior to use for analysis in an environmental impact statement as well as a basic workflow that can produce the required summary tables for post-harvest effects for alternative Alpha and Beta treatments. The following is a list of steps that an ID team member completing the data freeze portion of the vegetation analysis must follow to reflect potential changes to forest attributes based on intended effects of the previous multiple timber sale projects. The workflow is intended to show how analysists will complete the analysis process "by hand" and it is assumed that individuals utilizing this workbook reference have a working knowledge of the State's timber inventory layer, current file structure, and have read previous environmental impact statements to understand the summary tables needed for analysis. The above custom geoprocessing tools follow the same basic methods to complete the same processes more efficiently. Changes and processes completed in this workbook should be carried out on the best available data at the time and could change once these projects are completed. Each vegetation analysis process and summary table should be verified, and quality controlled with the state silviculturist to ensure that overall summary tables make sense. This serves as a baseline for analysis, and further analysis might be required based on comments and issues from the ID team or from the public.

Changing fields like age class and some of the index fields are not to be carried forward into future timber inventory releases because they are modeled fields. This data is intended to be used locally for analysis at the Swan Unit for the EIS.

Steps for Data Prep before Vegetation Analysis:

1. Pull the most up to date timber inventory layer for the whole state from the TLMD_Data folder on the K: drive.
    a. Ensure that this is the most up-to-date version of the SLI with Andy Moffett/Gina Mazza
    b. Save the **FMB_Timberinventory** layer to a local .gdb to be shared with ID team once updates are made. *This is intended to be a local copy and will be used for the rest of the project.



c. Clip the timber inventory layer to the **cumulative effects project area.** In the Swan this will be the SRSF boundary that includes internal adjacent landowners.

2. Organize previous EIS timber sales and determine which ones have been "closed out" and which ones are still "proposed".
    a. "Closed out sales"- Ensure that those updates are in the current version of the SLI that was pulled for this exercise.
    b. "Proposed" – Ensure that proposed units are in the ddit proposed harvest units' layer on the sale prep road log group.
        i. Check if layout has been completed and update sale boundary areas to reflect the most up-to-date shape and area, even if it has not sold! *This will generally be only for the previous EIS but if there is still projects lingering from two EIS's ago, make sure these potential changes are captured.
    c. "Open" – Ensure that open units are in the harvest history boundary areas to reflect the most up-to-date shape and area. *This will include any sales currently under contract.

3. Union the “Proposed” and “open” units from Edit proposed harvest units and harvest history from the previous projects to the **clipped timber inventory** layer from step 1. This file shows all of the unit polygons mixed with the Stand Level Inventory (SLI) with all attributes intact.

4. In order to only update the fields that will be affected by proposed timber sale harvest, apply a definition query to the union feature layer that shows only the SLI units from “proposed” and “open” timber sales. This will look like timber sale boundaries with SLI polygons splitting them in various arrangements.

5. Edit the following attribute fields based on the methods below. Ensure that you are editing the “union” Stand Level Inventory (SLI)

6. **FOGI indexes: Change the indexes that influence the FOGI score for all potentially treated OG in the Union layer.**
    - **a.** Query out the units that were age class old growth (These should be units that are only from the “proposed” and “Open” timber sales *These are stands that have been field verified).
    - **b.** **VIGORINDEX**- Vigor will not change post-harvest. Do not change.
    - **c.** **STRUCINDEX**- Change the structure index field to single story if the silviculture RX is Seed Tree or Overstory Removal. Change the index to more than two stories if Old Growth Maintenance is the silviculture prescription. OGM changed to greater than two stories.
        - **i. Change Condition: RX = ST or OSR, Change STRUCINDEX to “0”**
        - **ii. Change Condition: RX = OGM, Change STRUCINDEX to “2”**
        - **iii. *changing pre harvest index value from one to another needs to be a concerted effort. *Check Silv Form.**
    - **d.** **SNAGSINDEX -** Change snags index class to few for any treatment stands. Check OG handbook. Why: This will reflect our 4 trees per acre of snags and snag recruits we require for all our sales. Sale units generally do not have issues maintaining adequate snags unless noted in Recon or if they weren’t there to begin with.
        - **i. Change Condition: RX = ALL, Change SNAGSINDEX to “2”**

e. **CWDINDEX**– Change the coarse woody debris (CWD) index class to few for any treated stands. Why: this includes OGM and should reflect our post-harvest condition that is required by our contract.
    i. **Change condition: RX = ALL Rx, Change CWDINDEX to "2"**
f. **STKINDEX** – Change the stocking index class to a 3 (Around 2 loads/acre TOTAL) for OGM treatments and 1 (around a load / acre TOTAL) for even aged management treatments.) Other treatments fall in the 4-5 range for commercial thins etc. See OG maintenance handbook.
    i. **Change condition: RX = ST or OSR, change STKINDEX to "0"**
    ii. **Change condition: RX= SW or ITS, change STKINDEX to "1"**
    iii. **Change condition: RX = OGM or GS, change STKINDEX to "3"**
    iv. **Change condition: RX = all other treatments, change STKINDEX to "3-5" based on Silviculture form or recon sheet. *check Silviculture Form.**
g. **CROWNINDEX** – Change the crown index class to Medium for OGM, CT, and GS; and OSR and Seed tree to poor.
    i. **Change condition: RX = OGM or CT or GS, Change CRWNINDEX to "2"**
    ii. **Change condition: RX = ST or SW or OSR, Change CRWNINDEX to "0"**

***It would be neat to do some ground truthing for some of these stands to see if these changes hold up.**

7. **FOGI CLASS**- Recalculate the Full Old Growth Index field for treated OG stands to be the sum of the FOGI indexes. Classifications for the FOGI are as follows.
    i. **Change Condition:** FOGI = 'VIGORINDEX', 'STRUCINDEX', 'SNAGSINDEX', 'CWDINDEX', 'LLTRINDEX', 'STKINDEX', 'CROWNINDEX'
        1. Low < 13
        2. Med 13-20
        3. High 21+

8. **TOTSTK:** Change the TOTSTK fields for ST, OSR, and CC to a poor stocking and everything else to medium stocked. Why: This should reflect the at least 40% canopy cover in stands that are treated with commercial thin, OGM, group select, and other units.
    a. **Change condition: RX = ST, OSR or CC, Change TOTSTK to "P"**
    b. **Change Condition: RX = All else, Change TOTSTK to "M"**

9. **SAWSTK:** Change the SAWSTK fields for ST, OSR, and CC to a poor stocking and everything else to medium stocked. Why: This should reflect the at least 40% canopy cover in stands that are treated with commercial thin, OGM, group select, and other units.
    a. **Change condition: RX = ST or OSR or CC, change SAWSTK to "P"**
    b. **Change condition: RX = All else, change SAWSTK to "M"**

10. **SSC:** Change the SSC for OSR treatments to be 7. Why: What will actually be out there will be at least 4 trees per acre (TPA) of leave trees and the snags and snag recruits. Some areas will look like seed trees, but some areas will be more open than that. The understory will always be regenerated but to varying degrees.
    a. **Change condition: RX = OSR, change SSC to "7"**

11. **TPA:** change the TPA for OSR treatments to reflect what will be left on the ground. This is an estimate from the silviculture form/recon sheet in the stand.

    a. **Change condition: RX = OSR, change TPA to current estimate from the recon sheet.**

12. **AGECLASS: Change the age class field if the silviculture RX is either ST or OSR. Age class should be changed to 000-039. All other treatments will remain at their current Age Class.**

    a. **Change condition: RX = ST or OSR or CC, change AGECLASS to 000-039**

Vegetation Analysis Process

1. Organize current EIS timber sales into alternative Alpha and Beta units. (The easiest workflow would be to utilize the PTU server hosted feature layer from the Forest Management Bureau to build potential units.)
    a. "Alternative Alpha" – Ensure that proposed units are in the PTU units' layer on the sale prep road log group.
        i. Check if layout has been completed and update sale boundary areas to reflect the most up-to-date shape and area. This should come from the data freeze step in the planning process.
    b. "Alternative Beta" – Ensure that proposed units are in the PTU units' layer on the sale prep road log group.
        i. Check if layout has been completed and update sale boundary areas to reflect the most up-to-date shape and area. This should come from the data freeze step in the planning process.

2. Union the "alternative Alpha" and "alternative Beta" units from PTU to the **updated and clipped timber inventory** layer from the data freeze process. This file shows all the proposed alternative A and B unit polygons mixed with the SLI with all attributes intact. (You should have 4 distinct feature layers you are going to use as inputs to the following processes, A cumulative and direct effects updated SLI. Both show alternative A and alternative B treatments.)

3. To only update the fields that will be affected by proposed timber sale harvest, apply a definition query to the union feature layer that shows only the desired alternative units from "proposed" timber sales. This will look like timber sale boundaries with SLI polygons splitting them in various arrangements.

4. Edit the following attribute fields based on the methods below. Ensure that you are editing the appropriate alternatives "union".

5. Habitat group analysis – No change or definition query required on the current clipped and union SLI necessary.

a. Create a summary table using the summary statistics tool. Input table will be the desired alternatives union and clipped SLI feature layer. The output table will be designated in the file directory of your choice. **Statistics field = ACRES, Statistic Type = SUM, Case Field = CVR_CURR**. The output table should add up to your respective cumulative effects or direct effects area.

6. Desired future condition analysis – No change or definition query required on the current clipped and union SLI necessary.
    a. Create a summary table using the summary statistics tool. Input table will be the desired alternatives union and clipped SLI feature layer. The output table will be designated in file directory of your choice. **Statistics field = ACRES, Statistic Type = SUM, Case Field = MAJPOTVEG**. The output table should add up to your respective cumulative effects or direct effects area.
7. Current cover analysis – Add definition query to input clipped datasets from data prep step that only show either alternative A, AB, or B depending on the summary table you are building.
    a. Create a summary table using the summary statistics tool. Input table will be the desired alternatives union and clipped SLI feature layer. The output table will be designated in the file directory of your choice. **Statistics field = ACRES, Statistic Type = SUM, Case Field = CVR_CURR**. The output table should add up to your respective cumulative effects or direct effects area. This table is your current condition for current cover type.
    b. Add an attribute field called CVR_CURR_POST and calculate field to equal CVR_CURR
    c. **CVR_CURR_POST** – Change the current cover post index to match major potential vegetation if the silvicultural Rx is seed tree, group Select, shelterwood, or OSR treatment.
        i. **Change Condition: RX = ST, GS, SW, or OSR, Change CVR_CURR_POST == MAJPOTVEG**
    d. Create a summary table using the summary statistics tool. Input table will be the desired alternatives union and clipped SLI feature layer. The output table will be designated in the file directory of your choice. **Statistics field = ACRES, Statistic Type = SUM, Case Field = CVR_CURR**. The output table should add up to your respective cumulative effects or direct effects area. Compare The post-harvest table you created to the current cover condition table to describe total change in cover type.

e. Create a summary table using the summary statistics tool. Input table will be the desired alternatives union and clipped SLI feature layer. The output table will be designated in the file directory of your choice. **Statistics field = ACRES, Statistic Type = SUM, Case Field = CVR_CURR and CVR_CURR_POST**. The output table should add up to your respective cumulative effects or direct effects area. This table will tell you which acres changed from CVR_CURR to CVR_CURR_POST.

8. Age class analysis – Add definition query to input clipped datasets from data prep step that only show either alternative A, AB, or B depending on the summary table you are building.
    a. Create a summary table using the summary statistics tool. Input table will be the desired alternatives union and clipped SLI feature layer. The output table will be designated in the file directory of your choice. **Statistics field = ACRES, Statistic Type = SUM, Case Field = AGECLASS**. Output table acres should add up to your respective cumulative effects or direct effects area. This table is your current condition for current cover type.
    b. Add an attribute field Called AGECLASS_POST and calculate field to equal AGECLASS.
    c. **AGECLASS_POST** – Change the Age class post index to match 000-039 age class if the silvicultural Rx is seed tree, clearcut, or OSR treatment.
        i. **Change Condition: RX = ST, CC, or OSR, Change AGECLASS_POST == '000-039'**
    d. Create a summary table using the summary statistics tool. Input table will be the desired alternatives union and clipped SLI feature layer. The output table will be designated in the file directory of your choice. **Statistics field = ACRES, Statistic Type = SUM, Case Field = AGECLASS_POST**. The output table should add up to your respective cumulative effects or direct effects area. Compare The post-harvest table you created to the current age class table to describe total change in cover type.
    e. Create a summary table using the summary statistics tool. Input table will be the desired alternatives union and clipped SLI feature layer. The output table will be designated in the file directory of your choice. **Statistics field = ACRES, Statistic Type = SUM, Case Field = AGECLASS and AGECLASS_POST**. The output table should add up to your respective cumulative effects or direct effects area. This table will tell you which acres changed from AGECLASS to AGECLASS_POST.
9. Old growth analysis – Add definition query to input clipped datasets from data prep step that only show either alternative A, AB, or B AND to show AGECLASS = Old growth. This will show you

only the treated acres that are currently old growth and being treated with some sort of treatment.

a. Create a summary table using the summary statistics tool. Input table will be the desired alternatives union and clipped SLI feature layer. Output table will be designated in file directory of your choice. **Statistics field = ACRES, Statistic Type = SUM, Case Field = AGECLASS**. Output table acres should add up to the total acres of old growth in your respective cumulative effects or direct effects area. This table is your current condition for old growth cover type.
b. Add an attribute field called OLDGROWTH_POST and calculate field to equal AGECLASS.
c. **OLDGROWTH_POST** – Change the old growth post index to match 000-039 age class if the silvicultural Rx is seed tree, clearcut, or OSR treatment. Change the old growth post index to match '200+ years at model run- non old growth' if the silvicultural RX is individual tree select, old growth Recruitment, or group select treatment.
    i. **Change Condition: RX = ST, CC, or OSR, Change OLDGROWTH_POST == '000-039'**
    ii. **Change Condition: RX =ITS, OGR, OSR, Change OLDGROWTH_POST = '200+'**
d. Create a summary table using the summary statistics tool. Input table will be the desired alternatives union and clipped SLI feature layer. The output table will be designated in the file directory of your choice. **Statistics field = ACRES, Statistic Type = SUM, Case Field = OLDGROWTH_POST**. The output table should add up to your respective cumulative effects or direct effects area. Compare The post-harvest table you created to the current age class table to describe total change in cover type.
e. Create a summary table using the summary statistics tool. Input table will be the desired alternatives union and clipped SLI feature layer. The output table will be designated in the file directory of your choice. **Statistics field = ACRES, Statistic Type = SUM, Case Field = AGECLASS and OLDGROWTH_POST**. The output table should add up to your respective cumulative effects or direct effects area. This table will tell you which acres changed from AGECLASS to Oldgrowth_POST.
f. Create a summary table using the summary statistics tool. Input table will be the desired alternatives union and clipped SLI feature layer. The output table will be designated in the file directory of your choice. **Statistics field = ACRES, Statistic Type = SUM, Case Field = CVR_CURR and OLDGROWTH_POST**. The output table should add up to your

respective cumulative effects or direct effects area. This table will tell you which type of old growth acres changed within Oldgrowth_POST. *This step will show you non old growth acres.

g. Create a summary table using the summary statistics tool. Input table will be the desired alternatives union and clipped SLI feature layer. The output table will be designated in file directory of your choice. **Statistics field = ACRES, Statistic Type = SUM, Case Field = Defect_Risk**. The output table should add up to your respective cumulative effects or direct effects area. This table will tell you the number of high, medium, and low risk old growth you will have pre- and post-harvest. This table will add up to the total number of old growth acres being treated. The high medium and low values will be treated old growth.

10. Age and cover type patch size analysis- Add definition query to input clipped datasets from data prep step that only show either alternative A, AB, or B depending on the summary table you are building. *Cover type follows the same exact process but the CVR_CURR field is the key field used for the dissolve tool, and a corresponding CVR_CURR_POST field is required to complete the analysis below. Cover type is not written in steps below.
    a. Use the dissolve geoprocessing tool to dissolve the AGECLASS field. Input features = the queried feature class from step 10. Select appropriate output feature class and location. **Dissolve fields = AGECLASS, statistics field = ACRES, statistic type = SUM.** Output feature class will dissolve adjacent areas with like age class attributes and show all areas that have like age class attributes as one big multipart polygon.
    b. Use the multipart to single part geoprocessing tool to explode the age class field. Input features = the output from step 10a. Select appropriate output feature class and location. Output feature class will show all adjacent areas that have like age class attributes as single part features.
    c. Calculate acres for the newly created output feature class from step 10b.
    d. Create a summary table using the summary statistics tool. Input table will be the feature class from 10b. The output table will be designated in the file directory of your choice. **Statistics field = AGECLASS, Statistic Type = MEAN**. The output Table should show your current mean patch size for age class for your respective cumulative effects or direct effects area.
    e. Add an attribute field called AGECLASS_POST and calculate field to equal AGECLASS.

f. **AGECLASS_POST** – Change the age class post index to match 000-039 age class if the silvicultural Rx is seed tree, clearcut, or OSR treatment.

    i. **Change Condition: RX = ST, CC, or OSR, Change AGECLASS_POST == '000-039'**

g. Use the dissolve geoprocessing tool to dissolve the AGECLASS_POST field. Input features = the queried feature class from step 10. Select appropriate output feature class and location. **Dissolve fields = AGECLASS_POST, statistics field = ACRES, statistic type = SUM.** The output feature class will dissolve adjacent areas with like age class post attributes and show all areas that have like age class post attributes as one big multipart polygon.

h. Use the multipart to single part geoprocessing tool to explode the age class post field. Input features = the output from step 10g. Select appropriate output feature class and location. Output feature class will show all adjacent areas that have like age class attributes as single part features.

i. Calculate acres for the newly created output feature class from step 10h.

j. Create a summary table using the summary statistics tool. Input table will be the feature class from 10h. Output table will be designated in file directory of your choice. **Statistics field = AGECLASS_POST, Statistic Type = MEAN**. The output table should show your current mean patch size for age class post treatment for your respective cumulative effects or direct effects area.

Appendix D: Geoprocessing tools Python script reference

Appendix D serves as a repository of Python scripts that were created to complete specific processes required by vegetation analysis for an environmental impact statement. The scripts are shown as an highlighted text copied from and online syntax highlighting service found at (https://tohtml.com/). The scripting comments above or directly following the lines of executable code show how each line of code operates and will give the user an idea of which process each part of the script is completing. All scripts utilize the same column names as well as the domains from the state level PTU's. This removes the potential for the code to "break" and allows for quality control and assurance at the time of unit created in the environmental review process.

Geoprocessing tool Python script 1: vegetation analysis data prep

```
#Imports arcpy site package to grant user access to geoprocessing tools and
functions.
import arcpy

#allows for updating and overwriting of data in the user defined workspace.
arcpy.env.overwriteOutput = True


#sets variables for workspace and sets user inputs as variables and input
tables.these variables dont have to be utilized but make a more user friendly
and readable code block when running geoprocessing tools below.

DirectFC = arcpy.GetParameterAsText(0) #param0 gets user input to define the
Direct Effects area for use in later code blocks.

CumulativeFC = arcpy.GetParameterAsText(1) #param1 gets user input to define
the Cumulative Effects area for use in later code blocks.

SLIFC = arcpy.GetParameterAsText(2) #param2 gets user input to define the
Stand Level inventory. This should be downloaded from the most current SDE
available from the state K: Drive.

OpenTS = arcpy.GetParameterAsText(3) #param3 gets user input to define a
Current harvest units/ activities that have been proposed. this includes open
harvest sales where updates have not made it to the SLI.

ws = arcpy.GetParameterAsText(4) #param4 gets user input to define the output
location of where updated feature classes will be sent when the tool is
executed.

date = arcpy.GetParameterAsText(5) #param5 gets user input to define the date
when the tool is executed


#Uses for loop to iterate over characters in string from user input 5 (the
date selected for the tool run)
for d in range(len(date)):
    year = date[5:9] #sets variable for year based on position in date list.
expected result should be YYYY ex. (2023) for the year 2023
```

```
    day = date[0:1] #sets variable for day based on position in the date
list. expected result should be formatted as MM ex. (1) for January
    month = date[2:4] #sets variable for month based on position in the date
list. expected result should be formatted as DD ex. (12) for the 12th day of
the month

#creates variable for datesring that rearanges elements/characters from the
date list created above. The variable is formatted to follow the EIS naming
conventions.
datestring= (year+day+month)


#clips input from user parameters to clip timberinventory to the designated
cumulative effects area.
CEtimberinventory = arcpy.analysis.Clip(SLIFC, CumulativeFC, ws +
'_CEtimberinventory_'+datestring, '')

#processes the input from CEtimberinventory to delete unneeded fields. this
makes it easier for user to define analysis and also makes the dataset more
manageable. POTENTIALLY NOT THE BEST WAY TO DO THIS.
arcpy.management.DeleteField(CEtimberinventory,
['HAB_GRP','MAJPOTVEG','CVR_CURR','AGECLASS','OG_STATUS','VIGORINDEX','STRUCI
NDEX','SNAGSINDEX','CWDINDEX','STKINDEX','CROWNINDEX','FOGI','TOTSTK','SAWSTK
','SSC','TPA','ACRES','LLTRINDEX'], 'KEEP_FIELDS')

#Unions the current clipped timber inventory with the incomplete timber
sales. This output will become the new SLI inputs for further geoprocessing
tools.
UnionCE = arcpy.analysis.Union([CEtimberinventory,OpenTS], ws +
'_UpdatedCETimberInventory_'+datestring, 'ALL', '', '')

#The following Block of code changes the STRUCINDEX field to meet the
criteria described in the Data freeze Methodology
StrucFLA = arcpy.management.MakeFeatureLayer(UnionCE,'1',where_clause="Rx IN
('Seed Tree', 'OSR')") #Makes temporary feature layer from the UnionCE that
just has the stands that show where Structure index would be affected by Seed
tree or OSR treatment.

arcpy.management.CalculateField(StrucFLA, 'STRUCINDEX', '0' , 'PYTHON3', '',
'', 'ENFORCE_DOMAINS') #Calculates the new field for StrucFLA to be 0 to
represent 1 canopy level for being treated with Seed Tree or OSR RX. See
methodology for reasoning.

StrcFLOGM = arcpy.management.MakeFeatureLayer(UnionCE,'2',where_clause="Rx IN
('OGM')") #Makes temporary feature layer from the UnionCE that just has the
stands that show where Snags index would be affected by Seed tree or OSR
treatment.

arcpy.management.CalculateField(StrcFLOGM,
'STRUCINDEX','2','PYTHON3','','','ENFORCE_DOMAINS') #Calculates the new field
for StrucFLA to be 2 to represent 3 or more canopy level for being treated
with old growth maintenance RX. See methodology for reasoning.

#the following Block of code changes the SNAGSINDEX field to meet the
criteria described in the Data freeze methodology
SnagFLA = arcpy.management.MakeFeatureLayer(UnionCE,'3',where_clause = "Rx
NOT IN ('')")#Makes temporary feature layer from unionCE that shows snags
```

```
index would be affected by all treatments. Anything that is treated will have
snags post treatment.

arcpy.management.CalculateField(SnagFLA, 'SNAGSINDEX',
'2','PYTHON3','','','ENFORCE_DOMAINS')#Calculates the new field for
SnagsIndex to be 2 to represent "some snags" in each of the treated stands.

#the following Block of code changes the CWDINDEX field to meet the criteria
described in the Data freeze methodology
CWDFLA = arcpy.management.MakeFeatureLayer(UnionCE,'4',where_clause = "Rx NOT
IN ('')")#Makes temporary feature layer from unionCE that shows coarse woody
debris index would be affected by all treatments.

#the following Block of code changes the STKINDEX field to meet the criteria
described in the Data freeze methodology
STKFLA = arcpy.management.MakeFeatureLayer(UnionCE,'5',where_clause ="Rx IN
('Seed Tree', 'OSR')")#Makes temporary feature layer from unionCE that shows
stocking index would be affected by all treatments.

arcpy.management.CalculateField(STKFLA, 'STKINDEX',
'0','PYTHON3','','','ENFORCE_DOMAINS')#treatments with ST and OSR will have
'0' or less than one load/acre stocking post treatment

STKFLB = arcpy.management.MakeFeatureLayer(UnionCE,'6',where_clause ="Rx IN
('Shelterwood', 'ITS')")#Makes temporary feature layer from unionCE that
shows stocking index would be affected by all treatments.

arcpy.management.CalculateField(STKFLB, 'STKINDEX',
'1','PYTHON3','','','ENFORCE_DOMAINS')#treatments with Shelterwood or ITS
will have around 1 load/acre stocking post treatment

STKFLC = arcpy.management.MakeFeatureLayer(UnionCE,'7',where_clause ="RX IN
('OGM','Group Select')")#Makes temporary feature layer from unionCE that
shows stocking index would be affected by all treatments.

arcpy.management.CalculateField(STKFLC, 'STKINDEX', '3',
'PYTHON3','','','ENFORCE_DOMAINS')#treatments with OGM and GS will have
around 2 loads/acre stocking post treatment

#the following block of code changes the CROWNINDEX field to meet the
criteria described in the data freeze methodology.
CROWNFLA = arcpy.management.MakeFeatureLayer(UnionCE,'8',where_clause ="RX IN
('OGM','Commercial Thin','Group Select')")#Makes temporary feature layer from
unionCE that shows crown index would be affected by all treatments.

arcpy.management.CalculateField(CROWNFLA, 'CROWNINDEX', '2',
'PYTHON3','','','ENFORCE_DOMAINS') #anything treated with above treatments
will have code '2' or medium stocking post treatment.

CROWNFLB = arcpy.management.MakeFeatureLayer(UnionCE,'9',where_clause ="RX IN
('Seed Tree','Shelterwood','OSR')")#Makes temporary feature layer from
unionCE that shows crown index would be affected by all treatments. Anything
that is treated will have snags post treatment.

arcpy.management.CalculateField(CROWNFLB, 'CROWNINDEX', '0',
'PYTHON3','','','ENFORCE_DOMAINS')#anything treated with above treatments
will have code '0' or low stocking post treatment.
```

```
#the following block of code changes the FOGI field to meet the criteria
described in the data freeze methodology.
FOGIFLA = arcpy.management.MakeFeatureLayer(UnionCE,'10',where_clause = "Rx
NOT IN ('')") #Makes temporary feature layer from unionCE that shows crown
index would be affected by all treatments.

arcpy.management.CalculateField(FOGIFLA, 'FOGI', '!VIGORINDEX! + !STRUCINDEX!
+ !SNAGSINDEX! + !CWDINDEX! + !LLTRINDEX! + !STKINDEX! +
!CROWNINDEX!','PYTHON3','','','ENFORCE_DOMAINS')#calculate field adds all of
the index class to show a final FOGI based on the above treatments.

#the following block of code changes the TOTSTK field to meet the criteria
described in the data freeze methodology.
TOTSTKFLA = arcpy.management.MakeFeatureLayer(UnionCE,'11',where_clause ="RX
IN ('Seed Tree','OSR','Clear Cut')") #Makes temporary feature layer from
unionCE that shows total stocking would be affected by all treatments.

arcpy.management.CalculateField(TOTSTKFLA, 'TOTSTK', str("'P'"),
'PYTHON3','','','')#anything in even aged management regime described above
is changed to poor total stocking post treatment.

TOTSTKFLB = arcpy.management.MakeFeatureLayer(UnionCE,'12',where_clause ="Rx
NOT IN ('','Seed Tree','OSR','Clear Cut')") #Makes temporary feature layer
from unionCE that shows how total stocking would be affected by all
treatments.

arcpy.management.CalculateField(TOTSTKFLB, 'TOTSTK',"'M'",
'PYTHON3','','','')#anything not in even aged management regime is changed to
medium total stocking post treatment.

#the following block of code changes the SAWSTK field to meet the criteria
described in the data freeze methodology.
SAWSTKFLA = arcpy.management.MakeFeatureLayer(UnionCE,'13',where_clause ="RX
IN ('Seed Tree','OSR','Clear Cut')")#Makes temporary feature layer from
unionCE that shows saw stocking would be affected by all treatments.

arcpy.management.CalculateField(SAWSTKFLA, 'SAWSTK',"'P'",
'PYTHON3','','','')#anything under an even aged management regime would be
changed to poor saw stocking post treatment.

SAWSTKFLB = arcpy.management.MakeFeatureLayer(UnionCE,'14',where_clause ="Rx
NOT IN ('','Seed Tree','OSR','Clear Cut')")#Makes temporary feature layer
from unionCE that shows saw stocking index would be affected by all
treatments.

arcpy.management.CalculateField(SAWSTKFLB, 'SAWSTK', "'M'",
'PYTHON3','','','') #anything not in even aged management regime is changed
to medium saw stocking post treatment.

#the following block of code changes the SSC field to meet the criteria
described in the data freeze methodology.
SSCFLA = arcpy.management.MakeFeatureLayer(UnionCE,'15',where_clause ="RX IN
('OSR')")#Makes temporary feature layer from unionCE that shows how forest
stand size class would be affected by all treatments.
```

```
arcpy.management.CalculateField(SSCFLA, 'SSC', '7',
'PYTHON3','','','ENFORCE_DOMAINS')#anything treated with overstory removal
treatment would be considered seedling sapling or ssc= '7'

#the following block of code changes the TPA field to meet the criteria
described in the data freeze methodology.

TPAFLA = arcpy.management.MakeFeatureLayer(UnionCE,'16',where_clause ="RX IN
('OSR')")#Makes temporary feature layer from unionCE that shows how trees per
acre left would be affected by all treatments. Anything that is treated will
have snags post treatment.

arcpy.management.CalculateField(TPAFLA, 'TPA', '350',
'PYTHON3','','','ENFORCE_DOMAINS')#esentially this turns the tpa into 350 for
OSR treatments. This sets the precedent that we need at least 350 TPA before
considering an overstory removal treatment.


#the following block of code changes the AGECLASS field to meet the criteria
described in the data freeze methodology.
AGECLASSFLA = arcpy.management.MakeFeatureLayer(UnionCE,'17',where_clause
="RX IN ('Seed Tree','OSR','Clear Cut')")#Makes temporary feature layer from
unionCE that shows how ageclass would be affected by all treatments. Anything
that is treated will have snags post treatment.

arcpy.management.CalculateField(AGECLASSFLA, 'AGECLASS', "'000-039'",
'PYTHON3','','','')#anything in the even aged management regime would be
considered 'reset' and the new age class would be 000-039 years of age. other
treatments would retain all of their current age attributes.

UnionDE = arcpy.analysis.Clip(UnionCE,DirectFC, ws +
'_UpdatedDETimberInventory_'+datestring,'') #this clip just takes data from
the cumulative effects area and puts it into the direct effects area. USER
NOTE, PROJECT AREA WILL NEED TO DIRECTLY COINCIDE OR FALL WITHIN PROJECT AREA
TO BE ACCURATE. this goes last because the source dataset has been updated
over various lines of code above.

arcpy.management.DeleteField( UnionDE,'Rx') #deletes the rx attribute field
from the final unionde feature class because further timber sale/ treatment
units will be added to the sli later.
arcpy.management.DeleteField( UnionCE,'Rx') #deletes the rx attribute field
from the final unionde feature class because further timber sale/ treatment
units will be added to the sli later.
```

Geoprocessing tool Python script 2: Habitat type group analysis (current condition)

```
import arcpy

arcpy.env.overwriteOutput = True #Allows existing datasets in the workspace
environment to be overwritten.
ProjectAreaFC = arcpy.GetParameterAsText(0) #Gets user input to set the
Direct Effects area updated timber inventory.
CumulativeFC = arcpy.GetParameterAsText(1) #Gets user input to set the Direct
Effects area updated timber invnentory.
OutFile = arcpy.GetParameterAsText(2) #Gets user input to set the output file
location and name for the Direct Effects Project Area.

#Calculates area for polygons from user inputs.
arcpy.management.CalculateGeometryAttributes(ProjectAreaFC,[["ACRES","AREA"]]
,"","ACRES")#function parameters calculate acres field with acres.
arcpy.management.CalculateGeometryAttributes(CumulativeFC,[["ACRES","AREA"]],
"","ACRES")#function parameters calculate acres field with acres.

#summarizes acres of habitat type groups for project area and cumulative
effects area
intablePA =
arcpy.analysis.Statistics(ProjectAreaFC,"Current_Habitat_Type_DE",[["ACRES","
SUM"]],["HAB_GRP"]) #Creates variable that holds the non-spatial table for
Direct Effects project area. Parameters sum the acres from the acres field
based on the Habitat Type Group Field.
intableCU =
arcpy.analysis.Statistics(CumulativeFC,"Current_Habitat_Type_CE",[["ACRES","S
UM"]],["HAB_GRP"]) #Creates variable that holds the non-spatial table for
Cumulative Effects Project area. Parameters sum the acres from the acres
field based on the Habitat Type Group Field.

#Converts non spatial tables from above block of code into Excel files for ID
team use.
arcpy.conversion.TableToExcel([intablePA,intableCU],OutFile,'ALIAS','DESCRIPT
ION') #Parameters from function set "in" non spatial table, the output file
path/location, and (ALIAS, DESCRIPTION) export row headers and domain names
for field attributes into the final excel table.


arcpy.management.Delete(intablePA) #Declutters the home environment of the
non spatial summary tables once the excel files are created.
arcpy.management.Delete(intableCU)#Declutters the home environment of the non
spatial summary tables once the excel files are created.
```

Geoprocessing tool Python script 3: Desired future condition analysis (current condition)

```
import arcpy

arcpy.env.overwriteOutput = True #Allows existing datasets in the workspace
environment to be overwritten.
ProjectAreaFC = arcpy.GetParameterAsText(0) #Gets user input to set the
Direct Effects area updated timber inventory.
CumulativeFC = arcpy.GetParameterAsText(1) #Gets user input to set the Direct
Effects area updated timber invnentory.
OutFile = arcpy.GetParameterAsText(2) #Gets user input to set the output file
location and name for the Direct Effects Project Area.

#Calculates area for polygons from user inputs.
arcpy.management.CalculateGeometryAttributes(ProjectAreaFC,[["ACRES","AREA"]]
,"","ACRES")#function parameters calculate acres field with acres.
arcpy.management.CalculateGeometryAttributes(CumulativeFC,[["ACRES","AREA"]],
"","ACRES")#function parameters calculate acres field with acres.

#summarizes acres of habitat type groups for project area and cumulative
effects area
intablePA =
arcpy.analysis.Statistics(ProjectAreaFC,"Current_Preferred_Vegetation_DE",[["
ACRES","SUM"]],["MAJPOTVEG"]) #Creates variable that holds the non-spatial
table for Direct Effects project area. Parameters sum the acres from the
acres field based on the Habitat Type Group Field.
intableCU =
arcpy.analysis.Statistics(CumulativeFC,"Current_Preferred_Vegetation_CE",[["A
CRES","SUM"]],["MAJPOTVEG"]) #Creates variable that holds the non-spatial
table for Cumulative Effects Project area. Parameters sum the acres from the
acres field based on the Habitat Type Group Field.

#Converts non spatial tables from above block of code into Excel files for ID
team use.
arcpy.conversion.TableToExcel([intablePA,intableCU],OutFile,'ALIAS','DESCRIPT
ION') #Parameters from function set "in" non spatial table, the output file
path/location, and (ALIAS, DESCRIPTION) export row headers and domain names
for field attributes into the final excel table.


arcpy.management.Delete(intablePA) #Declutters the home environment of the
non spatial summary tables once the excel files are created.
arcpy.management.Delete(intableCU)#Declutters the home environment of the non
spatial summary tables once the excel files are created.
```

Geoprocessing tool Python script 4: Current cover analysis (current condition and post treatment condition)

```
import arcpy

#sets variables for workspace allows overwrite/output in workspace, sets user
inputs as variables and input tables.
arcpy.env.overwriteOutput = True #allows for overwrite output within the home
workspace.
arcpy.env.transferDomains = True #sets home environment to transfer domain
names and descriptions automatically. this creates a more user friendly
experience in the final output tables.
DEbase = arcpy.GetParameterAsText(0) #user input Direct effects area timber
inventory. Should be "updatedDEtimberinventory_DATE" output from vegetation
analysis Data Prep
CEbase = arcpy.GetParameterAsText(1)#user input Cumulative effects area
timber inventory. Should be "updatedCEtimberinventory_DATE" output from
vegetation analysis Data Prep
OutFile = arcpy.GetParameterAsText(2) #user input Excel file name for output
tables. All tables will be passed to this outfile
NewTreatmentA = arcpy.GetParameterAsText(3) #user defined Alternative A
treatment units.
NewTreatmentB = arcpy.GetParameterAsText(4) #user defined Alternative B
treatment Units.

ProjectAreaFCA =
arcpy.analysis.Union([DEbase,NewTreatmentA],'Updated_SLI_CoverType_DE_AltA')
#unions Alternative A treatment units to the updated DE timber inventory
CumulativeFCA =
arcpy.analysis.Union([CEbase,NewTreatmentA],'Updated_SLI_CoverType_CE_AltA')
#unions Alternative A treatment units to the updated CE timber inventory

ProjectAreaFCB =
arcpy.analysis.Union([DEbase,NewTreatmentB],'Updated_SLI_CoverType_DE_AltB')
#unions Alternative B treatment units to the update DE timber inventory
CumulativeFCB =
arcpy.analysis.Union([CEbase,NewTreatmentB],'Updated_SLI_CoverType_CE_AltB')
#unions Alterative B treatment units to the updated CE timber inventory

#Creates post treatment cover field and gives it a user friendly alias for
reporting in an excel table later.
arcpy.management.CalculateField(ProjectAreaFCA,
'CVR_CURR_POST','!CVR_CURR!','PYTHON3','','','ENFORCE_DOMAINS') #calculates
new field called CVR_CURR_POST to hold changes to current cover for DE
alternative A.
arcpy.management.CalculateField(CumulativeFCA,
'CVR_CURR_POST','!CVR_CURR!','PYTHON3','','','ENFORCE_DOMAINS') #Calculates
new field called CVR_CURR_POST to hold changes to the current cover for CE
alternative A.
arcpy.management.AlterField(ProjectAreaFCA,'CVR_CURR_POST',new_field_alias =
"Cover Type Post Treatment") #gives new field an easier to read alias.
arcpy.management.AlterField(CumulativeFCA,'CVR_CURR_POST',new_field_alias =
"Cover Type Post Treatment") #gives new field an easier to read alias.
```

```
#Creates post treatment cover field and gives it a user friendly alias for
reporting in an excel table later. #READ COMMENT FIELDS ABOVE, THIS PROCESS
DOES THE SAME THING FOR ALTERNATIVE B
arcpy.management.CalculateField(ProjectAreaFCB,
'CVR_CURR_POST','!CVR_CURR!','PYTHON3','','','ENFORCE_DOMAINS')#calculates
new field called CVR_CURR_POST to hold changes to current cover for DE
alternative B.
arcpy.management.CalculateField(CumulativeFCB,
'CVR_CURR_POST','!CVR_CURR!','PYTHON3','','','ENFORCE_DOMAINS')#Calculates
new field called CVR_CURR_POST to hold changes to the current cover for CE
alternative B.
arcpy.management.AlterField(ProjectAreaFCB,'CVR_CURR_POST',new_field_alias =
"Cover Type Post Treatment")#gives new field an easier to read alias.
arcpy.management.AlterField(CumulativeFCB,'CVR_CURR_POST',new_field_alias =
"Cover Type Post Treatment")#gives new field an easier to read alias.



#Calculates area for polygons from user inputs.
arcpy.management.CalculateGeometryAttributes(ProjectAreaFCA,[["ACRES","AREA"]
],"","ACRES")
arcpy.management.CalculateGeometryAttributes(CumulativeFCA,[["ACRES","AREA"]]
,"","ACRES")

#Calculates area for polygons from user inputs.
arcpy.management.CalculateGeometryAttributes(ProjectAreaFCB,[["ACRES","AREA"]
],"","ACRES")
arcpy.management.CalculateGeometryAttributes(CumulativeFCB,[["ACRES","AREA"]]
,"","ACRES")



#the following Block of code changes the CVR_CURR_POST field to meet the
criteria described in the Data freeze methodology for the direct effects area
CVRPOSTFLDEA =
arcpy.management.MakeFeatureLayer(ProjectAreaFCA,'1',where_clause ="Rx IN
('Seed Tree','Group Select','Shelterwood', 'OSR')") #uses temporary feature
layer to select affected treatments.
arcpy.management.CalculateField(CVRPOSTFLDEA, 'CVR_CURR_POST',
'!MAJPOTVEG!','PYTHON3','','','ENFORCE_DOMAINS') #changes current cover post
field to equal major potential vegetation based on criteria above.

#the following Block of code changes the CVR_CURR_POST field to meet the
criteria described in the Data freeze methodology for the direct effects area
CVRPOSTFLDEB =
arcpy.management.MakeFeatureLayer(ProjectAreaFCB,'2',where_clause ="Rx IN
('Seed Tree','Group Select','Shelterwood', 'OSR')")#uses temporary feature
layer to select affected treatments.
arcpy.management.CalculateField(CVRPOSTFLDEB, 'CVR_CURR_POST',
'!MAJPOTVEG!','PYTHON3','','','ENFORCE_DOMAINS') #changes current cover post
field to equal major potential vegetation based on criteria above.

#the following Block of code changes the CVR_CURR_POST field to meet the
criteria described in the Data freeze methodology
CVRPOSTFLCEA =
arcpy.management.MakeFeatureLayer(CumulativeFCA,'3',where_clause ="Rx IN
```

```
('Seed Tree','Group Select','Shelterwood', 'OSR')")#uses temporary feature
layer to select affected treatments.
arcpy.management.CalculateField(CVRPOSTFLCEA, 'CVR_CURR_POST',
'!MAJPOTVEG!','PYTHON3','','','ENFORCE_DOMAINS') #changes current cover post
field to equal major potential vegetation based on criteria above.


#the following Block of code changes the CVR_CURR_POST field to meet the
criteria described in the Data freeze methodology
CVRPOSTFLCEB =
arcpy.management.MakeFeatureLayer(CumulativeFCB,'4',where_clause ="Rx IN
('Seed Tree','Group Select','Shelterwood', 'OSR')")#uses temporary feature
layer to select affected treatments.
arcpy.management.CalculateField(CVRPOSTFLCEB, 'CVR_CURR_POST',
'!MAJPOTVEG!','PYTHON3','','','ENFORCE_DOMAINS') #changes current cover post
field to equal major potential vegetation based on criteria above.



#summarizes acres of Current Cover for project area and cumulative effects
area
CurrentCoverDEA =
arcpy.analysis.Statistics(ProjectAreaFCA,"Alt_A_Current_Cover_DE",[["ACRES","
SUM"]],["CVR_CURR"])
CurrentCoverCEA =
arcpy.analysis.Statistics(CumulativeFCA,"Alt_A_Current_Cover_CE",[["ACRES","S
UM"]],["CVR_CURR"])

#summarizes acres of Current Cover for project area and cumulative effects
area
CurrentCoverDEB =
arcpy.analysis.Statistics(ProjectAreaFCB,"Alt_B_Current_Cover_DE",[["ACRES","
SUM"]],["CVR_CURR"])
CurrentCoverCEB =
arcpy.analysis.Statistics(CumulativeFCB,"Alt_B_Current_Cover_CE",[["ACRES","S
UM"]],["CVR_CURR"])

#summarizes acres of Current cover post treatment for project area and
cumulative effects area
PostCoverDEA =
arcpy.analysis.Statistics(ProjectAreaFCA,'Alt_A_Post_Cover_DE',[["ACRES","SUM
"]],["CVR_CURR_POST"])
PostCoverCEA =
arcpy.analysis.Statistics(CumulativeFCA,'Alt_A_Post_Cover_CE',[["ACRES","SUM"
]],["CVR_CURR_POST"])

#summarizes acres of Current cover post treatment for project area and
cumulative effects area
PostCoverDEB =
arcpy.analysis.Statistics(ProjectAreaFCB,'Alt_B_Post_Cover_DE',[["ACRES","SUM
"]],["CVR_CURR_POST"])
PostCoverCEB =
arcpy.analysis.Statistics(CumulativeFCB,'Alt_B_Post_Cover_CE',[["ACRES","SUM"
]],["CVR_CURR_POST"])

#Summarizes acres of current cover post treatment and current cover so users
can quantify the change in acreage between old and new cover types.
```

```
PostCoverCHANGEDEA =
arcpy.analysis.Statistics(ProjectAreaFCA,'Alt_A_Post_Change_DE',[["ACRES","SU
M"]],["CVR_CURR","CVR_CURR_POST"])
PostCoverCHANGECEA =
arcpy.analysis.Statistics(CumulativeFCA,'Alt_A_Post_Change_CE',[["ACRES","SUM
"]],["CVR_CURR","CVR_CURR_POST"])

#Summarizes acres of current cover post treatment and current cover so users
can quantify the change in acreage between old and new cover types.
PostCoverCHANGEDEB =
arcpy.analysis.Statistics(ProjectAreaFCB,'Alt_B_Post_Change_DE',[["ACRES","SU
M"]],["CVR_CURR","CVR_CURR_POST"])
PostCoverCHANGECEB =
arcpy.analysis.Statistics(CumulativeFCB,'Alt_B_Post_Change_CE',[["ACRES","SUM
"]],["CVR_CURR","CVR_CURR_POST"])

#exports tables to excel
arcpy.conversion.TableToExcel([CurrentCoverDEA,PostCoverDEA,PostCoverCHANGEDE
A,CurrentCoverDEB,PostCoverDEB,PostCoverCHANGEDEB,CurrentCoverCEA,PostCoverCE
A,PostCoverCHANGECEA,CurrentCoverCEB,PostCoverCEB,PostCoverCHANGECEB],OutFile
,'ALIAS','DESCRIPTION') # this piece uses the table to excel function to
output the summary tables into the final resting place for the output excel
file. Each variable will show up as the workbook name in the excel file.



#following block deletes all summary tables that were used to export to
excels as well as deleting the project area and cumulative area feature
classes. These feature classes will not be used for mapping purposes in the
environmental impact statement.
arcpy.management.Delete(CurrentCoverDEA)
arcpy.management.Delete(CurrentCoverCEA)
arcpy.management.Delete(PostCoverDEA)
arcpy.management.Delete(PostCoverCEA)
arcpy.management.Delete(PostCoverCHANGEDEA)
arcpy.management.Delete(PostCoverCHANGECEA)
arcpy.management.Delete(CurrentCoverDEB)
arcpy.management.Delete(CurrentCoverCEB)
arcpy.management.Delete(PostCoverDEB)
arcpy.management.Delete(PostCoverCEB)
arcpy.management.Delete(PostCoverCHANGEDEB)
arcpy.management.Delete(PostCoverCHANGECEB)
arcpy.management.Delete([ProjectAreaFCA,CumulativeFCA,ProjectAreaFCB,Cumulati
veFCB])
```

Geoprocessing tool Python script 5: Forest age class analysis (current condition and post treatment condition)

```
import arcpy

#sets variables for workspace allows overwrite/output in workspace, sets user
inputs as variables and input tables.
arcpy.env.overwriteOutput = True
arcpy.env.transferDomains = True
DEbase = arcpy.GetParameterAsText(0) #gets the user input parameter of the
Direct Effects Area Updated timber Inventory, This serves as the Base layer
for further analysis in this script for Direct Effects (DE).
CEbase = arcpy.GetParameterAsText(1) #gets the user input parameter of the
Cumulative Effects Area Updated timber Inventory, This serves as the Base
layer for further analysis in this script for Cumulative Effects (DE).
OutFile = arcpy.GetParameterAsText(2) #gets the user input parameter of the
file path and name for the output excel files. All summary tables will be
written to this location later in the script.
NewTreatmentA = arcpy.GetParameterAsText(3) #gets the user input parameter of
Alternative A treatments.
NewTreatmentB = arcpy.GetParameterAsText(4) #gets the user input parameter of
Alternative B treatments.

ProjectAreaFCA =
arcpy.analysis.Union([DEbase,NewTreatmentA],'Updated_SLI_AgeClass_DE_AltA')
#unions the Alt A treatments to the input base inventory layer for the Direct
Effects Area
CumulativeFCA =
arcpy.analysis.Union([CEbase,NewTreatmentA],'Updated_SLI_AgeClass_CE_AltA')
#unions the Alt A treaments to the input base inventory layer for the
Cumulative Effects Area

ProjectAreaFCB =
arcpy.analysis.Union([DEbase,NewTreatmentB],'Updated_SLI_AgeClass_DE_AltB')
#unions the Alt B treatments to the input base inventory layer for the Direct
Effects Area
CumulativeFCB =
arcpy.analysis.Union([CEbase,NewTreatmentB],'Updated_SLI_AgeClass_CE_AltB')
#unions the Alt B treatments to the input base inventory layer for the
Cumulative Effects Area


#Creates post treatment cover field and gives it a user friendly alias for
reporting in an excel table later.
arcpy.management.CalculateField(ProjectAreaFCA,
'AGECLASS_POST','!AGECLASS!','PYTHON3','','','ENFORCE_DOMAINS')
arcpy.management.CalculateField(CumulativeFCA,
'AGECLASS_POST','!AGECLASS!','PYTHON3','','','ENFORCE_DOMAINS')
arcpy.management.AlterField(ProjectAreaFCA,'AGECLASS_POST',new_field_alias =
"Age Class Post Treatment")
arcpy.management.AlterField(CumulativeFCA,'AGECLASS_POST',new_field_alias =
"Age Class Post Treatment")

#Creates post treatment cover field and gives it a user friendly alias for
reporting in an excel table later.
arcpy.management.CalculateField(ProjectAreaFCB,
'AGECLASS_POST','!AGECLASS!','PYTHON3','','','ENFORCE_DOMAINS')
```

```
arcpy.management.CalculateField(CumulativeFCB,
'AGECLASS_POST','!AGECLASS!','PYTHON3','','','ENFORCE_DOMAINS')
arcpy.management.AlterField(ProjectAreaFCB,'AGECLASS_POST',new_field_alias =
"Age Class Post Treatment")
arcpy.management.AlterField(CumulativeFCB,'AGECLASS_POST',new_field_alias =
"Age Class Post Treatment")


#Calculates area for polygons from user inputs.
arcpy.management.CalculateGeometryAttributes(ProjectAreaFCA,[["ACRES","AREA"]
],"","ACRES")
arcpy.management.CalculateGeometryAttributes(CumulativeFCA,[["ACRES","AREA"]]
,"","ACRES")

#Calculates area for polygons from user inputs.
arcpy.management.CalculateGeometryAttributes(ProjectAreaFCB,[["ACRES","AREA"]
],"","ACRES")
arcpy.management.CalculateGeometryAttributes(CumulativeFCB,[["ACRES","AREA"]]
,"","ACRES")

#the following Block of code changes the AGECLASS_POST field to meet the
criteria described in the Data freeze methodology for the direct effects area
alternative A
AGEPOSTFLDEA =
arcpy.management.MakeFeatureLayer(ProjectAreaFCA,'1',where_clause ="Rx IN
('Seed Tree','Clearcut','OSR')") #areas with seed tree clearcut and OSR
treatments will be changed to 000-039 age classification
arcpy.management.CalculateField(AGEPOSTFLDEA, 'AGECLASS_POST',"'0 to 39 years
at model run'",'PYTHON3','','','')

#the following Block of code changes the AGECLASS_POST field to meet the
criteria described in the Data freeze methodology for the direct effects area
alternative B
AGEPOSTFLDEB =
arcpy.management.MakeFeatureLayer(ProjectAreaFCB,'2',where_clause ="Rx IN
('Seed Tree','Clearcut','OSR')")#areas with seed tree clearcut and OSR
treatments will be changed to 000-039 age classification
arcpy.management.CalculateField(AGEPOSTFLDEB, 'AGECLASS_POST',"'0 to 39 years
at model run'",'PYTHON3','','','')


#the following Block of code changes the AGECLASS_POST field to meet the
criteria described in the Data freeze methodology for the cumulative effects
area alternative A
AGEPOSTFLCEA =
arcpy.management.MakeFeatureLayer(CumulativeFCA,'3',where_clause ="Rx IN
('Seed Tree','Clearcut','OSR')")#areas with seed tree clearcut and OSR
treatments will be changed to 000-039 age classification
arcpy.management.CalculateField(AGEPOSTFLCEA, 'AGECLASS_POST',"'0 to 39 years
at model run'",'PYTHON3','','','')

#the following Block of code changes the AGECLASS_POST field to meet the
criteria described in the Data freeze methodology for the cumulative effects
area alternative B
AGEPOSTFLCEB =
arcpy.management.MakeFeatureLayer(CumulativeFCB,'4',where_clause ="Rx IN
```

```
('Seed Tree','Clearcut','OSR')")#areas with seed tree clearcut and OSR
treatments will be changed to 000-039 age classification
arcpy.management.CalculateField(AGEPOSTFLCEB, 'AGECLASS_POST',"'0 to 39 years
at model run'",'PYTHON3','','','')


#summarizes acres of age class for project area and cumulative effects area
these variables will then be passed to the table to excel function below.
These can be deleted post script.
CurrentAgeDEA =
arcpy.analysis.Statistics(ProjectAreaFCA,"ALTA_Current_Ageclass_DE",[["ACRES"
,"SUM"]],["AGECLASS"])
CurrentAgeCEA =
arcpy.analysis.Statistics(CumulativeFCA,"ALTA_Current_Ageclass_CE",[["ACRES",
"SUM"]],["AGECLASS"])

#summarizes acres of age class for project area and cumulative effects area
these variables will then be passed to the table to excel function below.
These can be deleted post script.
CurrentAgeDEB =
arcpy.analysis.Statistics(ProjectAreaFCB,"ALTB_Current_Ageclass_DE",[["ACRES"
,"SUM"]],["AGECLASS"])
CurrentAgeCEB =
arcpy.analysis.Statistics(CumulativeFCB,"ALTB_Current_Ageclass_CE",[["ACRES",
"SUM"]],["AGECLASS"])

#summarizes acres of age class post treatment for project area and cumulative
effects area these variables will then be passed to the table to excel
function below. These can be deleted post script.
PostAgeDEA =
arcpy.analysis.Statistics(ProjectAreaFCA,'ALTA_Post_Ageclass_DE',[["ACRES","S
UM"]],["AGECLASS_POST"])
PostAgeCEA =
arcpy.analysis.Statistics(CumulativeFCA,'ALTA_Post_Ageclass_CE',[["ACRES","SU
M"]],["AGECLASS_POST"])

#summarizes acres of age class post treatment for project area and cumulative
effects area these variables will then be passed to the table to excel
function below. These can be deleted post script.
PostAgeDEB =
arcpy.analysis.Statistics(ProjectAreaFCB,'ALTB_Post_Ageclass_DE',[["ACRES","S
UM"]],["AGECLASS_POST"])
PostAgeCEB =
arcpy.analysis.Statistics(CumulativeFCB,'ALTB_Post_Ageclass_CE',[["ACRES","SU
M"]],["AGECLASS_POST"])

#summarizes acres of age class post treatment and for age class current for
project area and cumulative effects area these variables will then be passed
to the table to excel function below. The resulting table will show the
change in acreage between pre existing condition and post harvest condition.
These can be deleted post script.
PostAgeCHANGEDEA =
arcpy.analysis.Statistics(ProjectAreaFCA,'ALTA_Post_Change_Ageclass_DE',[["AC
RES","SUM"]],["AGECLASS","AGECLASS_POST"])
PostAgeCHANGECEA =
arcpy.analysis.Statistics(CumulativeFCA,'ALTA_Post_Change_Ageclass_CE',[["ACR
ES","SUM"]],["AGECLASS","AGECLASS_POST"])
```

```
#summarizes acres of age class post treatment for project area and cumulative
effects area these variables will then be passed to the table to excel
function below. The resulting table will show the change in acreage between
pre existing condition and post harvest condition. These can be deleted post
script.
PostAgeCHANGEDEB =
arcpy.analysis.Statistics(ProjectAreaFCB,'ALTB_Post_Change_Ageclass_DE',[["AC
RES","SUM"]],["AGECLASS","AGECLASS_POST"])
PostAgeCHANGECEB =
arcpy.analysis.Statistics(CumulativeFCB,'ALTB_Post_Change_Ageclass_CE',[["ACR
ES","SUM"]],["AGECLASS","AGECLASS_POST"])

#exports tables to excel by using the table to excel function. All excel
tables will be output into individual workbooks within the outfile variable
set by the user. Alias and Description allow the domain names and
descriptions to be passed into the excel file so that it is easier to
understand.
arcpy.conversion.TableToExcel([CurrentAgeDEA,PostAgeDEA,PostAgeCHANGEDEA,Curr
entAgeDEB,PostAgeDEB,PostAgeCHANGEDEB,CurrentAgeCEA,PostAgeCEA,PostAgeCHANGEC
EA,CurrentAgeCEB,PostAgeCEB,PostAgeCHANGECEB],OutFile,'ALIAS','DESCRIPTION')

#uses the arcpy management delete function to get rid of all intermediate and
final feature classes and non spatial tables that are already exported to the
final excel outfile location.
arcpy.management.Delete(CurrentAgeDEA)
arcpy.management.Delete(CurrentAgeCEA)
arcpy.management.Delete(PostAgeDEA)
arcpy.management.Delete(PostAgeCEA)
arcpy.management.Delete(PostAgeCHANGEDEA)
arcpy.management.Delete(PostAgeCHANGECEA)
arcpy.management.Delete(CurrentAgeDEB)
arcpy.management.Delete(CurrentAgeCEB)
arcpy.management.Delete(PostAgeDEB)
arcpy.management.Delete(PostAgeCEB)
arcpy.management.Delete(PostAgeCHANGEDEB)
arcpy.management.Delete(PostAgeCHANGECEB)
arcpy.management.Delete(ProjectAreaFCA)
arcpy.management.Delete(CumulativeFCA)
arcpy.management.Delete(ProjectAreaFCB)
arcpy.management.Delete(CumulativeFCB)
```

Geoprocessing tool Python script 6: Forest old growth analysis (current condition and post treatment condition)

```
import arcpy

#sets variables for workspace allows overwrite/output in workspace, sets user
inputs as variables and input tables.
arcpy.env.overwriteOutput = True
arcpy.env.transferDomains = True
DEbase = arcpy.GetParameterAsText(0)#gets the user input parameter of the
Direct Effects Area Updated timber Inventory, This serves as the Base layer
for further analysis in this script for Direct Effects (DE).
CEbase = arcpy.GetParameterAsText(1)#gets the user input parameter of the
Cumulative Effects Area Updated timber Inventory, This serves as the Base
layer for further analysis in this script for Cumulative Effects (DE).
OutFile = arcpy.GetParameterAsText(2)#gets the user input parameter of the
file path and name for the output excel files. All summary tables will be
written to this location later in the script.
NewTreatmentA = arcpy.GetParameterAsText(3)#gets the user input parameter of
Alternative A treatments.
NewTreatmentB = arcpy.GetParameterAsText(4)#gets the user input parameter of
Alternative B treatments.

ProjectAreaFCA =
arcpy.analysis.Union([DEbase,NewTreatmentA],'Updated_SLI_Oldgrowth_DE_AltA')#
unions the Alt A treatments to the input base inventory layer for the Direct
Effects Area
CumulativeFCA =
arcpy.analysis.Union([CEbase,NewTreatmentA],'Updated_SLI_Oldgrowth_CE_AltA')#
unions the Alt A treaments to the input base inventory layer for the
Cumulative Effects Area

ProjectAreaFCB =
arcpy.analysis.Union([DEbase,NewTreatmentB],'Updated_SLI_Oldgrowth_DE_AltB')
#unions the Alt B treatments to the input base inventory layer for the Direct
Effects Area
CumulativeFCB =
arcpy.analysis.Union([CEbase,NewTreatmentB],'Updated_SLI_Oldgrowth_CE_AltB')#
unions the Alt B treatments to the input base inventory layer for the
Cumulative Effects Area


#Creates post treatment cover field and gives it a user friendly alias for
reporting in an excel table later.
arcpy.management.CalculateField(ProjectAreaFCA,
'OLDGROWTH_POST','!AGECLASS!','PYTHON3','','','ENFORCE_DOMAINS')
arcpy.management.CalculateField(CumulativeFCA,
'OLDGROWTH_POST','!AGECLASS!','PYTHON3','','','ENFORCE_DOMAINS')
arcpy.management.AlterField(ProjectAreaFCA,'OLDGROWTH_POST',new_field_alias =
"Old Growth Post Treatment")
arcpy.management.AlterField(CumulativeFCA,'OLDGROWTH_POST',new_field_alias =
"Old Growth Post Treatment")

#Creates post treatment cover field and gives it a user friendly alias for
reporting in an excel table later.
arcpy.management.CalculateField(ProjectAreaFCB,
'OLDGROWTH_POST','!AGECLASS!','PYTHON3','','','ENFORCE_DOMAINS')
```

```
arcpy.management.CalculateField(CumulativeFCB,
'OLDGROWTH_POST','!AGECLASS!','PYTHON3','','','ENFORCE_DOMAINS')
arcpy.management.AlterField(ProjectAreaFCB,'OLDGROWTH_POST',new_field_alias =
"Old Growth Post Treatment")
arcpy.management.AlterField(CumulativeFCB,'OLDGROWTH_POST',new_field_alias =
"Old Growth Post Treatment")


#Calculates area for polygons from user inputs.
arcpy.management.CalculateGeometryAttributes(ProjectAreaFCA,[["ACRES","AREA"]
],"","ACRES")
arcpy.management.CalculateGeometryAttributes(CumulativeFCA,[["ACRES","AREA"]]
,"","ACRES")

#Calculates area for polygons from user inputs.
arcpy.management.CalculateGeometryAttributes(ProjectAreaFCB,[["ACRES","AREA"]
],"","ACRES")
arcpy.management.CalculateGeometryAttributes(CumulativeFCB,[["ACRES","AREA"]]
,"","ACRES")



#the following Block of code makes custom feature layers that only show
ageclasses in Oldgrowth
OLDCURRENTFLDEA =
arcpy.management.MakeFeatureLayer(ProjectAreaFCA,'1',where_clause ="AGECLASS
IN ('OLD GROWTH')") #new feature layer will only show ageclass stands that
are Oldgrowth. This can be output to the final Geodatabase for map making
later.
OLDCURRENTFLCEA =
arcpy.management.MakeFeatureLayer(CumulativeFCA,'2',where_clause ="AGECLASS
IN ('OLD GROWTH')") #new feature layer will only show ageclass stands that
are Oldgrowth. This will be output to the final Geodatabase for map making
later.

#the following Block of code makes custom feature layers that only show
ageclasses in Oldgrowth
OLDCURRENTFLDEB =
arcpy.management.MakeFeatureLayer(ProjectAreaFCB,'3',where_clause ="AGECLASS
IN ('OLD GROWTH')")#new feature layer will only show ageclass stands that are
Oldgrowth. This can be output to the final Geodatabase for map making later.
OLDCURRENTFLCEB =
arcpy.management.MakeFeatureLayer(CumulativeFCB,'4',where_clause ="AGECLASS
IN ('OLD GROWTH')") #new feature layer will only show ageclass stands that
are Oldgrowth. This can be output to the final Geodatabase for map making
later.


#the following Block of code creates and updates an oldgrowth post treatment
field. Even aged management treatments will take oldgrowht down to 0 to 39
years of age, old growth maintenance will remain old growth, and uneven aged
management treatments will remain at old age, but not maintain oldgrowth
status.
RemoveOLDPOSTFLDEA =
arcpy.management.MakeFeatureLayer(OLDCURRENTFLDEA,'5',where_clause = "Rx IN
('Seed Tree','Clearcut','OSR')")#areas with seed tree clearcut and OSR
treatments will be changed to 000-039 age classification
```

```
arcpy.management.CalculateField(RemoveOLDPOSTFLDEA, 'OLDGROWTH_POST',"'0 to
39 years at model run'",'PYTHON3','','','')

MaintainOLDPOSTFLDEA =
arcpy.management.MakeFeatureLayer(OLDCURRENTFLDEA,'6',where_clause = "Rx IN
('OGM')")#areas with old growth maintenance (OGM) treatment will retain
oldgrowth status
arcpy.management.CalculateField(MaintainOLDPOSTFLDEA, 'OLDGROWTH_POST',"'Old
growth'",'PYTHON3','','','')

LightOLDPOSTFLDEA =
arcpy.management.MakeFeatureLayer(OLDCURRENTFLDEA,'7',where_clause = "Rx IN
('Individual Tree Select','OGR','Group Select')")#areas with Individual tree
Selection, old growth recruitment (OGR) and Group Select (GS) treatment will
retain old age status but not Oldgrowth status.
arcpy.management.CalculateField(LightOLDPOSTFLDEA, 'OLDGROWTH_POST',"'200+
years at model run - non oldgrowth'",'PYTHON3','','','')


#the following Block of code creates and updates an oldgrowth post treatment
field. Even aged management treatments will take oldgrowht down to 0 to 39
years of age, old growth maintenance will remain old growth, and uneven aged
management treatments will remain at old age, but not maintain oldgrowth
status.
RemoveOLDPOSTFLCEA =
arcpy.management.MakeFeatureLayer(OLDCURRENTFLCEA,'8',where_clause = "Rx IN
('Seed Tree','Clearcut','OSR')")#areas with seed tree clearcut and OSR
treatments will be changed to 000-039 age classification
arcpy.management.CalculateField(RemoveOLDPOSTFLCEA, 'OLDGROWTH_POST',"'0 to
39 years at model run'",'PYTHON3','','','')

MaintainOLDPOSTFLCEA =
arcpy.management.MakeFeatureLayer(OLDCURRENTFLCEA,'9',where_clause = "Rx IN
('OGM')")#areas with old growth maintenance (OGM) treatment will retain
oldgrowth status
arcpy.management.CalculateField(MaintainOLDPOSTFLDEA, 'OLDGROWTH_POST',"'Old
growth'",'PYTHON3','','','')

LightOLDPOSTFLCEA =
arcpy.management.MakeFeatureLayer(OLDCURRENTFLCEA,'10',where_clause = "Rx IN
('Individual Tree Select','OGR','Group Select')")#areas with Individual tree
Selection, old growth recruitment (OGR) and Group Select (GS) treatment will
retain old age status but not Oldgrowth status.
arcpy.management.CalculateField(LightOLDPOSTFLDEA, 'OLDGROWTH_POST',"'200+
years at model run - non oldgrowth'",'PYTHON3','','','')

RemainingOLDPOSTFLCEA =
arcpy.management.MakeFeatureLayer(OLDCURRENTFLCEA,'17',where_clause =
"OLDGROWTH_POST IN ('Old growth')")
RemainingOLDPOSTFLCEB =
arcpy.management.MakeFeatureLayer(OLDCURRENTFLCEB,'18',where_clause =
"OLDGROWTH_POST IN ('Old growth')")

#the following Block of code creates and updates an oldgrowth post treatment
field. Even aged management treatments will take oldgrowht down to 0 to 39
years of age, old growth maintenance will remain old growth, and uneven aged
```

```
management treatments will remain at old age, but not maintain oldgrowth
status.
RemoveOLDPOSTFLDEB =
arcpy.management.MakeFeatureLayer(OLDCURRENTFLDEB,'11',where_clause = "Rx IN
('Seed Tree','Clearcut','OSR')")#areas with seed tree clearcut and OSR
treatments will be changed to 000-039 age classification
arcpy.management.CalculateField(RemoveOLDPOSTFLDEB, 'OLDGROWTH_POST',"'0 to
39 years at model run'",'PYTHON3','','','')

MaintainOLDPOSTFLDEB =
arcpy.management.MakeFeatureLayer(OLDCURRENTFLDEB,'12',where_clause = "Rx IN
('OGM')")#areas with old growth maintenance (OGM) treatment will retain
oldgrowth status
arcpy.management.CalculateField(MaintainOLDPOSTFLDEB, 'OLDGROWTH_POST',"'Old
growth'",'PYTHON3','','','')

LightOLDPOSTFLDEB =
arcpy.management.MakeFeatureLayer(OLDCURRENTFLDEB,'13',where_clause = "Rx IN
('Individual Tree Select','OGR','Group Select')")#areas with Individual tree
Selection, old growth recruitment (OGR) and Group Select (GS) treatment will
retain old age status but not Oldgrowth status.
arcpy.management.CalculateField(LightOLDPOSTFLDEB, 'OLDGROWTH_POST',"'200+
years at model run - non oldgrowth'",'PYTHON3','','','')



#the following Block of code creates and updates an oldgrowth post treatment
field. Even aged management treatments will take oldgrowth down to 0 to 39
years of age, old growth maintenance will remain old growth, and uneven aged
management treatments will remain at old age, but not maintain oldgrowth
status.
RemoveOLDPOSTFLCEB =
arcpy.management.MakeFeatureLayer(OLDCURRENTFLCEB,'14',where_clause = "Rx IN
('Seed Tree','Clearcut','OSR')")#areas with seed tree clearcut and OSR
treatments will be changed to 000-039 age classification
arcpy.management.CalculateField(RemoveOLDPOSTFLCEB, 'OLDGROWTH_POST',"'0 to
39 years at model run'",'PYTHON3','','','')

MaintainOLDPOSTFLCEB =
arcpy.management.MakeFeatureLayer(OLDCURRENTFLCEB,'15',where_clause = "Rx IN
('OGM')")#areas with old growth maintenance (OGM) treatment will retain
oldgrowth status
arcpy.management.CalculateField(MaintainOLDPOSTFLDEB, 'OLDGROWTH_POST',"'Old
growth'",'PYTHON3','','','')

LightOLDPOSTFLCEB =
arcpy.management.MakeFeatureLayer(OLDCURRENTFLCEB,'16',where_clause = "Rx IN
('Individual Tree Select','OGR','Group Select')")#areas with Individual tree
Selection, old growth recruitment (OGR) and Group Select (GS) treatment will
retain old age status but not Oldgrowth status.
arcpy.management.CalculateField(LightOLDPOSTFLDEB, 'OLDGROWTH_POST',"'200+
years at model run - non oldgrowth'",'PYTHON3','','','')


#summarizes acres of Current Cover and age class of oldgrowth for ALT A
project area and cumulative effects area
```

```
CurrentOldgrowthDEA =
arcpy.analysis.Statistics(OLDCURRENTFLDEA,"Current_Oldgrowth_CVRtype_DE",[["A
CRES","SUM"]],["AGECLASS","CVR_CURR",])
CurrentOldgrowthCEA =
arcpy.analysis.Statistics(OLDCURRENTFLCEA,"Current_Oldgrowth_CVRtype_CE",[["A
CRES","SUM"]],["AGECLASS","CVR_CURR",])

#summarizes acres of Current cover and age class of oldgrowth for alt A post
treatment for project area and cumulative effects area
PostOldgrowthDEA =
arcpy.analysis.Statistics(OLDCURRENTFLDEA,'ALTA_Post_OG_CVRtype_DE',[["ACRES"
,"SUM"]],["OLDGROWTH_POST","CVR_CURR"])
PostOldgrowthCEA =
arcpy.analysis.Statistics(OLDCURRENTFLCEA,'ALTA_Post_OG_CVRtype_CE',[["ACRES"
,"SUM"]],["OLDGROWTH_POST","CVR_CURR"])

#summarizes acres of Current cover and ageclass for alt B post treatment for
project area and cumulative effects area
PostOldgrowthDEB =
arcpy.analysis.Statistics(OLDCURRENTFLDEB,'ALTB_Post_OG_CVRtype_DE',[["ACRES"
,"SUM"]],["OLDGROWTH_POST","CVR_CURR"])
PostOldgrowthCEB =
arcpy.analysis.Statistics(OLDCURRENTFLCEB,'ALTB_Post_OG_CVRtype_CE',[["ACRES"
,"SUM"]],["OLDGROWTH_POST","CVR_CURR"])

#summarizes acres of old growth and defect risk stands for project area and
cumulative effects area. This output table will show the number of high,
medium, and low risk stands that are being treated with treatment.
RiskOldgrowthDEA =
arcpy.analysis.Statistics(OLDCURRENTFLDEA,"ALTA_Post_OG_Risk_DE",[["ACRES","S
UM"]],["Defect_Risk"])
RiskOldgrowthCEA =
arcpy.analysis.Statistics(OLDCURRENTFLCEA,"ALTA_Post_OG_Risk_CE",[["ACRES","S
UM"]],["Defect_Risk"])

#summarizes acres of old growth and defect risk stands for project area and
cumulative effects area. This output table will show the number of high,
medium, and low risk stands that are being treated with treatment.
RiskOldgrowthDEB =
arcpy.analysis.Statistics(OLDCURRENTFLDEB,"ALTB_Post_OG_Risk_DE",[["ACRES","S
UM"]],["Defect_Risk"])
RiskOldgrowthCEB =
arcpy.analysis.Statistics(OLDCURRENTFLCEB,"ALTB_Post_OG_Risk_CE",[["ACRES","S
UM"]],["Defect_Risk"])



#exports tables to excel usning the table to excel function. Alias and
description keywords allow for the domain names and descriptions to be sent
to the final output excel workbooks. this makes it easier to read in the
final table.
arcpy.conversion.TableToExcel([CurrentOldgrowthDEA,CurrentOldgrowthCEA,PostOl
dgrowthDEA,PostOldgrowthDEB,PostOldgrowthCEA,PostOldgrowthCEB,RiskOldgrowthDE
A,RiskOldgrowthCEA,RiskOldgrowthDEB,RiskOldgrowthCEB],OutFile,'ALIAS','DESCRI
PTION')
```

```
arcpy.management.CopyFeatures('2','Current_OG_SRSF')#permenatly writes the
current old growth for the Cumulative effects area to the home geodatabase
set in the GIS environment.
arcpy.management.CopyFeatures('8','Removed_OG_AltA')#permenantly writes the
removed old growth for alt a and the Cumulative effects area to the home
goedatabase set in the GIS environment.
arcpy.management.CopyFeatures('17','Remaining_OG_AltA')#permenantly writes
the remaining Old growth post harvest for alt a and the cumulative effects
area to the home geodatabase set in the GIS environment.
arcpy.management.CopyFeatures('14','Removed_OG_AltB')#permenantly writes the
removed old growth post harvest for alt b and the cumulative effects area to
the home geodatabase set in the GIS environment.
arcpy.management.CopyFeatures('18','Remaining_OG_AltB')#permenantly writes
the remaining old growth post harvest for alt b and the cumulative effects
area to the home geodatabase set in the GIS environment.

# uses the arcpy management delete function to clean up the temporary and
final files that were not summarized and output into the final excel outfile
location.
arcpy.management.Delete(CurrentOldgrowthDEA)
arcpy.management.Delete(CurrentOldgrowthCEA)
arcpy.management.Delete(PostOldgrowthDEA)
arcpy.management.Delete(PostOldgrowthCEA)
arcpy.management.Delete(PostOldgrowthDEB)
arcpy.management.Delete(PostOldgrowthCEB)
arcpy.management.Delete(RiskOldgrowthDEA)
arcpy.management.Delete(RiskOldgrowthCEA)
arcpy.management.Delete(RiskOldgrowthDEB)
arcpy.management.Delete(RiskOldgrowthCEB)
arcpy.management.Delete(ProjectAreaFCA)
arcpy.management.Delete(CumulativeFCA)
arcpy.management.Delete(ProjectAreaFCB)
arcpy.management.Delete(CumulativeFCB)
```

Geoprocessing fool Python script 7: Age class patch size analysis

```
import arcpy

arcpy.env.overwriteOutput = True #Allows existing datasets in the workspace
environment to be overwritten.
arcpy.env.transferDomains = True
DEbase = arcpy.GetParameterAsText(0) #Gets user input to set the Direct
Effects area updated timber inventory.
CEbase = arcpy.GetParameterAsText(1) #Gets user input to set the Direct
Effects area updated timber invnentory.
OutFile = arcpy.GetParameterAsText(2) #Gets user input to set the output file
location and name for the Direct Effects Project Area.
NewTreatmentA = arcpy.GetParameterAsText(3) #user defined Alternative A
treatment units.
NewTreatmentB = arcpy.GetParameterAsText(4) #user defined Alternative B
treatment Units.

CurrentAgeclasssizeDE =
arcpy.management.Dissolve(DEbase,'Current_Cover_PatchDE',['AGECLASS'])#uses
the dissolve function to dissolve boundaries between polygons with the same
ageclass attributes. used for current condition age class patch size
calculation.

explodepatchDE =
arcpy.management.MultipartToSinglepart(CurrentAgeclasssizeDE,'Current_Ageclas
s_PatchSize_DE')#uses the multipart to single part function to take the
dissolved age classes and break them back apart so that islands will be
represented in the mean calculation. Used for current condition age class
patch size calculation.

CurrentAgeclasssizeCE =
arcpy.management.Dissolve(CEbase,'Current_Cover_PatchCE',['AGECLASS'])#uses
the dissolve function to dissolve boundaries between polygons with the same
ageclass attributes. used for current condition age class patch size
calculation.

explodepatchCE =
arcpy.management.MultipartToSinglepart(CurrentAgeclasssizeCE,'Current_Ageclas
s_PatchSize_CE') #uses the multipart to single part function to take the
dissolved age classes and break them back apart so that islands will be
represented in the mean calculation. Used for current condition age class
patch size calculation.


#Calculates area for polygons from user inputs.
arcpy.management.CalculateGeometryAttributes(explodepatchDE,[["ACRES","AREA"]
],"","ACRES")#function parameters calculate acres field with acres.
arcpy.management.CalculateGeometryAttributes(explodepatchCE,[["ACRES","AREA"]
],"","ACRES")#function parameters calculate acres field with acres.

#summarizes acres of habitat type groups for project area and cumulative
effects area
CurrentagepatchmeanDE =
arcpy.analysis.Statistics(explodepatchDE,"CurrentAgeclass_MeanPatch_DE",[["AC
RES","MEAN"]],["AGECLASS"]) #Creates variable that holds the non-spatial
```

```
table for Direct Effects project area. Parameters calculate the mean acres
from the acres field based on the exploded age class field.
CurrentagepatchmeanCE =
arcpy.analysis.Statistics(explodepatchCE,"CurrentAgeclass_MeanPatch_CE",[["AC
RES","MEAN"]],["AGECLASS"]) #Creates variable that holds the non-spatial
table for Cumulative Effects Project area. Parameters calculate the mean
acres from the acres field based on the exploded age class Field.

#creates union of current SLI and alternative a and B proposed treatment
units
ProjectAreaFCA =
arcpy.analysis.Union([DEbase,NewTreatmentA],'Updated_SLI_Ageclass_DE_AltA')
#unions Alternative A treatment units to the updated DE timber inventory
CumulativeFCA =
arcpy.analysis.Union([CEbase,NewTreatmentA],'Updated_SLI_Ageclass_CE_AltA')
#unions Alternative A treatment units to the updated CE timber inventory

ProjectAreaFCB =
arcpy.analysis.Union([DEbase,NewTreatmentB],'Updated_SLI_Ageclass_DE_AltB')
#unions Alternative B treatment units to the update DE timber inventory
CumulativeFCB =
arcpy.analysis.Union([CEbase,NewTreatmentB],'Updated_SLI_Ageclass_CE_AltB')
#unions Alterative B treatment units to the updated CE timber inventory

#Creates post treatment cover field and gives it a user friendly alias for
reporting in an excel table later.
arcpy.management.CalculateField(ProjectAreaFCA,
'AGECLASS_POST','!AGECLASS!','PYTHON3','','','ENFORCE_DOMAINS') #calculates
new field called AGECLASS_POST to hold changes to current cover for DE
alternative A.

arcpy.management.CalculateField(CumulativeFCA,
'AGECLASS_POST','!AGECLASS!','PYTHON3','','','ENFORCE_DOMAINS') #Calculates
new field called AGECLASS_POST to hold changes to the current cover for CE
alternative A.

arcpy.management.AlterField(ProjectAreaFCA,'AGECLASS_POST',new_field_alias =
"Age Class Post Treatment") #gives new field an easier to read alias.

arcpy.management.AlterField(CumulativeFCA,'AGECLASS_POST',new_field_alias =
"Age Class Post Treatment") #gives new field an easier to read alias.

#Creates post treatment cover field and gives it a user friendly alias for
reporting in an excel table later. #READ COMMENT FIELDS ABOVE, THIS PROCESS
DOES THE SAME THING FOR ALTERNATIVE B
arcpy.management.CalculateField(ProjectAreaFCB,
'AGECLASS_POST','!AGECLASS!','PYTHON3','','','ENFORCE_DOMAINS')#calculates
new field called AGECLASS_POST to hold changes to current cover for DE
alternative B.

arcpy.management.CalculateField(CumulativeFCB,
'AGECLASS_POST','!AGECLASS!','PYTHON3','','','ENFORCE_DOMAINS')#Calculates
new field called AGECLASS_POST to hold changes to the current cover for CE
alternative B.

arcpy.management.AlterField(ProjectAreaFCB,'AGECLASS_POST',new_field_alias =
"Age Class Post Treatment")#gives new field an easier to read alias.
```

```
arcpy.management.AlterField(CumulativeFCB,'AGECLASS_POST',new_field_alias =
"Age Class Post Treatment")#gives new field an easier to read alias.


#the following Block of code changes the CVR_CURR_POST field to meet the
criteria described in the Data freeze methodology for the direct effects area
AGEPOSTFLDEA =
arcpy.management.MakeFeatureLayer(ProjectAreaFCA,'1',where_clause ="Rx IN
('Seed Tree','Clearcut','OSR')")#uses the make feature layer function to
select treatments with Seed Tree, Clearcut, and OSR treatments.
arcpy.management.CalculateField(AGEPOSTFLDEA, 'AGECLASS_POST',"'0 to 39 years
at model run'",'PYTHON3','','','') #changes ageclass_post to 000-039 if the
Rx meets the criteria above.

#the following Block of code changes the CVR_CURR_POST field to meet the
criteria described in the Data freeze methodology for the direct effects area
AGEPOSTFLDEB =
arcpy.management.MakeFeatureLayer(ProjectAreaFCB,'1',where_clause ="Rx IN
('Seed Tree','Clearcut','OSR')")#uses the make feature layer function to
select treatments with Seed Tree, Clearcut, and OSR treatments.
arcpy.management.CalculateField(AGEPOSTFLDEB, 'AGECLASS_POST',"'0 to 39 years
at model run'",'PYTHON3','','','')#changes ageclass_post to 000-039 if the Rx
meets the criteria above.


#the following Block of code changes the CVR_CURR_POST field to meet the
criteria described in the Data freeze methodology
AGEPOSTFLCEA =
arcpy.management.MakeFeatureLayer(CumulativeFCA,'2',where_clause ="Rx IN
('Seed Tree','Clearcut','OSR')")#uses the make feature layer function to
select treatments with Seed Tree, Clearcut, and OSR treatments.
arcpy.management.CalculateField(AGEPOSTFLCEA, 'AGECLASS_POST',"'0 to 39 years
at model run'",'PYTHON3','','','')#changes ageclass_post to 000-039 if the Rx
meets the criteria above.

#the following Block of code changes the CVR_CURR_POST field to meet the
criteria described in the Data freeze methodology
AGEPOSTFLCEB =
arcpy.management.MakeFeatureLayer(CumulativeFCB,'2',where_clause ="Rx IN
('Seed Tree','Clearcut','OSR')")#uses the make feature layer function to
select treatments with Seed Tree, Clearcut, and OSR treatments.
arcpy.management.CalculateField(AGEPOSTFLCEB, 'AGECLASS_POST',"'0 to 39 years
at model run'",'PYTHON3','','','')#changes ageclass_post to 000-039 if the Rx
meets the criteria above.


PostagepatchsizeDEA =
arcpy.management.Dissolve(ProjectAreaFCA,'AltA_Post_Age_PatchDE',['AGECLASS_P
OST'])#uses the dissolve function to dissolve boundaries between polygons
with the same ageclass attributes. used for post condition age class post
patch size calculation.

PostageexplodepatchDEA =
arcpy.management.MultipartToSinglepart(PostagepatchsizeDEA,'AltA_Post_age_Pat
chSize_DE')#uses the multipart to single part function to take the dissolved
post age classes and break them back apart so that islands will be
```

```
represented in the mean calculation. Used for post condition age class patch
size calculation.

PostagepatchsizeCEA =
arcpy.management.Dissolve(CumulativeFCA,'AltA_Post_Age_PatchCE',['AGECLASS_PO
ST'])#uses the dissolve function to dissolve boundaries between polygons with
the same ageclass attributes. used for post condition age class post patch
size calculation.

PostageexplodepatchCEA =
arcpy.management.MultipartToSinglepart(PostagepatchsizeCEA,'AltA_Post_age_Pat
chSize')#uses the multipart to single part function to take the dissolved
post age classes and break them back apart so that islands will be
represented in the mean calculation. Used for post condition age class patch
size calculation.

PostagepatchsizeDEB =
arcpy.management.Dissolve(ProjectAreaFCB,'AltB_Post_Age_PatchDE',['AGECLASS_P
OST'])#uses the dissolve function to dissolve boundaries between polygons
with the same ageclass attributes. used for post condition age class post
patch size calculation.

PostageexplodepatchDEB =
arcpy.management.MultipartToSinglepart(PostagepatchsizeDEB,'AltB_Post_age_Pat
chSize_DE')#uses the multipart to single part function to take the dissolved
post age classes and break them back apart so that islands will be
represented in the mean calculation. Used for post condition age class patch
size calculation.

PostagepatchsizeCEB =
arcpy.management.Dissolve(CumulativeFCB,'AltB_Post_Age_PatchCE',['AGECLASS_PO
ST'])#uses the dissolve function to dissolve boundaries between polygons with
the same ageclass attributes. used for post condition age class post patch
size calculation.

PostageexplodepatchCEB =
arcpy.management.MultipartToSinglepart(PostagepatchsizeCEB,'AltB_Post_age_Pat
chSize')#uses the multipart to single part function to take the dissolved
post age classes and break them back apart so that islands will be
represented in the mean calculation. Used for post condition age class patch
size calculation.

#Calculates area for polygons from user inputs.
arcpy.management.CalculateGeometryAttributes(PostageexplodepatchDEA,[["ACRES"
,"AREA"]],"","ACRES")
arcpy.management.CalculateGeometryAttributes(PostageexplodepatchCEA,[["ACRES"
,"AREA"]],"","ACRES")

#Calculates area for polygons from user inputs.
arcpy.management.CalculateGeometryAttributes(PostageexplodepatchDEB,[["ACRES"
,"AREA"]],"","ACRES")
arcpy.management.CalculateGeometryAttributes(PostageexplodepatchCEB,[["ACRES"
,"AREA"]],"","ACRES")


#summarizes acres of habitat type groups for project area and cumulative
effects area
```

```
PostagepatchmeanDEA =
arcpy.analysis.Statistics(PostageexplodepatchDEA,"AltA_PostAgeclass_MeanPatch
_DE",[["ACRES","MEAN"]],["AGECLASS_POST"]) #Creates variable that holds the
non-spatial table for Direct Effects project area. Parameters calculate mean
acres from the acres field based on the ageclass_post Field.
PostagepatchmeanCEA =
arcpy.analysis.Statistics(PostageexplodepatchCEA,"AltA_PostAgeclass_MeanPatch
_CE",[["ACRES","MEAN"]],["AGECLASS_POST"]) #Creates variable that holds the
non-spatial table for Cumulative Effects Project area. Parameters calculate
mean acres from the acres field based on the ageclass_post Field.

#summarizes acres of habitat type groups for project area and cumulative
effects area
PostagepatchmeanDEB =
arcpy.analysis.Statistics(PostageexplodepatchDEB,"AltB_PostAgeclass_MeanPatch
_DE",[["ACRES","MEAN"]],["AGECLASS_POST"]) #Creates variable that holds the
non-spatial table for Direct Effects project area. Parameters sum the acres
from the acres field based on the Habitat Type Group Field.
PostagepatchmeanCEB =
arcpy.analysis.Statistics(PostageexplodepatchCEB,"ALTB_PostAgeclass_MeanPatch
_CE",[["ACRES","MEAN"]],["AGECLASS_POST"]) #Creates variable that holds the
non-spatial table for Cumulative Effects Project area. Parameters sum the
acres from the acres field based on the Habitat Type Group Field.


#Converts non spatial tables from above block of code into Excel files for ID
team use.
arcpy.conversion.TableToExcel([CurrentagepatchmeanDE,PostagepatchmeanDEA,Post
agepatchmeanDEB,CurrentagepatchmeanCE,PostagepatchmeanCEA,PostagepatchmeanCEB
],OutFile,'ALIAS','DESCRIPTION') #Parameters from function set "in" non
spatial table, the output file path/location, and (ALIAS, DESCRIPTION) export
row headers and domain names for field attributes into the final excel table.


#uses the arcpy delete function to get rid of non spatial datasets that were
exported to the output excel files. Current ageclass patch size and Post age
class patch size will be output to the workspace geodatabase.
arcpy.management.Delete([PostageexplodepatchDEA,PostageexplodepatchDEB,Projec
tAreaFCA,CumulativeFCA,ProjectAreaFCB,CumulativeFCB,explodepatchDE])
arcpy.management.Delete([CurrentAgeclasssizeDE,CurrentAgeclasssizeCE,Currenta
gepatchmeanDE,CurrentagepatchmeanCE,PostagepatchsizeDEA,PostagepatchsizeCEA,P
ostagepatchsizeDEB,PostagepatchsizeCEB,PostagepatchmeanDEA,PostagepatchmeanCE
A,PostagepatchmeanDEB,PostagepatchmeanCEB])
```

Geoprocessing tool Python script 8: Cover patch size analysis

```
import arcpy

arcpy.env.overwriteOutput = True #Allows existing datasets in the workspace
environment to be overwritten.
arcpy.env.transferDomains = True
DEbase = arcpy.GetParameterAsText(0) #Gets user input to set the Direct
Effects area updated timber inventory.
CEbase = arcpy.GetParameterAsText(1) #Gets user input to set the Direct
Effects area updated timber invnentory.
OutFile = arcpy.GetParameterAsText(2) #Gets user input to set the output file
location and name for the Direct Effects Project Area.
NewTreatmentA = arcpy.GetParameterAsText(3) #user defined Alternative A
treatment units.
NewTreatmentB = arcpy.GetParameterAsText(4) #user defined Alternative B
treatment Units.

CurrentpatchsizeDE =
arcpy.management.Dissolve(DEbase,'Current_Cover_PatchDE',['CVR_CURR']) #uses
the dissolve function to dissolve boundaries between covertypes that share
attribute values.

explodepatchDE =
arcpy.management.MultipartToSinglepart(CurrentpatchsizeDE,'Current_Cover_Patc
hSize_DE')#uses the multipart to single part features to identify islands so
that the true mean patch size can be calculated later.

CurrentpatchsizeCE =
arcpy.management.Dissolve(CEbase,'Current_Cover_PatchCE',['CVR_CURR'])#uses
the dissolve function to dissolve boundaries between covertypes that share
attribute values.

explodepatchCE =
arcpy.management.MultipartToSinglepart(CurrentpatchsizeCE,'Current_Cover_Patc
hSize_CE')#uses the multipart to single part features to identify islands so
that the true mean patch size can be calculated later.


#Calculates area for polygons from user inputs.
arcpy.management.CalculateGeometryAttributes(explodepatchDE,[["ACRES","AREA"]
],"","ACRES")#function parameters calculate acres field with acres.
arcpy.management.CalculateGeometryAttributes(explodepatchCE,[["ACRES","AREA"]
],"","ACRES")#function parameters calculate acres field with acres.

#summarizes acres of habitat type groups for project area and cumulative
effects area
CurrentcoverpatchmeanDE =
arcpy.analysis.Statistics(explodepatchDE,"CurrentCovertype_MeanPatch_DE",[["A
CRES","MEAN"]],["CVR_CURR"]) #Creates variable that holds the non-spatial
table for Direct Effects project area. Parameters sum the acres from the
acres field based on the Habitat Type Group Field.
CurrentcoverpatchmeanCE =
arcpy.analysis.Statistics(explodepatchCE,"CurrentCovertyp_MeanPatch_CE",[["AC
RES","MEAN"]],["CVR_CURR"]) #Creates variable that holds the non-spatial
table for Cumulative Effects Project area. Parameters sum the acres from the
acres field based on the Habitat Type Group Field.
```

```
#creates union of current SLI and alternative a and B proposed treatment
units
ProjectAreaFCA =
arcpy.analysis.Union([DEbase,NewTreatmentA],'Updated_SLI_CoverType_DE_AltA')
#unions Alternative A treatment units to the updated DE timber inventory
CumulativeFCA =
arcpy.analysis.Union([CEbase,NewTreatmentA],'Updated_SLI_CoverType_CE_AltA')
#unions Alternative A treatment units to the updated CE timber inventory

ProjectAreaFCB =
arcpy.analysis.Union([DEbase,NewTreatmentB],'Updated_SLI_CoverType_DE_AltB')
#unions Alternative B treatment units to the update DE timber inventory
CumulativeFCB =
arcpy.analysis.Union([CEbase,NewTreatmentB],'Updated_SLI_CoverType_CE_AltB')
#unions Alterative B treatment units to the updated CE timber inventory

#Creates post treatment cover field and gives it a user friendly alias for
reporting in an excel table later.
arcpy.management.CalculateField(ProjectAreaFCA,
'CVR_CURR_POST','!CVR_CURR!','PYTHON3','','','ENFORCE_DOMAINS') #calculates
new field called CVR_CURR_POST to hold changes to current cover for DE
alternative A.
arcpy.management.CalculateField(CumulativeFCA,
'CVR_CURR_POST','!CVR_CURR!','PYTHON3','','','ENFORCE_DOMAINS') #Calculates
new field called CVR_CURR_POST to hold changes to the current cover for CE
alternative A.
arcpy.management.AlterField(ProjectAreaFCA,'CVR_CURR_POST',new_field_alias =
"Cover Type Post Treatment") #gives new field an easier to read alias.
arcpy.management.AlterField(CumulativeFCA,'CVR_CURR_POST',new_field_alias =
"Cover Type Post Treatment") #gives new field an easier to read alias.

#Creates post treatment cover field and gives it a user friendly alias for
reporting in an excel table later. #READ COMMENT FIELDS ABOVE, THIS PROCESS
DOES THE SAME THING FOR ALTERNATIVE B
arcpy.management.CalculateField(ProjectAreaFCB,
'CVR_CURR_POST','!CVR_CURR!','PYTHON3','','','ENFORCE_DOMAINS')
arcpy.management.CalculateField(CumulativeFCB,
'CVR_CURR_POST','!CVR_CURR!','PYTHON3','','','ENFORCE_DOMAINS')
arcpy.management.AlterField(ProjectAreaFCB,'CVR_CURR_POST',new_field_alias =
"Cover Type Post Treatment")
arcpy.management.AlterField(CumulativeFCB,'CVR_CURR_POST',new_field_alias =
"Cover Type Post Treatment")



#the following Block of code changes the CVR_CURR_POST field to meet the
criteria described in the Data freeze methodology for the direct effects area
CVRPOSTFLDEA =
arcpy.management.MakeFeatureLayer(ProjectAreaFCA,'1',where_clause ="Rx IN
('Seed Tree','Group Select','Shelterwood', 'OSR')") #uses temporary feature
layer to select affected treatments.
arcpy.management.CalculateField(CVRPOSTFLDEA, 'CVR_CURR_POST',
'!MAJPOTVEG!','PYTHON3','','','ENFORCE_DOMAINS') #changes current cover post
field to equal major potential vegetation based on criteria above.
```

```
#the following Block of code changes the CVR_CURR_POST field to meet the
criteria described in the Data freeze methodology for the direct effects area
CVRPOSTFLDEB =
arcpy.management.MakeFeatureLayer(ProjectAreaFCB,'2',where_clause ="Rx IN
('Seed Tree','Group Select','Shelterwood', 'OSR')")#uses temporary feature
layer to select affected treatments.
arcpy.management.CalculateField(CVRPOSTFLDEB, 'CVR_CURR_POST',
'!MAJPOTVEG!','PYTHON3','','','ENFORCE_DOMAINS') #changes current cover post
field to equal major potential vegetation based on criteria above.

#the following Block of code changes the CVR_CURR_POST field to meet the
criteria described in the Data freeze methodology
CVRPOSTFLCEA =
arcpy.management.MakeFeatureLayer(CumulativeFCA,'3',where_clause ="Rx IN
('Seed Tree','Group Select','Shelterwood', 'OSR')")#uses temporary feature
layer to select affected treatments.
arcpy.management.CalculateField(CVRPOSTFLCEA, 'CVR_CURR_POST',
'!MAJPOTVEG!','PYTHON3','','','ENFORCE_DOMAINS') #changes current cover post
field to equal major potential vegetation based on criteria above.


#the following Block of code changes the CVR_CURR_POST field to meet the
criteria described in the Data freeze methodology
CVRPOSTFLCEB =
arcpy.management.MakeFeatureLayer(CumulativeFCB,'4',where_clause ="Rx IN
('Seed Tree','Group Select','Shelterwood', 'OSR')")#uses temporary feature
layer to select affected treatments.
arcpy.management.CalculateField(CVRPOSTFLCEB, 'CVR_CURR_POST',
'!MAJPOTVEG!','PYTHON3','','','ENFORCE_DOMAINS') #changes current cover post
field to equal major potential vegetation based on criteria above.


PostpatchsizeDEA =
arcpy.management.Dissolve(ProjectAreaFCA,'AltA_Post_Cover_PatchDE',['CVR_CURR
_POST'])#uses the dissolve function to dissolve boundaries between post
harvest covertypes that share attribute values.

PostexplodepatchDEA =
arcpy.management.MultipartToSinglepart(PostpatchsizeDEA,'AltA_Post_Cover_Patc
hSize_DE')#uses the multipart to single part features to identify islands so
that the true mean patch size can be calculated later.

PostpatchsizeCEA =
arcpy.management.Dissolve(CumulativeFCA,'AltA_Post_Cover_PatchCE',['CVR_CURR_
POST'])#uses the dissolve function to dissolve boundaries between post
harvest covertypes that share attribute values.

PostexplodepatchCEA =
arcpy.management.MultipartToSinglepart(PostpatchsizeCEA,'AltA_Post_Cover_Patc
hSize')#uses the multipart to single part features to identify islands so
that the true mean patch size can be calculated later.

PostpatchsizeDEB =
arcpy.management.Dissolve(ProjectAreaFCB,'AltB_Post_Cover_PatchDE',['CVR_CURR
_POST'])#uses the dissolve function to dissolve boundaries between post
harvest covertypes that share attribute values.
```

```
PostexplodepatchDEB =
arcpy.management.MultipartToSinglepart(PostpatchsizeDEB,'AltB_Post_Cover_Patc
hSize_DE')#uses the multipart to single part features to identify islands so
that the true mean patch size can be calculated later.

PostpatchsizeCEB =
arcpy.management.Dissolve(CumulativeFCB,'AltB_Post_Cover_PatchCE',['CVR_CURR_
POST'])#uses the dissolve function to dissolve boundaries between post
harvest covertypes that share attribute values.

PostexplodepatchCEB =
arcpy.management.MultipartToSinglepart(PostpatchsizeCEB,'AltB_Post_Cover_Patc
hSize')#uses the multipart to single part features to identify islands so
that the true mean patch size can be calculated later.

#Calculates area for polygons from user inputs.
arcpy.management.CalculateGeometryAttributes(PostexplodepatchDEA,[["ACRES","A
REA"]],"","ACRES")
arcpy.management.CalculateGeometryAttributes(PostexplodepatchCEA,[["ACRES","A
REA"]],"","ACRES")

#Calculates area for polygons from user inputs.
arcpy.management.CalculateGeometryAttributes(PostexplodepatchDEB,[["ACRES","A
REA"]],"","ACRES")
arcpy.management.CalculateGeometryAttributes(PostexplodepatchCEB,[["ACRES","A
REA"]],"","ACRES")


#summarizes acres of habitat type groups for project area and cumulative
effects area
PostcoverpatchmeanDEA =
arcpy.analysis.Statistics(PostexplodepatchDEA,"AltA_PostCovertype_MeanPatch_D
E",[["ACRES","MEAN"]],["CVR_CURR_POST"]) #Creates variable that holds the
non-spatial table for Direct Effects project area. Parameters sum the acres
from the acres field based on the Habitat Type Group Field.
PostcoverpatchmeanCEA =
arcpy.analysis.Statistics(PostexplodepatchCEA,"AltA_PostCovertype_MeanPatch_C
E",[["ACRES","MEAN"]],["CVR_CURR_POST"]) #Creates variable that holds the
non-spatial table for Cumulative Effects Project area. Parameters sum the
acres from the acres field based on the Habitat Type Group Field.

#summarizes acres of habitat type groups for project area and cumulative
effects area
PostcoverpatchmeanDEB =
arcpy.analysis.Statistics(PostexplodepatchDEB,"AltB_PostCovertype_MeanPatch_D
E",[["ACRES","MEAN"]],["CVR_CURR_POST"]) #Creates variable that holds the
non-spatial table for Direct Effects project area. Parameters sum the acres
from the acres field based on the Habitat Type Group Field.
PostcoverpatchmeanCEB =
arcpy.analysis.Statistics(PostexplodepatchCEB,"ALTB_PostCovertype_MeanPatch_C
E",[["ACRES","MEAN"]],["CVR_CURR_POST"]) #Creates variable that holds the
non-spatial table for Cumulative Effects Project area. Parameters sum the
acres from the acres field based on the Habitat Type Group Field.


#Converts non spatial tables from above block of code into Excel files for ID
team use.
```

```
arcpy.conversion.TableToExcel([CurrentcoverpatchmeanDE,PostcoverpatchmeanDEA,
PostcoverpatchmeanDEB,CurrentcoverpatchmeanCE,PostcoverpatchmeanCEA,Postcover
patchmeanCEB],OutFile,'ALIAS','DESCRIPTION') #Parameters from function set
"in" non spatial table, the output file path/location, and (ALIAS,
DESCRIPTION) export row headers and domain names for field attributes into
the final excel table.


#uses the delete function to clean up already output excel tables. Program
will not delete current and post cover patch size feature classes. They will
be used for mapmaking in the EIS.
arcpy.management.Delete([PostexplodepatchDEA,PostexplodepatchDEB])
arcpy.management.Delete([ProjectAreaFCA,CumulativeFCA,ProjectAreaFCB,Cumulati
veFCB,CurrentpatchsizeDE,CurrentpatchsizeCE,CurrentcoverpatchmeanDE,Currentco
verpatchmeanCE,PostpatchsizeDEA,PostpatchsizeCEA,PostpatchsizeDEB,Postpatchsi
zeCEB,PostcoverpatchmeanDEA,PostcoverpatchmeanCEA,PostcoverpatchmeanDEB,Postc
overpatchmeanCEB,explodepatchDE])
```