# NEW SHOOTS:
# A TREE LIST GENERATION DATABASE TUTORIAL

KEVIN R. GEHRINGER


STAND MANAGEMENT COOPERATIVE
SMC WORKING PAPER NUMBER 4
AUGUST 2001

COLLEGE OF FOREST RESOURCES
UNIVERSITY OF WASHINGTON
BOX 352100
SEATTLE, WASHINGTON 98115-2100

# TABLE OF CONTENTS

# LIST OF FIGURES

# LIST OF TABLES

## DISCLAIMER

THERE IS NO WARRANTY FOR THE SOFTWARE OR PROGRAMS PROVIDED, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE SOFTWARE OR PROGRAMS (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

# Chapter 1

# INTRODUCTION

This tutorial briefly describes how to use the tree list generation database developed for the Stand Management Cooperative (SMC). The tree list generation database was designed to generate simulated tree lists for Douglas-fir (*Pseudotsuga menziesii*) stands, western hemlock (*Tsuga heterophylla*) stands, and stands that are mixtures of these two species. Three stand types are supported by the tree list generation database for these two species: stands having at least 80% of their basal area represented by one of these two species will be referred to as pure stands for that species, stands having at least 50% of their basal area represented by one of these two species will be referred to as dominant stands for that species, and stand having less than 50% in either or both of these species will be referred to as mixed stands.

The tree list generation database uses actual tree measurements from forest stands spanning the Pacific Northwest from southern Oregon to southern British Columbia, west of the Cascade Mountains. The tree list generation process is broken down into two parts: a stand classification procedure based on a multidimensional histogram and a stand generation procedure based on a nearest neighbor criterion for selecting the trees used to generate a simulated stand.

A tree list generation database is available in two distributions: a standard distribution and a developer distribution. The only difference between the two distributions is the inclusion of the source code and object libraries in the developer distribution. The standard distribution should be all that is necessary for most users, unless cus-

tomized versions of the software are desired. DLLs are not provided at this time.

This tutorial is intended to supplement and augment the comprehensive documentation of the tree list generation database *Tree List Generation Database User's Guide and Reference Manual* that is provided with the tree list generation database distributions. The user's guide and reference manual should be consulted for file formats and other relevant details.

### *1.1 Conventions and notations*

The following typographic conventions are followed throughout the this document.

- Data file contents when presented in the text will appear in a teletype or fixed pitch font, e.g., `TREATMENT_TYPE = UNTREATED`.

- Species common names, e.g., Douglas-fir, will appear in the standard font, and species latin names, e.g., *Pseudotsuga menziesii* will appear in an italic font.

- References to file names or program names will be in a teletype or fixed pitch font, e.g., `input.dat` or `TGNEW`.

- References to interest addresses , email addresses, and uniform resource locators (URLs) will be in a teletype or fixed pitch font `http://www.website.com` or `emailhere.there.com`.

- In the examples, input typed by a user would will be in a teletype or fixed pitch font preceded by a prompt indicator, e.g., `PROMPT> dir *.dat`. Output from commands that would be displayed on the computer screen or shell will appear in an italic teletype or fixed pitch font, e.g., *`File successfully created`*.

- A forward slash, '/', is used as the path element separator throughout this document for consistency. When using the tree list generation database software,

either a forward slash, '/', or a backslash, '\', may be used in file names or directory paths to separate the individual path elements. The software will substitute the correct path separation character for the computing platform being used.

### *1.2 Obtaining the tree list generation database*

To obtain the tree list generation database, contact the Stand Management Cooperative in the College of Forest Resources at the University of Washington in Seattle. In order to download a tree list generation database distribution a logon ID and password are required. These must be obtained from the SMC prior to attempting a download of a tree list generation distribution. The tree list generation website may, however, be browsed without the logon ID and password. The SMC may be contacted using the addresses provided below.

Stand Management Cooperative
University of Washington
College of Forest Resources
Room 164 Bloedel Hall
Box 352100
Seattle, WA 98195-2100

email: `silvproj@u.washington.edu`

FAX: (206) 685-3091

SMC Home Page:
`http://www.cfr.washington.edu/smc/`

SMC Tree List Generation Database Home Page:
`http://depts.washington.edu/silvproj/tlghome/`

### *1.3 Installing the tree list generation database*

For the purposes of this tutorial, we assume that you have not already downloaded the standard distribution of the tree list generation database, so we will be starting from scratch. We assume further that you have a home directory `/MyHome` containing the subdirectories `/MyHome/MyData` containing data files for your use, `/MyHome/MyDocs` containing program documentation for your use, and `/MyHome/MyExe` containing executable files for your use. The directory `/MyHome/MyExe` is assumed to be on the search path for executable programs. The tree list generation database data will be installed in the directory `/MyHome/MyData`, the tree list generation documentation will be installed in the directory `/MyHome/MyDocs`, and the tree list generation database programs will be installed in the directory `/MyHome/MyExe` for easy access and use. These installation directories will also be assumed throughout the examples in Chapter 3 and Chapter 4. Installation of the developer distribution is similar and will not be discussed.

Basic installation instructions follow for the standard tree list generation database distribution. The installation is a manual process, and instructions are provided as a basic list of steps that should be followed to install the software, documentation, and data. Depending on the operating system used installation details may be slightly different, so a basic familiarity and facility with the operating system used is presumed for the installation procedure. If the instructions are followed, no warning or error messages should appear. If such a message does appear, please refer back to the installation instructions for clarification. For the installation instructions we assume that you are beginning in your home directory `MyHome`. We will begin by downloading the `.zip` archive for the standard distribution and a command line program for extracting the files from the archive.

For convenience all of the steps which manipulate files are preformed via a command shell interface. There are *drag and drop* and *point and click* equivalents for

each task via the graphical user interface for your operating system. Where possible, generic commands are used to accomplish the installation tasks. This is, however, not possible for all of the steps. Commands specific to a Microsoft Windows command shell are therefore used when necessary. This will pose no difficulties as the software currently only exists for the Windows 9x or later operating systems.

1. Decide where the tree list generation database executable files, documentation files, and data files should reside on the hard drive, creating a directory if necessary. As mentioned, the executable files will be installed in the directory `/MyHome/MyExe`, the data will be installed in the directory `/MyHome/MyData`, and the documentation will be installed in the directory `/MyHome/MyDocs`. New directories may be created within the data and documentation directories to segregate the tree list generation data and documentation if desired.

2. In your web browser, go to the SMC tree list generation database home page

   ```
   http://depts.washington.edu/silvproj/tlghome/
   ```

   and enter the download page.

3. Download the program `unzip.exe` from the download page and save it in your home directory.

4. Download the standard distribution of the tree list generation database to get the file `tlgv1r0s.zip` from the download page and save it in your home directory. You will need a logon ID and password to do this. If your browser gives an error message at this point and you have the logon ID and password, you must configure your browser to allow prompting for this information.

5. Create a temporary directory `/MyHome/tlgtemp`,

   ```
   PROMPT> mkdir tlgtemp
   ```

6. Place a copy of the `tlgv1r0s.zip` containing the tree list generation database standard distribution and the program `unzip.exe` into the temporary directory `/MyHome/tlgtemp`.

```
PROMPT> copy tlgv1r0s.zip tlgtemp

PROMPT> copy unzip.exe tlgtemp
```

   We use a temporary directory for the file extraction to make cleanup easier in the event that an error occurs during the installation procedure.

7. Change directories to `tlgtemp`,

```
PROMPT> cd tlgtemp
```

   and extract the files and directories for the standard distribution by program `unzip.exe` on the file `tlgv1r0s.zip`,

```
PROMPT> unzip tlgv1r0s
```

   which automatically preserves the directory structure in `.zip` archive. This creates a directory tree in the current directory with the top level directory name `tlgv1r0s`, which contains subdirectories for the documentation, the executable files, and a tree list generation database. The directory `tlgv1r0s` will contain the following subdirectories.

   - `data` This directory contains the various data files used to create the tree list generation database `tgdb1r00` and the tree list generation database itself.
   - `doc` This directory contains the documentation for the tree list generation database software and files.
   - `exe` This directory contains the executable programs which comprise the interface to a tree list generation database.

8. Place the executable files from the directory `tlgv1r0s/exe` into the directory `MyExe`

   ```
   PROMPT> copy tlgv1r0s/exe/*.exe ../MyExe
   ```

   which is presumed to already be on the search path for executable programs.

9. Check to be sure the tree list generation database files are on the execution path by running `TGRAND` using the option `-version`,

   ```
   PROMPT> tgrand -version
   ```

   which should produce the following output.

   *TGRAND: Version 1.0.0*

   If you did not obtain the version information from the program `TGRAND`, the directory `MyHome/MyExe` was not on the execution path. Simply add it to the execution path and try running `TGRAND` again.

10. Place the data for the tree list generation database sample files and data directory contained in the directory `tlgv1r0s/data`, and all of its subdirectories, into the directory `MyData`.

    ```
    PROMPT> xcopy/s tlgv1r0s\data\*.* ..\MyData
    ```

    This copy command will work *only* in a Microsoft Windows command shell. The option `/s` causes a recursive descent of the source directory tree while copying. The backslash is used here as the path separator to distinguish it from the option specifier. This installs the tree list generation database `tgdb1r00` and a set of example files in the directory `MyHome/MyData`.

11. Clean up after installing the tree list generation database executable programs, documentation, and data by deleting the temporary directory `/MyHome/tlgtemp` if desired.

Also available from the tree list generation database web site are a set of example files for use with this tutorial. The examples may be obtained from the *Tutorial* page accessed through the tree list generation database web site. A logon ID and password are not necessary to obtain the tutorial examples. All of the examples use files contained within the tree list generation database distributions, but they are packaged to be used easily with this tutorial. The tutorial example files but are obtained from the tutorial web page as the file `tutexamp.zip`. The tutorial examples `.zip` file contains two directories: `using` and `utils` which correspond to the using and utilities chapters of this tutorial, Chapter 3 and Chapter 4, respectively. We assume that the two examples directories in the file `tutexamp.zip` have been extracted into the directory `/MyHome/MyExamp` using the program `unzip.exe`, or some other program.

### *1.4 What's next?*

The remainder of this tutorial presents a brief introduction to the tree list generation database design, Chapter 3, and a series of examples demonstrating the use of the tree list generation database programs. The examples are divided into two parts: using a tree list generation database, Chapter 3, and the tree list generation database utilities, Chapter 4. All of the examples use the tree list generation database `tgdb1r00` as installed above.

Chapter 2 provides a brief overview of the tree list generation database design and assumptions. An understanding of the material in this chapter is important for the correct use of the tree list generation database software. A brief introduction to the structure of a tree list generation database and the relationships of its components are also presented in this chapter. The chapter closes with a brief description of the primary input and output file formats used with a tree list generation database and the features common to all of the executable programs.

Chapter 3 begins with a description of the process of the generating simulated tree lists using `TGRAND` and the installed tree list generation database `tgdb1r00` for a variety of scenarios comes next in Section 3.1. A significant advantage of the tree list generation database is the ability to update or augment a particular database with new stand measurement data. The procedures for adding stand measurement data to an existing tree list generation database using `TGADD` are discussed with examples in Section 3.2. Finally, given that data may be added to a tree list generation database, there must be a means to prevent accidental additions. The procedures for locking and unlocking a tree list generation database using `TGLOCK` and `TGUNLOCK`, respectively, are presented in Section 3.3.

Chapter 4 provides brief descriptions of the utility programs for summarizing, creating, or modifying the structure of a tree list generation database. There are two utilities for summarizing a tree list generation database. The first summary program, `TGSUMRY`, provides a brief description of the stand attributes and the numbers of multidimensional histogram bins, stand measurement data sets, and trees used for each treatment. A description of `TGSUMRY` and its use appears in Section 4.1. The second summary program, `TGSCATRP`, generates a detailed description of a tree list generation database via the centers of the multidimensional histogram bins used for the stand classification. These data may be used to determine gaps in the coverage of a tree list generation database. A description of `TGSCATRP` appears in Section 4.2.

A new tree list generation database is created using the program `TGNEW`. An example of creating a new, empty tree list generation database is in Section 4.3. Finally, an existing tree list generation database may be expanded into a set of component files using the program `TGXPLODE`. This capability permits the creation of a tree list generation database using a different stand set of stand attributes while keeping the data from an existing tree list generation database. The procedure for modifying a tree list generation database and conditions where it is warranted are described in Section 4.4.

# Chapter 2

# TREE LIST GENERATION DATABASE OVERVIEW

This chapter provides a brief overview of the tree list generation database design philosophy and assumptions, its physical structure, and the common program options. It is important that you read this overview of the tree list generation database before proceeding to the examples. A basic understanding of the tree list generation database design philosophy is essential to guarantee a correct interpretation of the simulated stands or tree lists and the correct use of the programs. The classical interpretation of the procedures for generating tree diameters from a diameter distribution may be somewhat misleading if applied to the tree list generation database, though they are solving the same basic type of problem.

We begin with a discussion of the tree list generation database design philosophy. This, then, defines the proper interpretation of simulated stands or tree lists generated using a tree list generation database and the associated software. Next, we refine these abstract concepts, defining the primary components of a tree list generation database, their relationships, and the processes of constructing and using a tree list generation database. The structure of a tree list generation database is then described, and specialized to define the characteristics of the initial tree list generation database `tgdb1r00`. Finally, the tree list generation programs are described, along with their input and output files. Only brief descriptions will be provided here, leaving the details to the examples.

### *2.1 Tree list generation database design philosophy*

The basic design philosophy for the tree list generation database follows closely the well understood paradigm of univariate random number generation or the generation of random vectors. Fundamental to the idea of random number generation is to first identify an appropriate representation for a probability density function, $f(x)$, or cumulative distribution function, $F(x) = \int_{-\infty}^{x} f(t)dt$. Frequently, a closed form for either function is not available, so appropriate in this context implies that the representation of the density function or distribution function facilitates the development of an algorithm to generate random numbers or random vectors from the distribution. We begin by identifying an appropriate representation for the distribution of trees over a region, and then specializing this general representation to enable its effective use for tree list generation.

A forested region may be represented as a collection or mosaic of more or less distinct forest patches that are distinguished by their dominant vegetation types, for example, their tree species compositions. Each patch, then, has an associated multivariate probability density function $f_i(x)$ representing its composition in terms of the individual tree sizes and tree species. Each probability density function $f_i$ may, in addition, be a function of other continuous or discrete variables, such as stand age, site index, and the distribution of individual tree ages, among other things. These are all considered to be components of the vector $x$. The distribution of trees across the entire region is then represented by the appropriately weighted summation of the individual forest patch probability density functions, i.e., $f(x) = \sum_{i=1}^{N} \alpha_i f_i(x)$, where $\sum_{i=1}^{N} \alpha_i = 1$, making $f$ a probability density function, and $N$ is the number of forest patches. This regional *mixture distribution* could then, theoretically at least, be used to generate simulated stands or tree lists that would be statistically representative of the overall forested region, both individually, as simulated patches, and for the region. This view of a forested region as a mixture distribution provides the underpinnings

of the tree list generation database.

Some forest patches in a forested region will be more similar in composition, having similar probability density functions, and some patches will be less similar, having different probability density functions. This implies that the aggregate attributes of some forest patches, e.g., stand age, site index, QMD, and species composition, will be more similar or less similar to those of other stands. Quantitatively we may think of the phrases "more similar" and "less similar" as representing a measure of the *distance* between the aggregate attributes of the patches, and, hence, in a coarse sense, as the differences in the individual forest patch probability density functions. Thus, the probability density functions for each of the individual patches may be considered to be *indexed* by a set of aggregate stand attributes, e.g., stand age, site index, QMD, and stand type. An *a priori* specified set of aggregate stand attributes may then be used to identify similar, or dissimilar, forest patches, and therefore clusters of forest patches having similar probability density functions. These considerations naturally lead to a nearest neighbor or most similar stand based approach for identifying the relevant forest patch probability density functions, and this is what is used in the tree list generation database.

So, to generate a simulated stand or tree list using the idea of the regional mixture distribution of forest patches takes three steps. First, define the attributes of the desired stand. Second, identify the most similar or nearest forest patch probability density functions within the mixture distribution. Third, use these density functions to generate the desired tree list. The remainder of this section describes how this regional mixture distribution of trees is represented and then used to generate simulated stands or tree lists in the tree list generation database.

A nonparametric, data based representation for the regional mixture distribution of forest patches, and their respective probability density functions, was used to allow the data to "speak for itself" in determining the shape of the regional mixture distribution. For the tree list generation database, a multidimensional histogram with

two resolutions was chosen as the basic representation for the regional mixture distribution of forest patches. The two resolutions of the multidimensional histogram are the stand or forest patch and the individual trees within a patch. At each resolution of the histogram, a nearest neighbor based procedure is used to sequentially, locally approximate the regional mixture distribution. This process yields an approximation to the probability density function representing the distribution of trees within each stand or forest patch through the actual tree measurement data associated with a histogram bin. Loosely speaking, we are creating a correspondence between the individual bins of the multidimensional histogram, and the trees associated with them, and the probability density functions for the forest patches, the $f_i$.

The process begins at the stand or forest patch resolution, where the aggregate stand attributes are used to associate similar stands or patches, identifying them with the same histogram bin. The individual trees from each stand or forest patch within a histogram bin are then also associated with that bin, providing a linkage from the aggregate stand or forest patch attributes to the individual trees from each stand or patch. The individual forest patch density functions are then represented by the actual individual tree diameter and height measurements and the tree species data for the trees in each histogram bin. Thus, given a description of a stand or forest patch, via a set of its aggregate attributes, the center of a histogram bin having similar attributes may be identified. The individual trees representing each stand or forest patch associated with the selected histogram bin may then be obtained. The individual trees thus obtained are then used with a nearest neighbor procedure to generate a simulated stand or tree list that is representative of the trees within the desired stand or forest patch.

The multidimensional histogram representation for the mixture distribution of the forest patches in a region has variety properties making it a natural choice. First, the histogram is a straightforward, empirically derivable representation for a probability density function. Second, the histogram, is a consistent, empirical estimator of the

*true* underlying mixture distribution. Third, the multidimensional histogram representation provides a natural index for the stands or forest patches: the centers of the histogram bins. Fourth, using the actual tree measurement data is an effective way to empirically represent the compositions of the individual stands or forest patches. Fifth, the problem of generating random numbers from a histogram representation of a probability density function is a well understood, and provides the framework for the stand generation procedures used in the tree list generation database. Finally, new data may be added to this type of mixture density representation without affecting the global properties of the tree list generation procedure, but only the local characteristics of the mixture density.

So, given a multidimensional histogram representation for the regional mixture distribution of trees within the forest patches, an algorithm for generating a simulated tree list that agrees, in a statistical or probabilistic sense, with a specified set of aggregate stand attributes may now be completely defined, and it is analogous to the more general algorithm defined for the mixture distribution. First, define the attributes of the desired stand. Second, identify the most similar or nearest histogram bin(s) to the desired attributes. Third, use the individual tree data associated with the selected histogram bin(s) to generate the desired tree list.

Within the overall framework of random number generation and this tree list generation procedure, it should be clear that the aggregate stand attributes specified *a priori* to select the tree data for the canonical stand are not necessarily exactly matched by a tree list generated by the tree list generation database. The generated tree list will have aggregate attributes that are similar in a statistical sense, within the region of coverage, to those attributes that were specified. Those attributes specified *a priori* are used to identify the appropriate, local portion of the multivariate mixture distribution of trees that is to be used when generating a simulated tree list. This interpretation is consistent with the typical use of a random number generator. Consider generating a random number $r$ from a standard normal distribution $N(0,1)$.

What we expect is that *on average*, the values of $r$ will have a mean value of zero, but we *cannot* expect that any particular random deviate $r_1$ will have a value near zero. Therefore, if a specific set of aggregate stand or forest patch attributes are to be matched to some small tolerance, repeated draws from the mixture distribution of trees may be necessary to achieve the objective. This may be somewhat counter to the traditional methods for generating simulated tree diameters. More will be said on this topic when describing the examples in Section 3.1.

The ready availability of inexpensive, high speed computers and large volume storage devices readily support a data based, nonparametric approach to the tree list generation problem as implemented in the tree list generation database. Further, the procedure for generating the simulated trees is robust, using actual tree measurement data, and therefore it can only produce simulated trees that are physically and physiologically realizable. Finally, the nonparametric, data based approach to tree list generation has proven to be quite effective in practice. The next section describes how a regional mixture distribution of trees is represented as a multidimensional histogram and then used to generate simulated stands or tree lists in the tree list generation database.

## *2.2 Tree list generation database concepts and components*

Abstractly, we think of the tree list generation database in the context of random number generation: we generate a simulated stand or tree list from the regional mixture distribution of the individual tree probability density functions on the separate forest patches. Practically, we need to translate the two resolution multidimensional histogram representation chosen for the regional mixture distribution of trees into a set of computer files and programs supporting the necessary tree list generation database capabilities. The four tree list generation database capabilities that must be supported are: (1) the addition of stand measurement data comprised of tree diam-

eters, tree heights, and tree species; (2) the indexing of the stand measurement data using a set of predetermined aggregate stand or forest patch attributes designated as the histogram dimensions; (3) the generation of simulated stands or tree lists, again comprised of tree diameters, tree heights, and tree species, that are representative of a prespecified set of aggregate stand or forest patch attributes.

The conceptual components of the tree list generation database, their relationships, and their use for generating simulated tree lists are described in this section. This description is intended to facilitate the understanding and correct use of the tree list generation database software. There are four basic components to a tree list generation database: an index of histogram bin centers, a mapping from the histogram bins to individual tree data, a data file containing individual tree measurement data from the stands or forest patches, and an ancillary data file which permits the reconstruction of the stand description information for each stand measurement. The index and tree data components are closely related to the multidimensional histogram representation used for the regional mixture distribution of tree distributions within the stands or forest patches. The issue of treated stands is then addressed, and shown to be consistent with the idea of representing the regional tree distribution in terms of a mixture distribution.

To make the discussion of the four tree list generation database components as concrete as possible, assume, for the moment, that we are concerned only with untreated stands or forest patches. We now define some useful notation. In the following description of the notation $x^T$ indicates the transpose of the vector $x$.

Let $S_1, S_2, \ldots, S_N$ represent tree measurement data sets for $N$ forest stands. Each stand $S_k$ has associated with it a vector of $p$ aggregate attributes, $A'_k = [a'_{1k}, a'_{2k}, \ldots, a'_{pk}]^T$, where each $a'_{jk}$ is a stand attribute such as stand age, site index, QMD, or plot size. The reason for the prime will become clear momentarily. Some of the attributes $a'_{pk}$ are derivable from the tree measurement data, e.g., QMD, and others are not and must be specified *a priori* if used, e.g., plot size. The same set of $p$

aggregate attributes is used for all untreated stands or forest patches. Also associated with each stand $S_k$ is a list of $n_k$ sets of tree measurements, $T_k = [t_{1k}, t_{2k}, \ldots, t_{n_k k}]^T$. Each set of tree measurements has three components, the tree diameter, the tree height, and the tree species, and the three components of tree $i$ in stand $S_k$ are given by $t_{ik} = \{d_{ik}, h_{ik}, s_{ik}\}$. Assume further that we want to represent the distribution of trees for this set of stands as a two resolution $d$-dimensional histogram, with $d \leq p$, using the aggregate attributes identified by the indices in the set $H = \{j_1, j_2, \ldots, j_d\}$. That is we want the histogram dimensions to be based on the subset of aggregate attributes defined by the vector $A_k = [a'_{j_1 k}, a'_{j_2 k}, \ldots, a'_{j_d k}]^T$. For notational convenience we will denote this $d$-dimensional subset of the aggregate stand attributes as $A_k = [a_{1k}, a_{2k}, \ldots a_{dk}]^T$. Finally, we let $H_m = [h_{1m}, h_{2m}, \ldots h_{dm}]^T$ denote the $d$-dimensional center of histogram bin $m$.

With this notation the four components of a tree list generation database for untreated stands are easily described. An index of histogram bin centers is obtained by placing a dictionary ordering on the components of the vectors $H_m$, $m = 1, 2, \ldots, M$, where $M$ is the number of histogram bins, and $M <= N$. The tree measurement data for each stand, $T_k$, are simply concatenated to form the tree measurement data file. The ancillary data file used to reconstruct the original stand descriptions are simply the complete aggregate attribute vectors $A'_k$ for each stand, which are also simply concatenated into a file. The association between the histogram bin centers $H_m$ and the tree measurement data $T_k$ is now also straightforward to describe. For each histogram bin $m$, we simply keep a list of the $n_m$ stands $S_{k_1}, S_{k_2}, \ldots, S_{k_{n_m}}$ whose attributes $A_{k_1}, A_{k_2}, \ldots, A_{k_{n_m}}$ map to the histogram bin center $H_m$. We may then easily move from vectors of stand attributes $A'_k$ to the histogram dimension vectors $A_k$, from the histogram dimension vectors to the histogram bin centers $H_m$, and finally from the histogram bin centers to the individual trees associated with a histogram bin.

The histogram bin to tree data mapping component is fundamental; it is the most

Table 2.1: Histogram bin to tree data mapping for a tree list generation database. In this hypothetical example, the stands were added to an initially empty tree list generation database in order, from $S_1$ to $S_{10}$. The bin centers were created in sequence, but they have the order shown in the table, indicating that $H_4 < H_1 < H_5 < H_2 < H_3 < H_6$.

| Histogram bin center | Associated tree lists |
|---|---|
| $H_4$ | $T_5, T_6$ |
| $H_1$ | $T_1, T_7, T_{10}$ |
| $H_5$ | $T_8$ |
| $H_2$ | $T_2, T_4$ |
| $H_3$ | $T_3$ |
| $H_6$ | $T_9$ |

critical component of the tree list generation database, enabling the generation of simulated stands or tree lists. This mapping is constructed dynamically using the following algorithm, called the *bin mapping algorithm*.

1. Compute the vector of attributes $A'_k$ for the stand and extract the subset of attributes $A_k$ associated with the histogram dimensions.

2. Compute the histogram bin center $H_m$ from the vector of attributes $A_k$.

3. If the histogram bin center $H_m$ exists in the index, say as $H_l$, associate the trees $T_k$ with histogram bin $l$. If $H_m$ does not exist, it is a *new* histogram bin, so add it to the index, and then associate the trees $T_k$ with histogram bin $m$.

4. If a new histogram bin was added to the index, resort the index of histogram bin centers.

These steps define the core processes involved in adding a stand to a tree list generation database and the simultaneous, dynamic construction of the histogram bin to tree data mapping that creates the multidimensional histogram representation for the regional mixture distribution of trees.

A hypothetical example of adding ten stands to an initially empty tree list generation database is now presented. The outcome, consisting of the ordered index of histogram bin centers and their associated tree lists, is given in Table 2.1. The stands $S_1, S_2, \ldots, S_{10}$ were added to the tree list generation database in the order specified by their subscripts. Assume that stands $S_1$, $S_7$ and $S_{10}$ are similar, that stands $S_2$ and $S_4$ are similar, and that stands $S_5$ and $S_6$ are similar. All other stands are assumed to be different. Similar stands will be mapped to the same histogram bin center. The steps of the algorithm will be shown in detail for a few stands to demonstrate how stands, and their respective tree lists, were associated with the histogram bins in this example.

We begin with an initially empty tree list generation database, so there are no histogram bins and no tree lists. We add stand $S_1$ first. Compute the attribute set $A_1'$, select the subset $A_1$ representing the histogram dimensions, and compute the histogram bin center $H_1$. There are no histogram bin centers in the index, so we add $H_1$ to the index and associate the trees $T_1$ with histogram bin 1. Now we add stand $S_2$. Compute $A_2'$, select the subset $A_2$, and compute $H_2$. $S_1$ is different from $S_2$, so $H_2$ is a new histogram bin center. Add $H_2$ to the index, associating the trees $T_2$ with histogram bin 2. Stand $S_3$ is added next. It is different from stands $S_1$ and $S_2$, so the histogram bin center $H_3$ is added to the index and the trees $T_3$ are associated with histogram bin 3. Now add stand $S_4$. We compute $A_4'$, select the subset $A_4$, and compute $H_4$. Stand $S_4$ is similar to stand $S_2$, so $H_4$ is identical to $H_2$ which is already in the index of histogram bin centers. In this case, we simply associate the trees $T_4$ with histogram bin 2. The remaining six stands are added similarly, creating three new histogram bin centers for the index.

In the example, notice that the histogram representation is built dynamically, stand by stand, as stands are added to a tree list generation database. As more stands are added the histogram bin centers forming the index fill out the space defined by the histogram dimensions. Thus, it is possible to have a tree list generation database

that contains gaps in its representation of the regional mixture distribution. That is, if no stands for a particular forest condition, or near to it, have been added to a tree list generation database it may not be possible to generate a tree list for those conditions. This limitation of the tree list generation database is fundamental, since simulated stands or tree lists are derived from actual stand measurement data. The ability to add data to a tree list generation database, however, should minimize the operational effects of this limitation.

Having described the basic algorithm for taking stand measurement data and constructing a tree list generation database, we now describe the basic algorithm for generating simulated stands. Simulated stands are generated by the following four steps which will be referred to as the *stand generation algorithm*.

1. Define a vector of desired stand attributes $A = [a_1, a_2, \ldots, a_d]$.

2. Find the $n$, or fewer, nearest histogram bin centers $H_{m_1}, H_{m_2}, \ldots, H_{m_n}$ to the vector of desired stand attributes $A$.

3. Extract the actual tree lists $T_k$ that are associated with the histogram bins $m_1, m_2, \ldots, m_n$ from the tree data file and concatenate them to form a large tree list or *canonical stand*.

4. Simulated trees are then generated using a nearest neighbor based procedure and the actual tree measurements contained in the canonical stand just created.

In the tree list generation database $n = 5$ is the default number of histogram bins used to create a canonical stand. Fewer than 5 histogram bins could be selected depending on the sparsity of the histogram bin centers near the desired stand attributes. A default cutoff distance of 5.0 is also used to restrict the number of possible matches within more densely populated regions in the space defined by the histogram bin dimensions.

Up to this point, no mention has been made of silvicultural treatments applied to stands or forest patches. Treatments naturally fit into the regional mixture distribution concept; they simply add an additional set of continuous or discrete variables to the forest patch probability density functions. An untreated forest patch $u$ could have a probability density function $f_i^u(x^u)$, where $x^u = [x_1^u, x_2^u, \ldots, x_{n_u}^u]^T$ is the vector of continuous and discrete variables affecting the value of the density function for untreated stands, and a treated forest patch $t$ could have a probability density function $f_i^t(x^t)$, where $x^t = [x_1^t, x_2^t, \ldots, x_{n_t}^t]^T$ is the vector of continuous and discrete variables affecting the value of the density function for treated stands. Thus, a mixture distribution representation including one treatment would have the form $f(x) = \sum_{i=1}^{N^u} \alpha_i^u f_i^u(x^u) + \sum_{i=1}^{N^t} \alpha_i^t f_i^t(x^t)$, where $N^u$ and $N^t$ are the numbers of untreated and treated stands, and $\sum_{i=1}^{N^u} \alpha_i^u + \sum_{i=1}^{N^t} \alpha_i^t = 1$. The vectors $x_u$ and $x_t$ may have some components in common, e.g., stand age, site index, QMD, or stand type, but could also be disjoint. Additional treatments may be added in a similar manner, e.g., $\sum_{i=1}^{N^{t_j}} \alpha_i^{t_j} f_i^{t_j}(x^{t_j})$, where $t_j$ is used to distinguish among the possible treatments.

The only potential difficulty with incorporating silvicultural treatments into the tree list generation database design occurs when the mixture distribution for the combined treatments is represented as a multidimensional histogram. The number or composition of histogram bin dimensions may be different for each treatment. This is easily resolved by separating the histograms for each treatment, allowing an independent assignment of stand attributes to histogram dimensions for each treatment. This interpretation of the mixture distribution as separate histograms for each treatment is used in the tree list generation database. The advantage is that the generation of simulated stands or tree lists for treated stands or forest patches may be performed with the *same* procedures used for untreated stands. The only treatment currently supported is thinning.

The next section describes the visible features of the tree list generation database structure, as well as defining some terminology used to refer to the conceptual com-

ponents just described.

### *2.3 Tree list generation database structure*

In this section we describe the visible features of the tree list generation database structure, e.g., the directory structure and files. We associate these visible structural components with the conceptual components and notation from the previous section to provide continuity for the ideas presented. Terminology necessary for understanding the tree list generation database file and program documentation will also be introduced in this section. The terminology introduced here will also be related to the concepts and notation from the previous section. This section acts, in part, as a bridge connecting the conceptual description of the tree list generation database to the specific implementation, files, and executable programs described in the next section.

A tree list generation database consists of a main database directory, e.g., `tlgdb`, containing a species mapping file, `spmap.dat`, a database information file having the same basename as the main database directory, `tlgdb.dat`, and a subdirectory for each available treatment. The species mapping file defines which tree species are recognized by a tree list generation database. The database information file contains data describing the current state of a tree list generation database, e.g., the number and type of treatments, the number of trees in each treatment, the number of histogram bins for each treatment, the stand attributes used for the histogram bin dimensions, and creation and modification dates. Each treatment subdirectory contains four files: a tree data file which contains the DBH, height, and species for all of the trees associated with the treatment, `treedata.dat`; a stand index file based on the histogram bin dimensions selected for the treatment, `pindex.dat`; and a file mapping the histogram bins to the tree data, `bktmap.dat`; an ancillary file of stand description data for each stand measurement associated with the treatment,

`stnddesc.dat`.

The four files within each treatment directory correspond exactly to the four conceptual components of the tree list generation database described in Section 2.2. The tree data file `treedata.dat` corresponds to the concatenated tree lists $T_1, T_2, \ldots, T_N$ for the $N$ stands contained in a tree list generation database. The stand index file `pindex.dat` corresponds to the index of histogram bin centers defined by the sorted $d$-dimensional vectors $H_m$, $m = 1, 2, \ldots, M$, derived from the histogram bin dimensions defined by the vectors of aggregate stand or forest patch attributes $A_k$. The histogram bin to tree data mapping file `bktmap.dat` corresponds to the mapping created by the bin mapping algorithm. This file is called `bktmap.dat` for historical reasons. Originally the histogram bins were called *data buckets*, so the mapping was from data buckets to trees, hence the file name. Finally, the ancillary data file `stnddesc.dat` permits the reconstruction of the stand attributes $a'_{pk}$ that are not derived from the tree data, e.g., stand age, site index, or plot size, and hence provides the data necessary to recreate the contents of the original data files used to construct a tree list generation database.

For all intents and purposes, the main database directory name, e.g., `tlgdb`, for a tree list generation database may be thought of as a single file. This interpretation is consistent with the tree list generation database executable programs and their use of the main database directory name.

## *2.4 Tree list generation database programs and files*

This section provides a brief introduction to the tree list generation database executable programs and their input and output files. Details for using the programs, the complete lists of program options and arguments, and the file formats may be found in *Tree List Generation Database User's Guide and Reference Manual*. The examples in Chapter 3 and Chapter 4 will provide specific examples of using the

programs and the relevant files.

When describing programs and files the programs use or create we are in a chicken and egg type of scenario: which do we describe first. We will describe the programs first, noting the files they use or create, and then describe the files.

### *2.4.1 Tree list generation database program descriptions*

The eight tree list generation database programs are command line oriented, and have their own online help that may be accessed by using the appropriate command line option, `-h` or `-help` with no other command line options. The tree list generation database programs are the primary means of access to a tree list generation database, providing a basic set of database oriented functions. The tree list generation database programs may be divided into two categories: programs for using a tree list generation database and utility programs. The tree list generation database programs and their functions are briefly described below. Their category, either *using* or *utility* will also be mentioned.

`TGNEW` Create a new, empty tree list generation database. This program uses the tree list generation database *schema file* to define the stand index parameters and their attributes, and the tree list generation database *species mapping file* to define the allowable tree species. Upon successful completion a new tree list generation database directory will be created and populated with the necessary initial files. Category: utility program.

`TGADD` Add a *stand measurement file* or a set of stand measurement files to a specified tree list generation database. This program is used to augment a tree list generation database with new stand measurement data. The program may be used in two modes: a single file mode or a batch mode, indicated by the appropriate command line option. The batch mode is more efficient if many stand measurement files are to be added to a tree list generation database. In single file

mode the program adds a single stand measurement file to a specified tree list generation database. In batch mode the program uses a *listing file* containing a list of stand measurement files that are to be added to a specified tree list generation database. This program produces no output files, but modifies the specified tree list generation database. Category: using program.

`TGRAND` Generate a simulated stand from a single *stand description file* or a set of simulated stands from a set of stand description files using a specified tree list generation database. For each stand description file a random stand measurement file is created. The random stand measurement file, or files, created by running the program contain the stand description and tree list data for each simulated stand that was generated. This program may be used in two modes: a single file mode or a batch mode, indicated by the appropriate command line option. The batch mode is more efficient if many random stand measurement files are to be generated from a tree list generation database. In single file mode the program generates a single random stand measurement file using a stand description file and a specified tree list generation database. In batch mode the program uses a *listing file* containing a list of stand description files for which random stand measurement files are desired, and a specified tree list generation database to generate the simulated stands. This program creates one or more random stand measurement files, but does not modify the specified tree list generation database. Category: using program.

`TGSCATRP` Obtain a summary of the histogram bin center data for a specified treatment and tree list generation database, optionally creating a *scatterplot file* containing the summary. This program provides a detailed summary of the data coverage within a tree list generation database for each treatment. It may be used to determine whether there are any gaps in the data coverage. This program may create a scatterplot file, if desired, but does not modify the specified

tree list generation database. Category: utility program.

`TGSUMRY` Summarize a specified tree list generation database, optionally creating a *summary file*. This program provides a brief summary of a tree list generation database. The summary includes the number of available treatments and the stand index parameters used for each treatment, the numbers of stand measurement files, histogram bins, and trees within each treatment, and the time and date of the last modification for the specified tree list generation database. This program may create a summary file, if desired, but does not modify the specified tree list generation database. Category: utility program.

`TGLOCK` Lock a specified tree list generation database. This prevents the addition of new stand measurement data to the specified tree list generation database. This program creates no files but changes the *locked/unlocked* status of a tree list generation database to *locked*. This makes the tree list generation database *read only*. Category: using program.

`TGUNLOCK` Unlock a specified tree list generation database. This permits the addition of new stand measurement data to the specified tree list generation database. This program creates no files but changes the *locked/unlocked* status of a tree list generation database to *unlocked*. This makes the tree list generation database *writable*. Category: using program.

`TGXPLODE` Explode a specified tree list generation database into its component files. This program will convert an existing tree list generation database into a set of text files containing all of the data necessary to reconstruct it. This program is used as a part of a three step process used to modify the set of stand index parameters and their attributes, or the tree species in the species mapping file for a specified tree list generation database. This program does not modify

the specified tree list generation database, but it creates a directory structure identical to that of a tree list generation database in a specified location, where the component files for the specified tree list generation database are placed, e.g., the schema file, the species mapping file, and the stand measurement files for each treatment. Category: utility program.

The eight tree list generation database programs are all command line based. The programs each have a set of options used to modify the program behavior or display information about the program and a set of arguments that identify the input files or the tree list generation database that are to be used. The program options are identified by having a dash or minus sign as their first character. Some of the options have arguments which must immediately follow the option name separated by one or more blanks or spaces. All of the program options must appear on the command line before any program arguments. The following list contains the command line options common to all of the programs. For the command line options specific to each program see the examples or the *Tree List Generation Database User's Guide and Reference Manual*.

`-h` Display an online help message for the program on the standard output device, typically a terminal screen, and exit.

`-help` Display an online help message for the program on the standard output device, typically a terminal screen, and exit.

`-silent` Suppress the normal program output that is written to the standard output device. Error messages, if an error occurs while the program is running, are however, still reported.

`-ver` Display the program version on the standard output device, typically a terminal screen, and exit.

`-version` Display the program version on the standard output device, typically a terminal screen, and exit.

`-path <path>` Define the directory path to a tree list generation database to be `<path>`. The tree list generation database path specifies the directory where a tree list generation database is located. If specified, the length of `<path>` may be limited depending on the operating system being used. Each path element may be at most eight (8) characters in length and all of the path elements specified must exist. This option is not required.

Of these options, the most important are the help options `-h` and `-help` and the tree list generation database directory path option `-path <path>`. The help options are important for the obvious reason that when used they present a concise description of the options and arguments necessary to use each program. The path option is important because it allows the tree list generation database programs to be used in a directory other than the specific directory in which a tree list generation database resides. The use of the `-path` option will be made clear in the examples of Chapter 3 and Chapter 4.

### *2.4.2 Tree list generation database file descriptions*

The eight tree list generation database programs use one or more of the following tree list generation database input or output files. The tree list generation database input and output files have been designed to be both easily scanned by software and easily understood by human beings. The files are text files and generally follow a *keyword equals value* format. The files may be viewed or edited using any text editor. For complete details on the formats and contents of the tree list generation database files see the *Tree List Generation Database User's Guide and Reference Manual.*

**stand measurement file** A tree list generation database stand measurement file defines the attributes of a measured stand of trees, e.g., stand age, site index, plot size, and includes a list of actual DBH and height measurements and an indication of species for each tree in the measured stand or plot. This file is used with the program `TGADD` to add actual stand measurement data to an existing tree list generation database.

**stand description file** A tree list generation database stand description file defines the attributes of a measured stand that are desired for a simulated or random stand produced by `TGRAND`. The attribute values defined in a stand description file specify values for the set of attributes used as the histogram bin dimensions, and hence the stand index, for each treatment in a specific tree list generation database.

**random measurement file** A tree list generation database random measurement file is identical in format to the stand measurement file. This file is produced by `TGRAND` and contains the attributes for stand and a simulated stand of trees. The attributes in the random stand measurement file and the tree list it contains are based on the set of attributes specified in a stand description file and a specific tree list generation database used with `TGRAND` to create the file.

**species mapping file** A tree list generation database species mapping file defines a mapping from short, 2-6 character, tree species identifiers to integer ID codes used internally by the tree list generation database software to identify the species. This file may be modified, if desired, to add new tree species or to change the tree species integer ID codes to agree with some other application. In either case, a new tree list generation database will need to be created and populated with data before the modified species mapping file will be used. This file is only used when creating a new tree list generation database with `TGNEW`.

**schema file** A tree list generation database schema file defines the stand attributes that are used to define the multidimensional histogram bin dimensions used in a particular tree list generation database and their attributes. This file is only used when creating a new tree list generation database with `TGNEW`.

A random stand measurement file may be distinguished from an actual stand measurement file by the comments appearing at the beginning of the file. At the time of writing there is no other way to distinguish actual stand measurement files from random stand measurement files. Some care must be taken to keep actual stand measurement files and random stand measurement files separated to a avoid the accidental addition of the simulated tree data to a tree list generation database.

In addition to the tree list generation database files just described, there are two output file types, a summary file and a scatterplot file, and one input file type, a listing file, that are used by the tree list generation database software. The tree list generation database summary and scatterplot files are used to provide summaries of the a tree list generation database: the available treatment types, the histogram bin centers, and the individual tree data. The listing file is used to provide a list of stand measurement or stand description file names to either `TGADD` or `TGRAND`, respectively, to be processed in a batch. The files specified in the listing file for each program must be of the appropriate type: stand measurement files for `TGADD` and stand description files for `TGRAND`, and all of the files must have valid contents and be properly formatted.

**summary file** A tree list generation database summary file is a text file that is produced by `TGSUMRY`. The text file contains a concise summary of a tree list generation database, indicating the numbers of stand measurement files, histogram bins, and trees for each treatment, the creation data and time, the date and time of last modification, among other information.

**scatter plot file** A tree list generation database scatterplot file is a comma delimited text file created by `TGSCATRP`, containing a list of the histogram bin centers for a specified treatment in a tree list generation database. The scatterplot file allows the individual data dimensions of the multidimensional histogram bin centers to be plotted against each other for each treatment. This file provides a detailed summary of the coverage for a treatment in a tree list generation database, and it may be used with graphics or plotting software to identify gaps in the data coverage.

**listing file** A tree list generation listing file is a text file containing a list of file names, one per line, specifying a set of stand measurement files or stand description files that are to be processed by `TGADD` or `TGRAND`, respectively. The file may contain comments, indicated by a percent symbol (%) as the first nonblank character of a line, or blank lines, both of which are ignored. This file provides a computationally efficient way to add multiple stand measurement files to a tree list generation database or to generate multiple random stand measurement files using a tree list generation database.

## *2.5 The tree list generation database:* `tgdb1r00`

At this point we know that the tree list generation database is derived from the idea of a multidimensional histogram approximation to a mixture distribution of probability density functions describing the distribution of trees in a forested region. We also know what the conceptual components of a tree list generation database are, how they are constructed from individual stand measurement data sets, and the relationships between the conceptual components and the files that comprise a tree list generation database. We are also familiar, and will become more so after the examples, with the tree list generation database programs and files. What we do not yet know is the specific set of aggregate stand or forest patch attributes that may be used as the

histogram dimensions in a tree list generation database, the $A'_k = [a'_{1k}, a'_{2k}, \ldots, a'_{pk}]^T$ from Section 2.2. In the context of the tree list generation database as currently implemented, these attributes are referred to as *stand index parameters* for the obvious reason: a subset of these attributes, $A_k$, is used to create the index of histogram bin centers $H_m$. The complete list of stand index parameters, $A' = [a'_1, a'_2, \ldots, a'_p]^T$ may be found in *Tree List Generation Database User's Guide and Reference Manual.*

This section defines the subsets of stand index parameters used for untreated stands and thinned stands in the initial tree list generation database `tgdb1r00`. Table 2.2 and Table 2.3 present the specific sets of stand index parameters that may be used for untreated and thinned stands, the only treatments supported by the tree list generation database implementation at the time of writing. Notice that the set of stand index parameters for untreated stands is a subset of those used for thinned stands. This creates a tree list generation database that is consistent across its treatments. This need not be the case, that is, a completely different set of stand index parameters could be used for each treatment in a tree list generation database.

A stand description file for each treatment in the tree list generation database `tgdb1r00` must contain a complete set of values for the stand index parameters to generate a simulated stands or tree lists. The description files must also be correctly formatted with appropriate attribute values specified for the stand index parameters. The stand index parameter names are not case sensitive, so `TREATMENT_TYPE` is equivalent to `Treatment_Type`. When defined in a stand description file, each stand index parameter name and its associated value must appear in a keyword equals value format, e.g., `TREATMENT_TYPE = UNTREATED`, on a single line. The sets of stand index parameters defined by Table 2.2 and Table 2.3 will be used in the stand description files in the examples. The stand index parameter names are intended to be self-explanatory. For more information on the interpretation of the stand index parameters see the *Tree List Generation Database User's Guide and Reference Manual.*

Table 2.2: Stand index parameters required in a stand description file for untreated stands, `TREATMENT_TYPE = UNTREATED` the tree list generation database `tgdb1r00`.

| |
|---|
| TREATMENT_TYPE |
| SITE_INDEX_50 |
| STAND_TOTAL_AGE |
| STAND_ORIGIN |
| QMD |
| STAND_DENSITY |
| STAND_TYPE |

Table 2.3: Stand index parameters required in a stand description file for thinned stands, `TREATMENT_TYPE = THINNED` the tree list generation database `tgdb1r00`.

| |
|---|
| TREATMENT_TYPE |
| SITE_INDEX_50 |
| STAND_TOTAL_AGE |
| STAND_ORIGIN |
| QMD |
| STAND_DENSITY |
| STAND_TYPE |
| NUMBER_OF_THINNINGS |
| PRE_THIN_DENSITY |
| PRE_THIN_BASAL_AREA |
| PCT_BASAL_AREA_REMOVED |
| YEARS_SINCE_TREATMENT |

# Chapter 3

# USING A TREE LIST GENERATION DATABASE

The tree list generation database was designed to solve the problem of generating initial tree lists for input to growth and yield models. The four aspects of the tree list generation problem that were deemed most important for the tree list generation database were: (1) generated tree lists should be as realistic as possible; (2) it should be possible to augment an existing tree list generation database with new stand measurement data; (3) the desired condition for a generated tree list should be specified by identifying a small set of average stand attributes, e.g., stand age, QMD, site index, and stand type; (4) the input and output files used with or produced by the tree list generation database software should be easy to understand, create, and use through the interface provided by the software and executable programs.

This chapter describes the typical use of a tree list generation database, demonstrating the two primary functions of the tree list generation database software. First is the generation of a simulated stand or set of stands using the program `TGRAND` in Section 3.1, and second is the augmentation of a tree list generation database with new stand measurement data using the program `TGADD` in Section 3.2. The examples presented are designed to illustrate the correct use of the tree list generation database programs and to identify potential pitfalls of their use and to identify potential misunderstandings. The chapter closes with a description of a basic locking mechanism for a tree list generation database. The locking mechanism provides a straightforward protection from the accidental addition of stand measurement data. The state of the locking mechanism is modified by using the programs `TGLOCK` and `TGUNLOCK` to lock and unlock a tree list generation database, respectively.

All of the examples rely on the installation of the tree list generation database as described in Section 1.3. In particular, we assume a home directory `/MyHome` containing the subdirectories `/MyHome/MyData`, `/MyHome/MyDocs`, `/MyHome/MyExe`, and `/MyHome/MyExamp`. These subdirectories contain the tree list generation database `tgdb1r00` and other miscellaneous files, the tree list generation database documentation, the executable files, and the examples for this tutorial broken down by topic, respectively. The directory `/MyHome/MyExe` is assumed to be on the executable search path to provide easy access to the programs for the examples. We assume that the examples for this chapter, from the `using` directory that is contained in the file `tutexamp.zip`, have been extracted and placed into the installation directory `/MyHome/MyExamp/using`.

The examples all stand alone, that is, they are all performed in separate subdirectories within the directory `/MyHome/MyExamp/using` and they have no overlapping dependencies. The name of the specific subdirectory within `/MyHome/MyExamp/using` containing each example is mentioned with their respective descriptions. The examples are intended to be performed *live* by typing the appropriate commands exactly as presented in this tutorial. All of the examples have been tested in exactly this manner, so they are known to work. The majority of the examples are for untreated stands. This should pose no difficulties as the procedures for working with thinned stands are identical, requiring only the addition of the stand index parameters and appropriate values for the thinning treatment to a stand description file. The mechanics of generating untreated and thinned stands are otherwise identical.

In several of the examples stand measurement files are added to the tree list generation database `tgdb1r00`. The stand measurement files used in the examples are already contained in the `tgdb1r00` and should not be added a second time. If you are following the tutorial and performing the examples you should first make a backup of the tree list generation database `tgdb1r00` to avoid corrupting it. If one of the example files is accidentally added to the tree list generation database `tgdb1r00`

don't panic. You can reinstall the database `tgdb1r00` or simply use it. The tree list generation database is robust to this sort of error as the number of stands it contains increases. Each example *assumes* that we begin with the unmodified tree list generation database `tgdb1r00`. If this is *not* the situation for you when you are working an example some of the output will be different than that contained in this tutorial.

All of the examples also make use of the sample files from the tree list generation database distributions, and that were installed in the directory `/MyHome/MyData`. The specific files used are the stand description files `untrt_df.dat` and `thin_df.dat` used to generate an untreated stand and a thinned stand, respectively, and the stand measurement files `untrt_mf.dat` and `thin_mf.dat` representing stand measurements for an untreated stand and a thinned stand, respectively. The contents of the files will be presented with the examples as necessary.

### *3.1 Generating simulated stands of trees*

The examples in this section demonstrate the primary function of the tree list generation database: the generation of simulated stands or tree lists that are representative of a specified stand condition. We describe four different stand generation scenarios. Each scenario describes a typical use of the tree list generation database program `TGRAND`, demonstrating its correct use with an example. Issues related to the use of `TGRAND` and its interpretation within the paradigm of random number generation will also be presented with the examples, as appropriate.

The first stand generation scenario demonstrates the generation of a single random stand measurement file from a single stand description file. This scenario also demonstrates the four stand generation modes supported by `TGRAND`. The second scenario demonstrates the generation of different simulated stands using the same stand description data. Issues relating to `TGRAND`, random number generation, and seeds

for random number generators are discussed in this example. The third scenario demonstrates how a tree list generation database may be used to generate simulated stands that agree closely with a specified set of stand attributes, e.g., QMD, mean DBH, top height, mean height, and stand basal area. `TGRAND` does not perform this task directly, but may be used with a rejection method to solve this problem. The fourth scenario demonstrates how to use `TGRAND` to generate a set of simulated stands using different stand description data. As can be imagined, this will be similar to the methods already presented, but there are again some issues relating to random number generation that are important.

The four stand generation scenarios use the stand measurement data associated with the untreated and thinned sample stands contained in the files `untrt_mf.dat` and `thin_mf.dat`, respectively, from the `data` directory of a tree list generation database distribution. The scenarios are all based on the sample stand description files Figure 3.1 and Figure 3.2, which are derived from the actual stand measurement files of similar name. These two description files are either used directly or copied to alternative file names that are then used in the examples. The two letter mnemonics `DF`, `WH`, and `OT` are used n the sample files and the examples to represent Douglas-fir trees, western hemlock trees, and trees of other species, that is, trees that are neither Douglas-fir nor western hemlock.

### *3.1.1 Generating a single simulated stand*

In this section we generate simulated stands using the sample stand description files `untrt_df.dat` and `thin_df.dat` contained in the tree list generation database distributions. These two files provide the basic contents for the files used in this example. The program `TGRAND` is used in a straightforward manner on each of the stand description files created to produce an associated random stand measurement file. This example is intended to demonstrate the basic use of `TGRAND` and some of the features available for generating simulated stands.

```
%
%  Tree list generator stand description file.
%
%  File created: 11/08/1999 at 14:19:49.55
%
MEASUREMENT_UNITS              = METRIC
TREATMENT_TYPE                 = UNTREATED
SITE_INDEX_50                  =   39.929
STAND_TOTAL_AGE                =   19.000
STAND_ORIGIN                   = NATURAL
QMD                            =   11.822
STAND_DENSITY                  = 2617.000
STAND_TYPE                     = PURE_DF
RANDOM_MF_UNITS                = METRIC
RANDOM_MF_PLOT_SIZE            =    0.041
RANDOM_MF_PURE                 = NO
RANDOM_MF_JITTER               = NO
RANDOM_MF_NAME                 = untrt.rmf
```

Figure 3.1: Sample stand description file `untrt_df.dat` from `/MyHome/MyData` which may be used as a template to generate random stand measurement files for untreated stands.

```
%
%  Tree list generator stand description file.
%
%  File created: 11/03/1999 at 16:29:24.75
%
MEASUREMENT_UNITS              = IMPERIAL
TREATMENT_TYPE                 = THINNED
SITE_INDEX_50                  =  135.900
STAND_TOTAL_AGE                =   57.000
STAND_ORIGIN                   = NATURAL
NUMBER_OF_THINNINGS            =    2.000
PRE_THIN_DENSITY               =  198.000
PRE_THIN_BASAL_AREA            =  164.000
PCT_BASAL_AREA_REMOVED         =   14.573
YEARS_SINCE_TREATMENT          =    0.000
QMD                            =   12.591
STAND_DENSITY                  =  162.000
STAND_BASAL_AREA               =  140.084
STAND_TYPE                     = PURE_DF
RANDOM_MF_UNITS                = IMPERIAL
RANDOM_MF_PLOT_SIZE            =    0.500
RANDOM_MF_PURE                 = NO
RANDOM_MF_JITTER               = NO
RANDOM_MF_NAME                 = thin.rmf
```

Figure 3.2: Sample stand description file `thin_df.dat` from `/MyHome/MyData` used to generate random stand measurement files for an thinned stands.

Table 3.1: The four stand generation modes of TGRAND and the file naming conventions used in this example. The asterisk in the file names (*) represents df, mf, or rmf for stand description files, stand measurement files, or random stand measurement files, respectively.

| Treatment | Mode | RANDOM_MF_JITTER | RANDOM_MF_PURE | File |
|---|---|---|---|---|
| Untreated | default | NO | NO | u_*.dat |
| | jitter | YES | NO | u_*_j.dat |
| | 100% | NO | YES | u_*_p.dat |
| | jitter, 100% | YES | YES | u_*_jp.dat |
| Thinned | default | NO | NO | t_*.dat |
| | jitter | YES | NO | t_*_j.dat |
| | 100% | NO | YES | t_*_p.dat |
| | jitter, 100% | YES | YES | t_*_jp.dat |

There are four possible stand generation modes supported by TGRAND: default, jitter tree heights, 100% pure, and jitter tree heights and 100% pure. The modes are determined by the values of the RANDOM_MF_JITTER and RANDOM_MF_PURE keywords in a stand description file. Both of these keywords are optional in a stand description file, and if neither appears the default mode is used. The RANDOM_MF_JITTER keyword indicates whether tree heights are to be jittered, and RANDOM_MF_PURE keywords indicates whether a generated stand is 100% pure. Values for these keywords may be YES or NO depending upon whether the modification of the stand generation procedure is desired or not. The RANDOM_MF_PURE keyword may only have a value of YES if the stand type is one of the pure stand types: PURE_DF or PURE_WH. Table 3.1 provides a concise list of the modes, the values of the description file keywords, and the file naming convention used to distinguish the stand generation modes in the example.

For this example, we create random stand measurement files using each of the four stand generation modes for an untreated stand and a thinned stand. The sample stand description files untrt_df.dat and thin_df.dat were copied to files having names conforming to those described in the table. Each of these files was then edited to set the stand generation mode by defining the values of the RANDOM_MF_JITTER

```
PROMPT> tgrand -path ../../../../MyData -df u_df.dat tgdb1r00
Processing description file:  u_df.dat
   Creating random measurement file: u_rmf.dat
PROMPT> tgrand -path ../../../../MyData -df u_df_j.dat tgdb1r00
Processing description file:  u_df_j.dat
   Creating random measurement file: u_rmf_j.dat
PROMPT> tgrand -path ../../../../MyData -df u_df_p.dat tgdb1r00
Processing description file:  u_df_p.dat
   Creating random measurement file: u_rmf_p.dat
PROMPT> tgrand -path ../../../../MyData -df u_df_jp.dat tgdb1r00
Processing description file:  u_df_jp.dat
   Creating random measurement file: u_rmf_jp.dat
```

Figure 3.3: Example of generating individual simulated stands using TGRAND. The stand description files u_df_*.dat, are used with TGRAND to create, respectively, a set of simulated untreated stands using the tree list generation database tgdb1r00 and the four stand generation modes defined in Table 3.1.

and RANDOM_MF_PURE keywords to produce the desired results. Finally, TGRAND was used to generate a simulated stand for each stand description file. Figure 3.3 and Figure 3.4 present the command lines used, and interactive results, for the untreated stands and thinned stands, respectively. The files used for this example are contained in the directory MyHome/MyExamp/using/generate/single, and from this directory the path to the tree list generation database is ../../../../MyData.

Figure 3.5 and Figure 3.6 were created using the actual data contained in the stand measurement file u_mf.dat for the sample untreated stand, and the simulated stands from the random stand measurement files: u_rmf.dat, u_rmf_j.dat, u_rmf_p.dat, u_rmf_jp.dat created in this example, see Figure 3.3. Figure 3.5 shows the untreated stand height-diameter relationships for the actual data and the four simulated stands. The actual height-diameter relationship appears to be somewhat lower than the height-diameter relationships of the simulated stands for the larger diameter trees, although it is still within the envelope of the simulated stands. Thus it appears that the simulated stands are, generally, representative of the actual stand described in the stand description files u_df_*.dat, that were derived from the actual stand measurement data. The actual height-diameter relationship is contained within

```
PROMPT> tgrand -path ../../../../MyData -df t_df.dat tgdb1r00
Processing description file:  thin_df.dat
   Creating random measurement file: t_rmf.dat
PROMPT> tgrand -path ../../../../MyData -df t_df_j.dat tgdb1r00
Processing description file:  thin_df_j.dat
   Creating random measurement file: t_rmf_j.dat
PROMPT> tgrand -path ../../../../MyData -df t_df_p.dat tgdb1r00
Processing description file:  thin_df_p.dat
   Creating random measurement file: t_rmf_p.dat
PROMPT> tgrand -path ../../../../MyData -df t_df_jp.dat tgdb1r00
Processing description file:  thin_df_jp.dat
   Creating random measurement file: t_rmf_jp.dat
```

Figure 3.4: Example of generating individual simulated stands using `TGRAND`. The stand description files `t_df_*.dat`, are used with `TGRAND` to create, respectively, a set of simulated thinned stands using the tree list generation database `tgdb1r00` and the four stand generation modes defined in Table 3.1.

the envelope of the simulated stands, though near its lower edge, and this should permit the generation of a simulated stand that is in better agreement with the actual data if this is desired.

Figure 3.6 plots the untreated stand species class percentages for Douglas-fir (`DF`), western hemlock (`WH`) and other tree species (`OT`) by basal area (BA) and number of stems, respectively, for the actual and simulated stands for each mode. This stand is a pure Douglas-fir stand, but only barely: over 19% of its basal area is not Douglas-fir and approximately 18% of its stems are not Douglas-fir. The lower percentages of western hemlock and other tree species in the simulated stands is a result of the canonical stand having a greater Douglas-fir composition than this particular actual stand. This in turn is a result of the composition pure Douglas-fir stands within the tree list generation database `tgdb1r00` which are generally in excess of 90% Douglas-fir.

Given that species percentages were not used as stand attributes for the multi-dimensional histogram dimensions this lack of agreement in these attributes is not surprising. These results indicate that the tree list generation database `tgdb1r00` may be more suited for 90% to 100% pure Douglas-fir stands. Further, marginal stands,

such as this one, which should possibly be relabelled as a Douglas-fir dominant stand, may be more difficult to simulate when broad classes are used. Tree species percentages may be used as histogram bin dimensions in a tree list generation database, but this was not done for `tgdb1r00`.

Notice the similarity in the height and diameter distributions for the untreated stand and each stand generation mode. This behavior is a result of the stand generation procedure and the random number generator used by `TGRAND`. The stand description data and the tree list generation database used to generate random stand measurement files for each generation mode were identical. Thus, the same set of histogram bins were selected, creating the same canonical stand, for each stand generation mode. The desired stand is a pure Douglas-fir stand, implying that the canonical stand is almost exclusively composed of Douglas-fir tree measurements. Tree measurements that are nearly the same must then be generated, since the sequence of random numbers generated within `TGRAND` was the same for each mode: the default random number seed was used in each case.

Figure 3.7 and Figure 3.8 were created using the actual data contained in the stand measurement file `t_mf.dat` for a thinned stand, and the simulated stands in the random stand measurement files: `t_rmf.dat`, `t_rmf_j.dat`, `t_rmf_p.dat`, `t_rmf_jp.dat` created in this example, see Figure 3.4. Figure 3.5 shows the thinned stand height-diameter relationships for the actual data and the four simulated stands. The actual height-diameter relationship appears to be somewhat higher than the height-diameter relationships of the simulated stands for the larger diameter trees, although it is still within the envelope of the simulated stands. Except for the smaller diameter trees it appears that the simulated stands are, generally, representative of the actual stand described in the stand description files `t_df_*.dat`, that were derived from the actual stand measurement data.

For the smaller diameter trees there appear to be tree measurements derived from two separate height-diameter curves. This artifact is caused by the use of computed

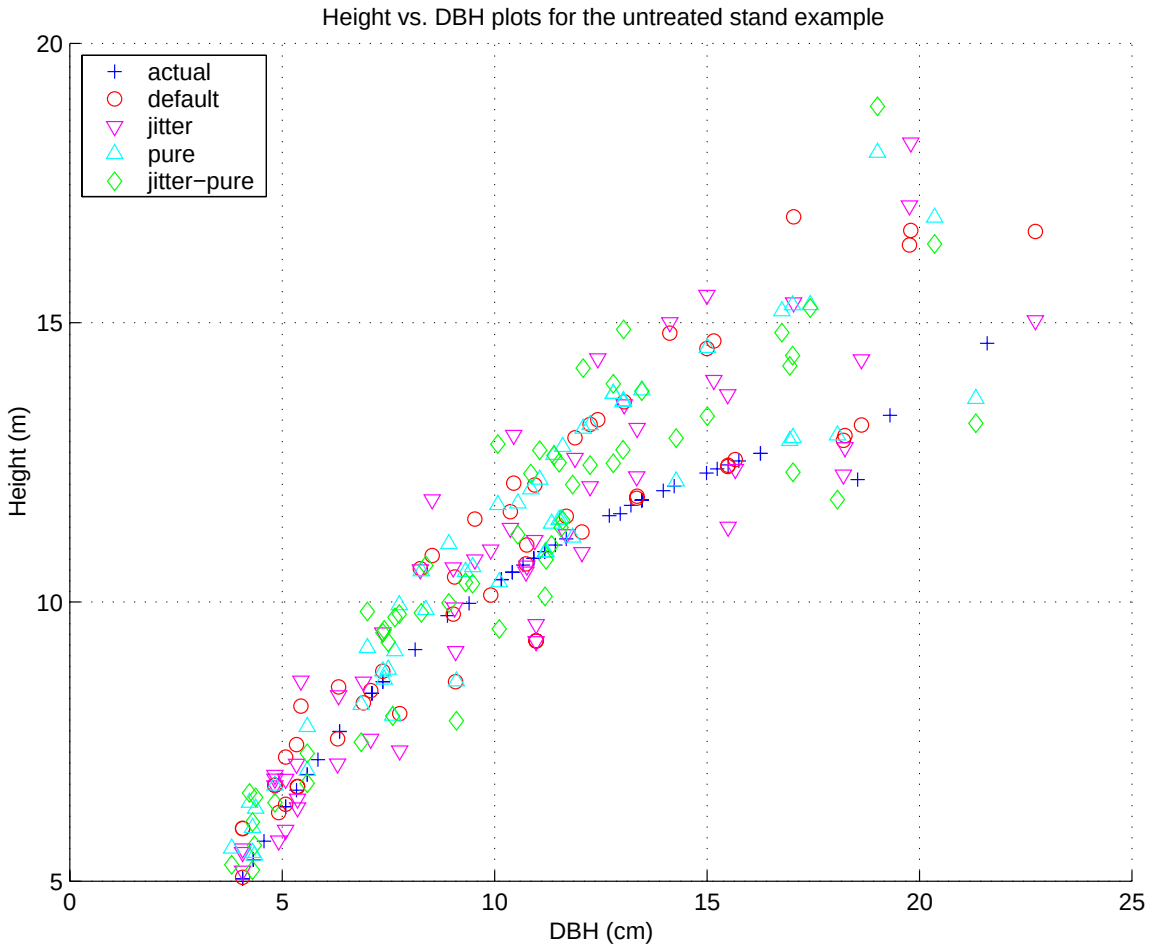


Figure 3.5: Height-diameter plots for the untreated sample stand. The actual height-diameter data for the untreated sample stand Figure 3.1 are plotted with the four simulated stands generated by the example given in Figure 3.3. Only 50% of the trees in the stands are plotted for clarity.

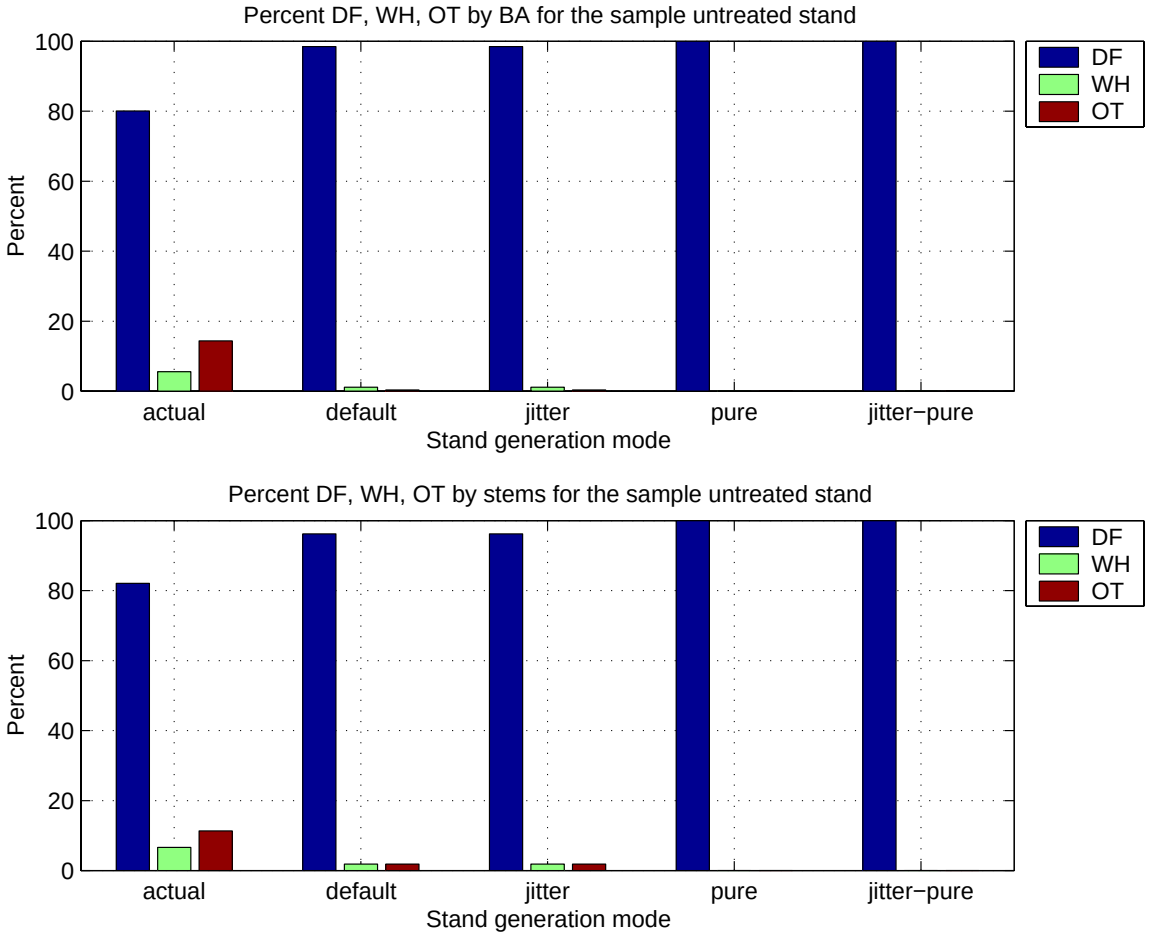


Figure 3.6: Basal area (BA) percentages (top) and stem count percentages (bottom) by species for the untreated sample stand. Percentages of Douglas-fir, western hemlock, and other tree species for the untreated sample stand Figure 3.1 are plotted with the four simulated stands generated by the example given in Figure 3.3. All of the actual or simulated trees were used for these computations.

heights for trees with unmeasured heights within a stand: these stands follow a height diameter curve with no local variability, making them seem *far* apart. Consider two similar stands of trees. In the actual stands, the tree heights are distributed around a pair of *mean* or *average* height-diameter curves, blurring the distinctions between the two stands. With the computed height-diameter curves, this blurring cannot occur, making it more difficult for the nearest neighbor algorithm to generate a specific stand. The actual height-diameter relationship is, however, more or less contained within the envelope of the simulated stands, though near its upper edge, and this should permit the generation of a simulated stand that is in better agreement with the actual data if this is desired.

Figure 3.8 plots the thinned stand species class percentages for Douglas-fir (`DF`), western hemlock (`WH`) and other tree species (`OT`) by basal area and number of stems, respectively, for the actual and simulated stands for each mode. This thinned stand and the four simulated stands were all 100% pure Douglas-fir, so their species compositions agreed exactly.

Again we see a great deal of similarity in the height and diameter distributions for the thinned stand and each stand generation mode. This occurs for exactly the same reasons as for the untreated stands: the input data are the same for each mode, so the same set of stands were selected for each mode, creating the same canonical stand, which then generated similar sized trees since the sequence of random numbers generated within `TGRAND` was the same for each mode. This issue is discussed more fully in the next section.

### *3.1.2 Generating similar simulated stands for the same attributes*

Recall the similarity in the simulated tree lists from the previous section. This similarity, as mentioned, is caused by two factors. First, the input data to the tree list generation database program `TGRAND` comprised of the description file and the tree list generation database `tgdb1r00`, were identical within each treatment except for

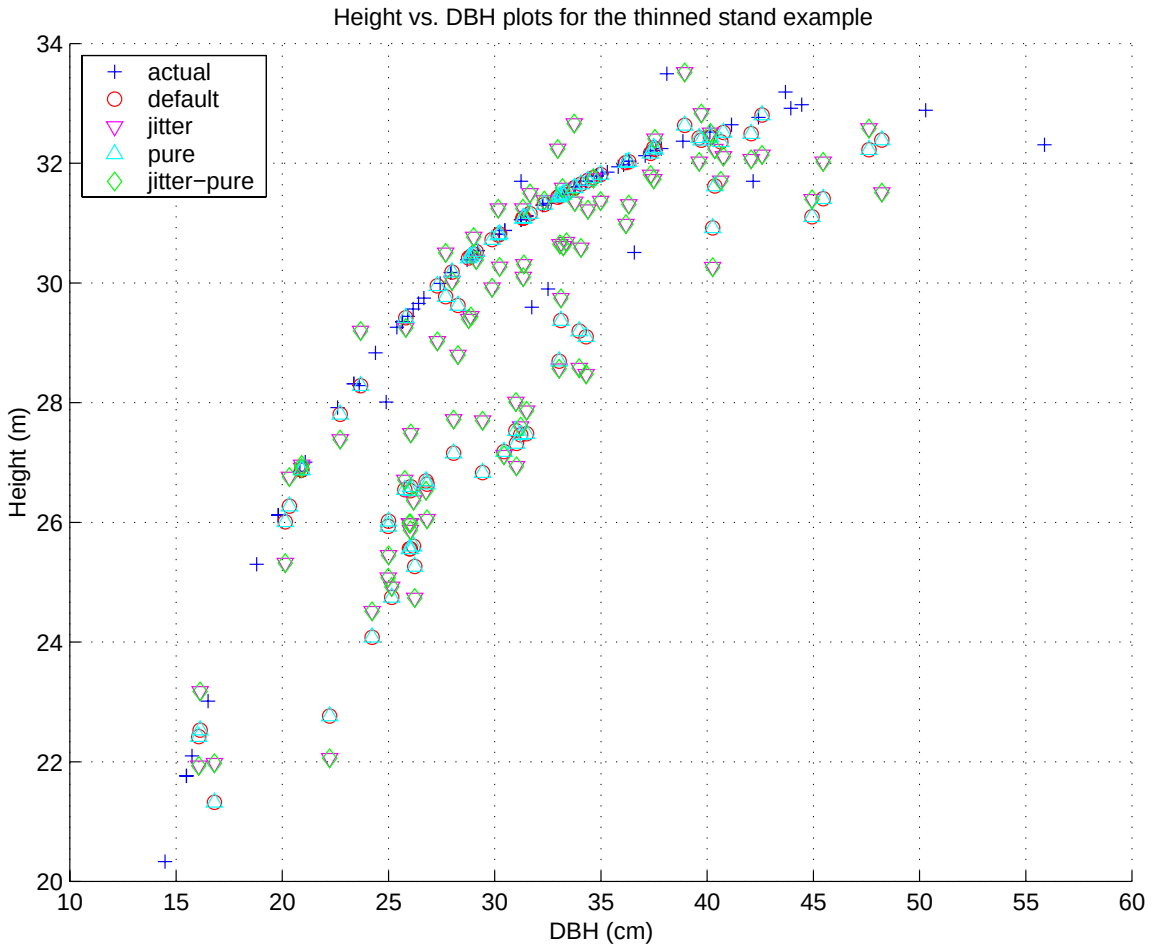


Figure 3.7: Height-diameter plots for the thinned sample stand. The actual height-diameter data for the thinned sample stand Figure 3.2 are plotted with the four simulated stands generated by the example given in Figure 3.4.

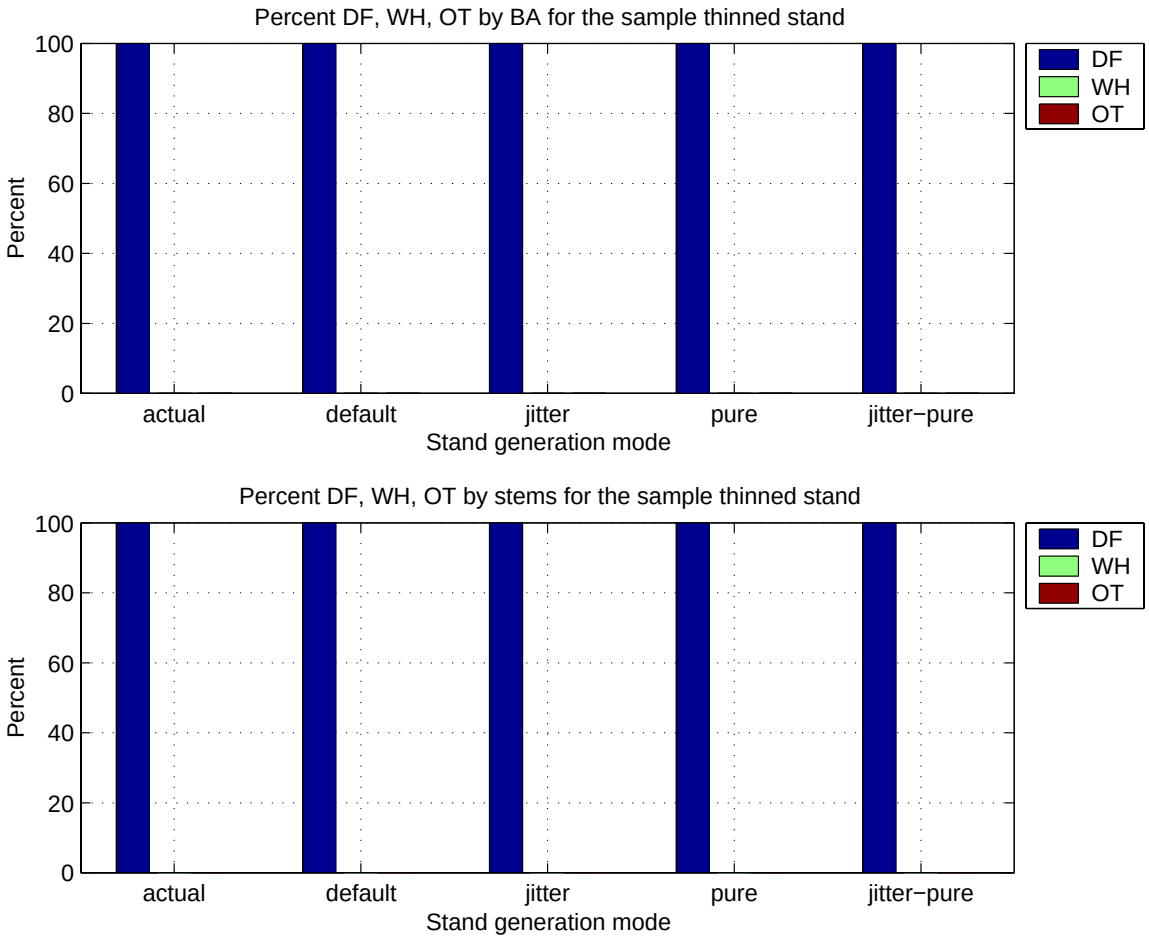


Figure 3.8: Basal area (BA) percentages (top) and stem count percentages (bottom) by species for the untreated sample stand. Percentages of Douglas-fir, western hemlock, and other tree species by basal area for the thinned sample stand Figure 3.2 are plotted with the four simulated stands generated by the example given in Figure 3.4.

the changes necessary to obtain the four stand generation modes. Second, all of the simulated stands generated used the default seed for the random number generator that is at the core of the tree list generation algorithm. For a correct understanding and use of the tree list generation database, a solid understanding of this second factor is necessary. This section describes the relevant random number generation related issues within the context of generating multiple untreated stands with `TGRAND` and an identical stand description, as specified in a stand description file. Only untreated stands are used for the examples, the issues and procedures being identical for thinned stands.

The majority of random number generators, or more precisely pseudorandom number generators, require one or more initial values, called a seed, as the beginning point for generating a sequence of (pseudo)random numbers using a deterministic or fixed algorithm. This reliance on an initial seed implies that if the *same* initial seed, whether a single value of a set of values, is used to generate two sequences of (pseudo)random numbers, then the two sequences will be *identical*. This is a direct result of the deterministic algorithms used to generate the (pseudo)random numbers. This behavior is not a defect: the repeatability of generating a sequence of (pseudo)random numbers is a desirable property for simulations for exactly the same reasons that it is desired in designed experiments. If the same results cannot be obtained for a repeated simulation or experiment, then the validity of the simulation or experiment cannot be determined.

Given the need for a seed to generate a sequence of (pseudo)random numbers, most generators have a *default* seed value that is used if an alternative seed is not specified. Thus, a program using a (pseudo)random number generator will produce *exactly* the same results for different runs if the default seed is not changed. This is the situation with the tree list generation database program `TGRAND`. If `TGRAND` is used to generate a simulated stand for two equivalent stand description files, or sets of files, without specifying different seeds, the random stand measurement files,

or sets of files, generated will be *identical*. For each stand description file, or set of files, the seed value would have been the same, the default, and hence the same sequence of random numbers would have been generated within `TGRAND`. `TGRAND`, therefore, behaves in the same manner when generating simulated stands as a random number generator behaves in generating (pseudo)random numbers. This conformance of `TGRAND` with the typical paradigm for random number generation decreases the possibility of confusion when it is used to generate simulated stands or tree lists.

In this example we consider four cases for generating similar random stand measurement files from identical stand descriptions specified in a pair of stand description files. These four cases demonstrate the random stand generation characteristics of `TGRAND`. The four cases for generating two similar simulated stands are: (1) generate them individually using the default seed; (2) generate them individually using a unique manually set seed for each file; (3) generate them simultaneously using a listing file and the default seed; and (4) generate them simultaneously using a listing file and a manually set seed. For each of the four cases we make two copies of the sample stand description file for untreated stands, `untrt_df.dat`, following the naming conventions in Table 3.2. Each pair of stand description files was then edited to produce the associated random stand measurement file indicated in the table. Finally, `TGRAND` is used to generate a pair of simulated stands for each of the four cases. A complete description of each case, the command lines used, the interactive results, and a set of figures displaying the results follows. The sample stand measurement file `untrt_mf.dat` used in each case for a comparison with the two simulated stands. The files used for this example are contained in the directory `MyHome/MyExamp/using/generate/multsame`, and from this directory the path to the tree list generation database is `../../../../MyData`.

For case (1) we are generating the random stand measurement files `u_rmfd1.dat` and `u_rmfd2.dat` individually from the identical stand descriptions in the stand description files `u_dfd1.dat` and `u_dfd2.dat` using `TGRAND` and the default seed. Fig-

Table 3.2: Files names used to demonstrate the correct ways to generate similar simulated stands from the same stand description using TGRAND. All of the stand description files u_df*.dat are identical except for their output random stand measurement file names. The * is replaced with 1 or 2 to obtain the stand description or random stand measurement file names for each pair of files.

| Simulated stand generation case | Stand description file | Random stand measurement file |
|---|---|---|
| Individually, default seed | u_dfd*.dat | u_rmfd*.dat |
| Individually, manual seed setting | u_dfm*.dat | u_rmfm*.dat |
| Listing file with default seed | u_dfld*.dat | u_rmfld*.dat |
| Listing file with manual seed | u_dflm*.dat | u_rmflm*.dat |

ure 3.9 presents the command lines and interactive output generated by TGRAND. We already know what the results in the case will be: the two generated stands, represented by the tree lists, will be *identical*. Figure 3.10 clearly demonstrates this. The plot symbols for the two simulated stands in the height-diameter plot are all coincident, indicating that the respective height values and DBH values are equal. Figure 3.11 shows that the species compositions for the two simulated stand are also identical. Both of the simulated stands also appear to be fairly reasonable when compared to the actual stand measurement data from the sample untreated stand.

An understanding of this issue is of such great importance for correctly generating simulated stands with TGRAND and a tree list generation database that we repeat the description of why it occurs from the previous section. The two simulated stands represented by the random stand measurement files u_rmfd1.dat and u_rmfd2.dat are identical for two reasons. First, identical data were used to generate the stands: the same stand description was contained in the stand description files u_dfd1.dat and u_dfd2.dat, and the same tree list generation database tgdb1r00 was used. Second, the same seed, the default seed, for the random number generator at the core of TGRAND was used to generate the two simulated stands. Specifically, the identical stand descriptions select the same set of multidimensional histogram bins, which in

```
PROMPT> tgrand -path ../../../../MyData -df u_dfd1.dat tgdb1r00
Processing description file:  u_dfd1.dat
   Creating random measurement file: u_rmfd1.dat

PROMPT> tgrand -path ../../../../MyData -df u_dfd2.dat tgdb1r00
Processing description file:  u_dfd2.dat
   Creating random measurement file: u_rmfd2.dat
```

Figure 3.9: Generate two simulated, untreated stands using the same stand description and the default `TGRAND` random number seed. The two random stand measurement files, `u_rmfd1.dat` and `u_rmfd2.dat`, are created separately using the stand description files `u_dfd1.dat` and `u_dfd2.dat`. The two stand description files differ only in the names of their respective output files and are otherwise identical, see Table 3.2 for the file naming conventions.

turn creates identical canonical stands, which finally produce identical tree lists. This last event occurs because using the same random number seed generates identical sequences of (pseudo)random numbers.

So, how can we generate similar stands from the same stand descriptions? There are actually three answers to this question, and they are described in cases (2), (3), and (4).

For case (2) we are generating the random stand measurement files `u_rmfm1.dat` and `u_rmfm2.dat` individually from the identical stand descriptions in the stand description files `u_dfm1.dat` and `u_dfm2.dat` using `TGRAND` and the `-random_seed` option to manually set unique initial seeds for each description file. Figure 3.12 presents the command lines and interactive output generated by `TGRAND`. We used seed values of `1` and `2`, which are themselves decidedly *not* random, they are sequential. They do, however, produce different sequences of (pseudo)random numbers, which is what we desire. Any two unique integers would perform the same task. Figure 3.13 clearly demonstrates that the two simulated stands are different. The plot symbols for the two simulated stands in the height-diameter plot are not coincident, thus the simulated stands contain different tree lists. Figure 3.14 shows that the species compositions for the two simulated stands are also different. Both of the simulated stands also appear

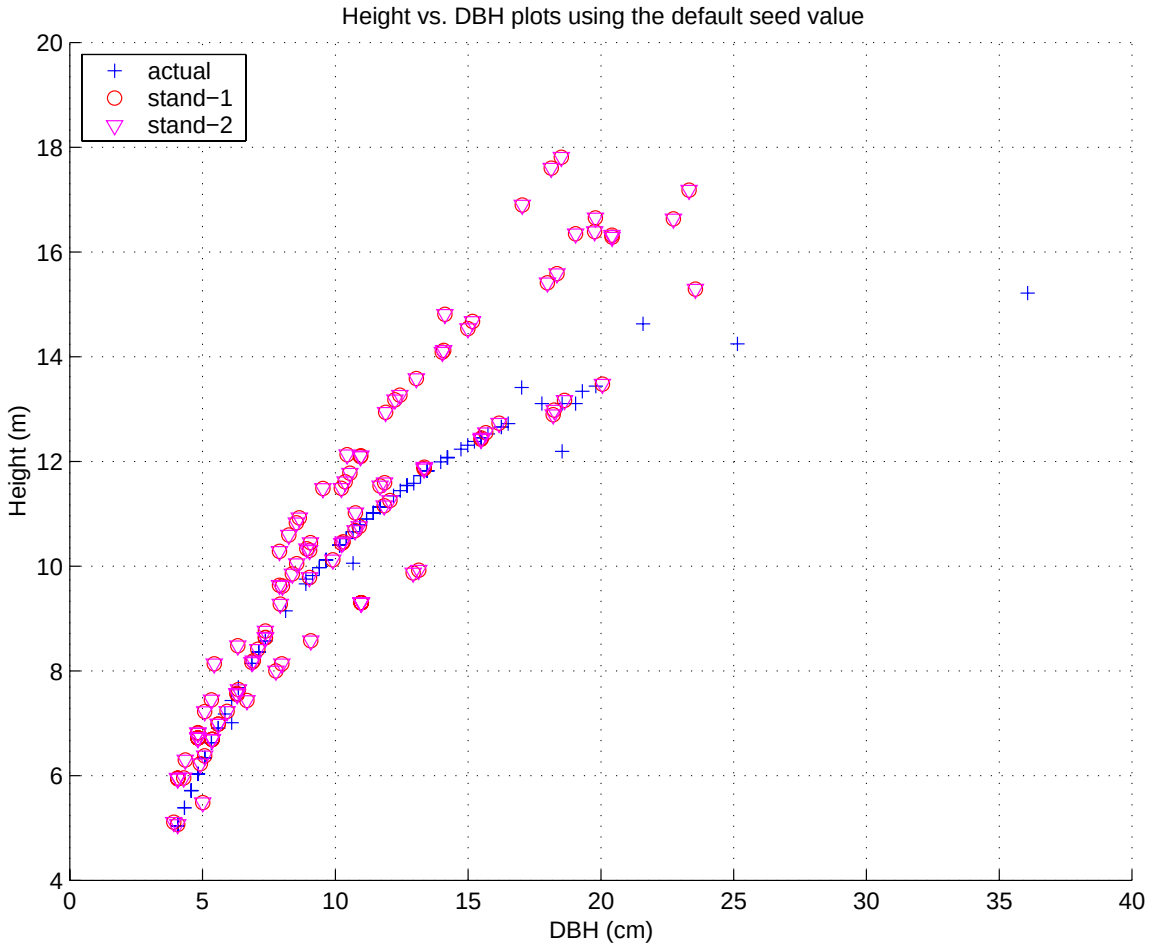


Figure 3.10: Height-diameter plots for the untreated sample stand. The plot shows the data for the two simulated stands generated individually using the same stand descriptions and the default `TGRAND` seed in Figure 3.9. Notice that the two simulated stands are identical (their symbols when plotted are coincident). The actual data are from the untreated sample stand measurement data from `untrt_mf.dat`.

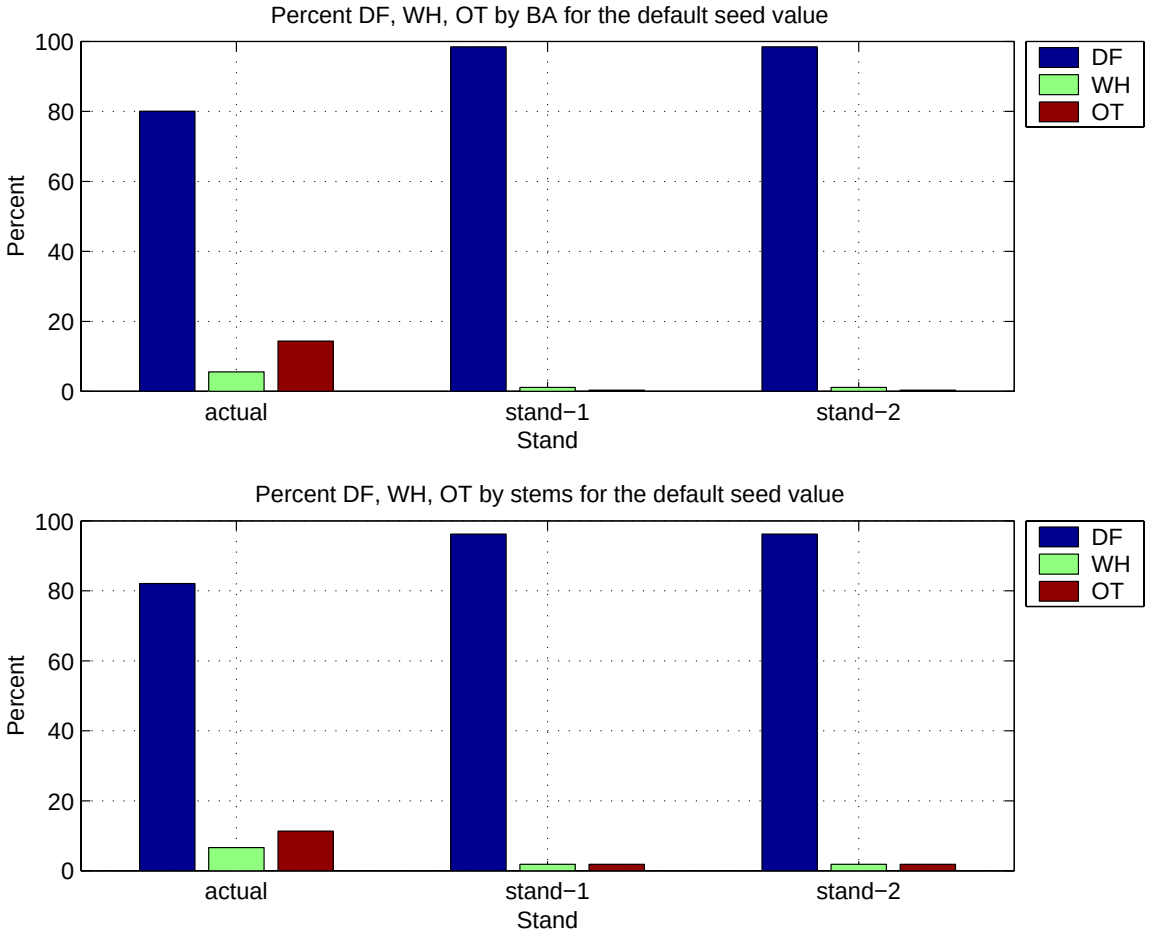


Figure 3.11: Basal area (BA) percentages (top) and stem count percentages (bottom) by species for the untreated sample stand. The plot shows the data for two simulated stands generated individually using the same stand descriptions and the default `TGRAND` seed in Figure 3.9. The actual data are from the untreated sample stand measurement data from `untrt_mf.dat`.

```
PROMPT> tgrand -path ../../../../MyData -random_seed 1 ...
   -df u_dfm1.dat tgdb1r00
Processing description file:  u_dfm1.dat
   Creating random measurement file: u_rmfm1.dat

PROMPT> tgrand -path ../../../../MyData -random_seed 2 ...
   -df u_dfm2.dat tgdb1r00
Processing description file:  u_dfm2.dat
   Creating random measurement file: u_rmfm2.dat
```

Figure 3.12: Generate two simulated, untreated stands using the same stand description and unique initial random number seeds defined by the `-random_seed` option of `TGRAND`. The two random stand measurement files, `u_rmfm1.dat` and `u_rmfm2.dat`, are created separately using the stand description files `u_dfm1.dat` and `u_dfm2.dat`. The two stand description files differ only in the names of their respective output files and are otherwise identical, see Table 3.2 for the file naming conventions. Ellipses, ..., indicate that a command line is continued on the next line.

to be fairly reasonable when compared to the actual stand measurement data from the sample untreated stand.

Generating simulated stands one at a time using the `-random_seed` option to guarantee that the simulated stands are different is a bit cumbersome, not to mention tedious if there are many random stand measurement files to generate. There is a better way to generate a set of random stand measurement files from a set of stand description files: use a listing file containing the names of the stand description files. This approach is described in case (3), with a variant in case (4).

For case (3) we are generating the random stand measurement files `u_rmfld1.dat` and `u_rmfld2.dat` simultaneously from the identical stand descriptions in the stand description files `u_dfld1.dat` and `u_dfld2.dat` with `TGRAND` using the listing file `ld_list.dat` and the default seed. The listing file simply contains the names of the stand description files, one per line, from which we want to generate a set random stand measurement file. Thus `TGRAND` may be used in a *batch* processing capacity. This method of generating the simulated stands is more efficient for large or small sets of files for two reasons. First, it *automatically* changes the seed for each stand

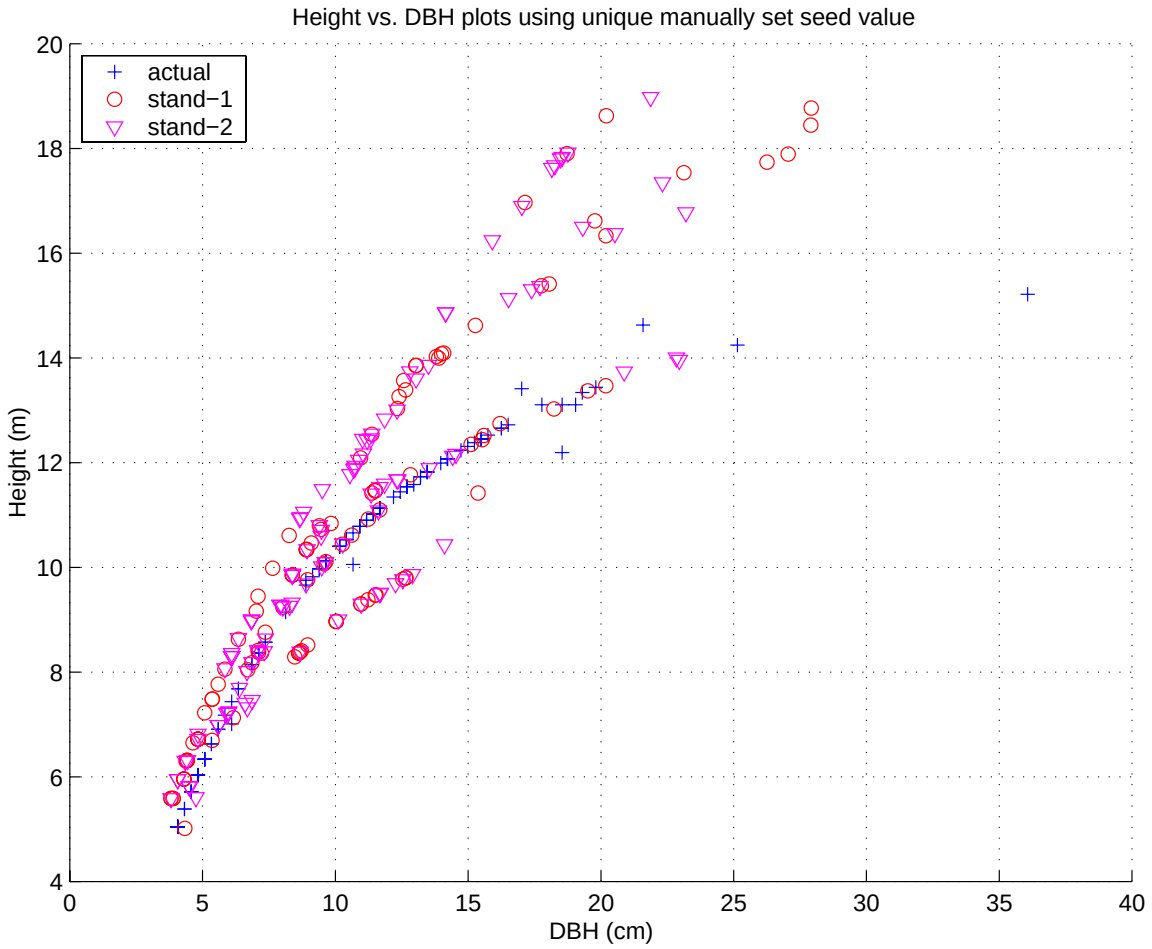


Figure 3.13: Height-diameter plots for the untreated sample stand. The plot shows the data for the two simulated stands generated individually using the same stand descriptions and unique initial seeds defined using the `-random_seed` option of `TGRAND` in Figure 3.12. The actual data are from the untreated sample stand measurement data from `untrt_mf.dat`.

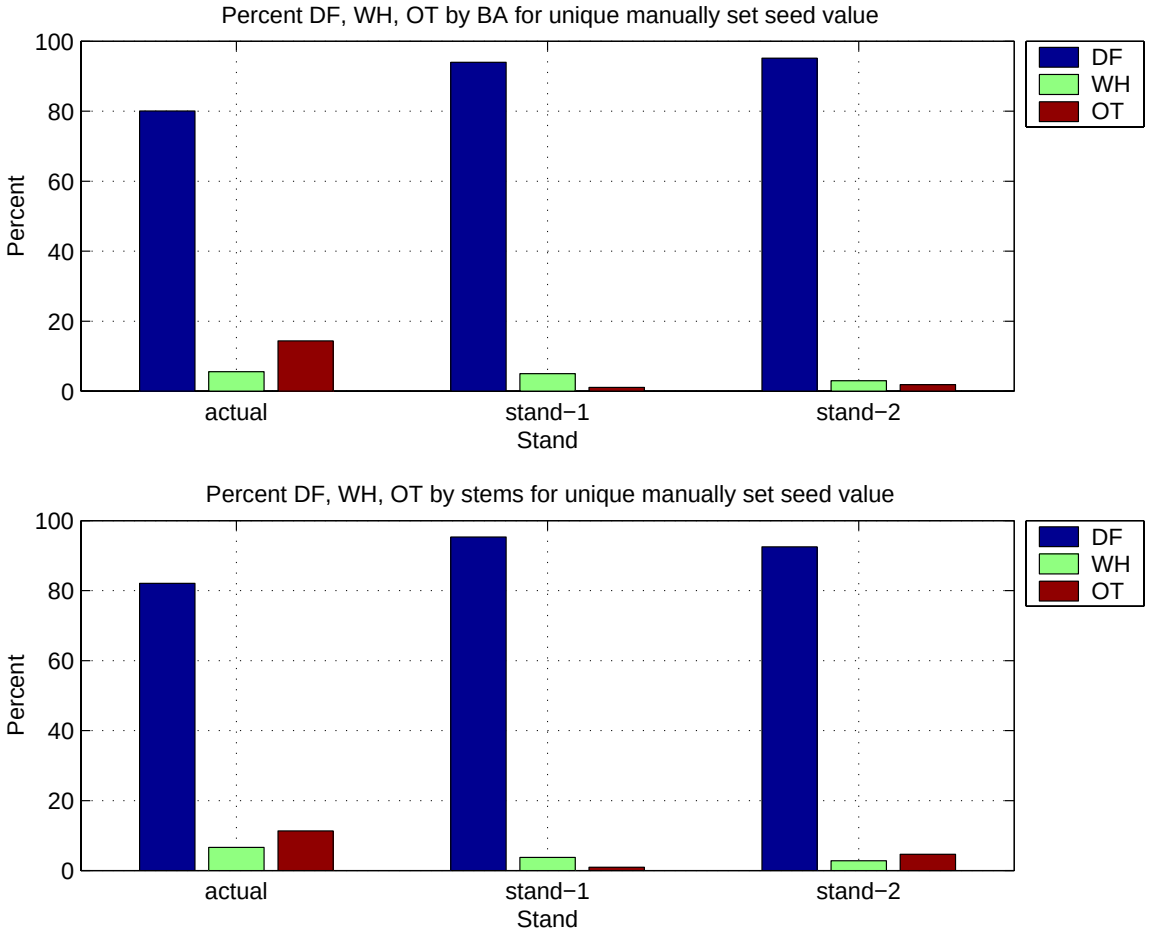


Figure 3.14: Basal area (BA) percentages (top) and stem count percentages (bottom) by species for the untreated sample stand. The plot shows the data for two simulated stands generated individually using the same stand descriptions and unique initial seeds defined using the `-random_seed` option of `TGRAND` in Figure 3.12. The actual data are from the untreated sample stand measurement data from `untrt_mf.dat`.

description file specified in the listing file, guaranteeing different simulated stands for identical stand descriptions. Second, the executable program does not need to be launched for every file, it is launched only once, so generating simulated stands will be faster.

Figure 3.15 presents the command line and the interactive output generated by `TGRAND`. Note the use of the `TGRAND` option `-lf`, rather than `-df`, indicating the file `ld_list.dat` is a listing file not a stand description file. Figure 3.16 clearly demonstrates that the two simulated stands are different. The plot symbols for the two simulated stands in the height-diameter plot are not coincident, thus the simulated stands contain different tree lists. Figure 3.17 shows that the species compositions for the two simulated stands are also different. The two simulated stands are both different from the stands generated in case (2), but the first simulated stand, `u_rmfld1.dat`, is identical to those from case (1) because the default seed was used. This may easily be seen by examining the appropriate figures. Again, the simulated stands appear to be fairly reasonable when compared to the actual stand measurement data from the sample untreated stand.

So far, so good. We can now generate similar, but different, stands individually, case (2), or in a batch case (3). Suppose, however, that we want to generate two sets of *different* stands for the same set of stand description files. This is useful for a situation where the description files represent the stands on a landscape and multiple growth and yield runs are desired to obtain an estimate of the potential variability for harvest planning or economic analyses. The procedure is identical to that for generating similar stands individually, as in case (2), but using listing files rather than stand description files, and is described as case (4).

For case (4) we are interested in the procedure for generating multiple independent sets of random stand measurement files from a given set of stand description files. We do this by using a listing file, `lm_list.dat`, containing the names of the stand description files, in this case `u_dflm1.dat` and `u_dflm2.dat`, with the `-random_seed`

```
PROMPT> tgrand -path ../../../../MyData -lf ld_list.dat tgdb1r00
Processing description file:  u_dfld1.dat
   Creating random measurement file: u_rmfld1.dat
Processing description file:  u_dfld2.dat
   Creating random measurement file: u_rmfld2.dat
```

Figure 3.15: Generate two simulated, untreated stands using the same stand description, a listing file, and the default `TGRAND` seed. The two random stand measurement files, `u_rmfld1.dat` and `u_rmfld2.dat`, are created simultaneously from the stand description files `u_dfld1.dat` and `u_dfld2.dat` using the listing file `ld_list.dat`. The two stand description files differ only in the names of their respective output files and are otherwise identical, see Table 3.2 for the file naming conventions.

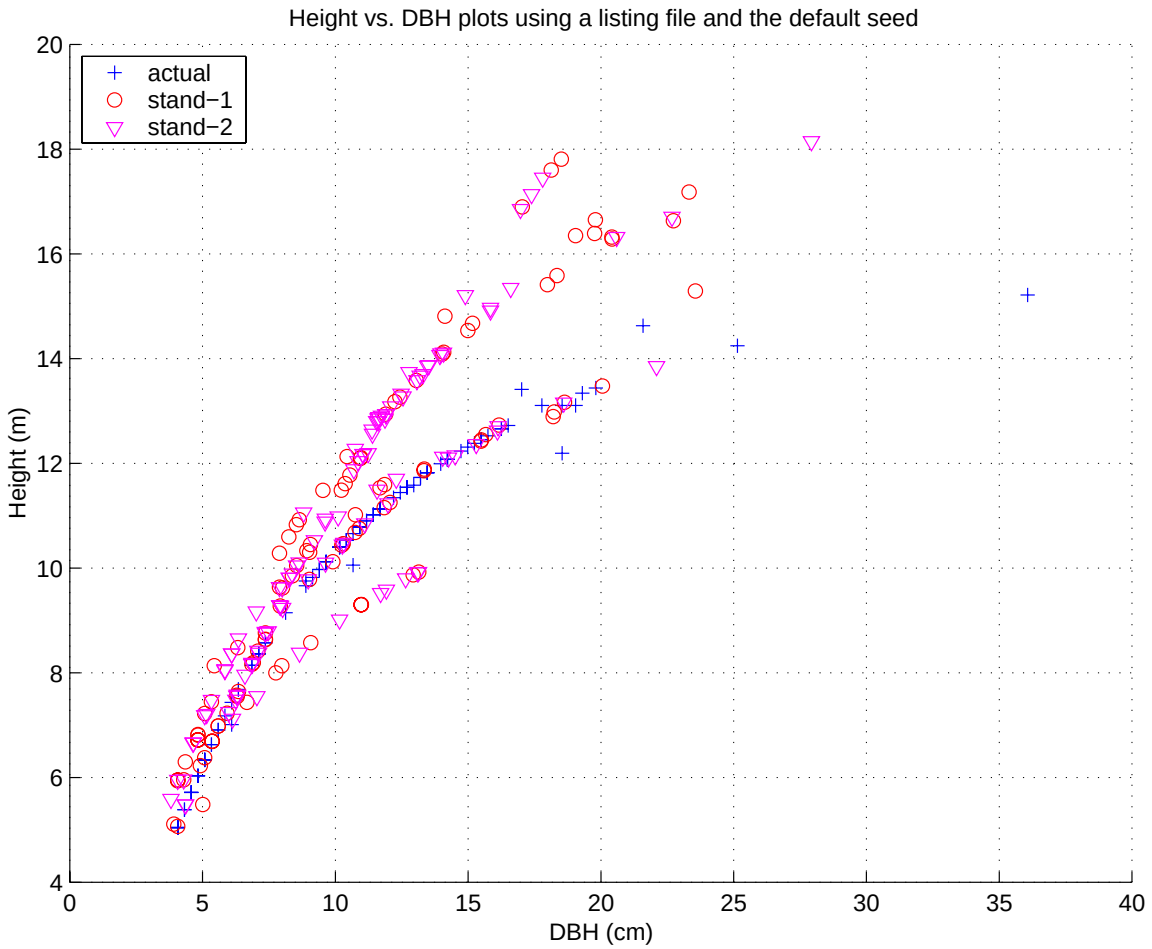


Figure 3.16: Height-diameter plots for the untreated sample stand. The plot shows the data for the two simulated stands generated simultaneously using the same stand descriptions, a listing file, and the default `TGRAND` seed in Figure 3.15. The actual data are from the untreated sample stand measurement data from `untrt_mf.dat`.

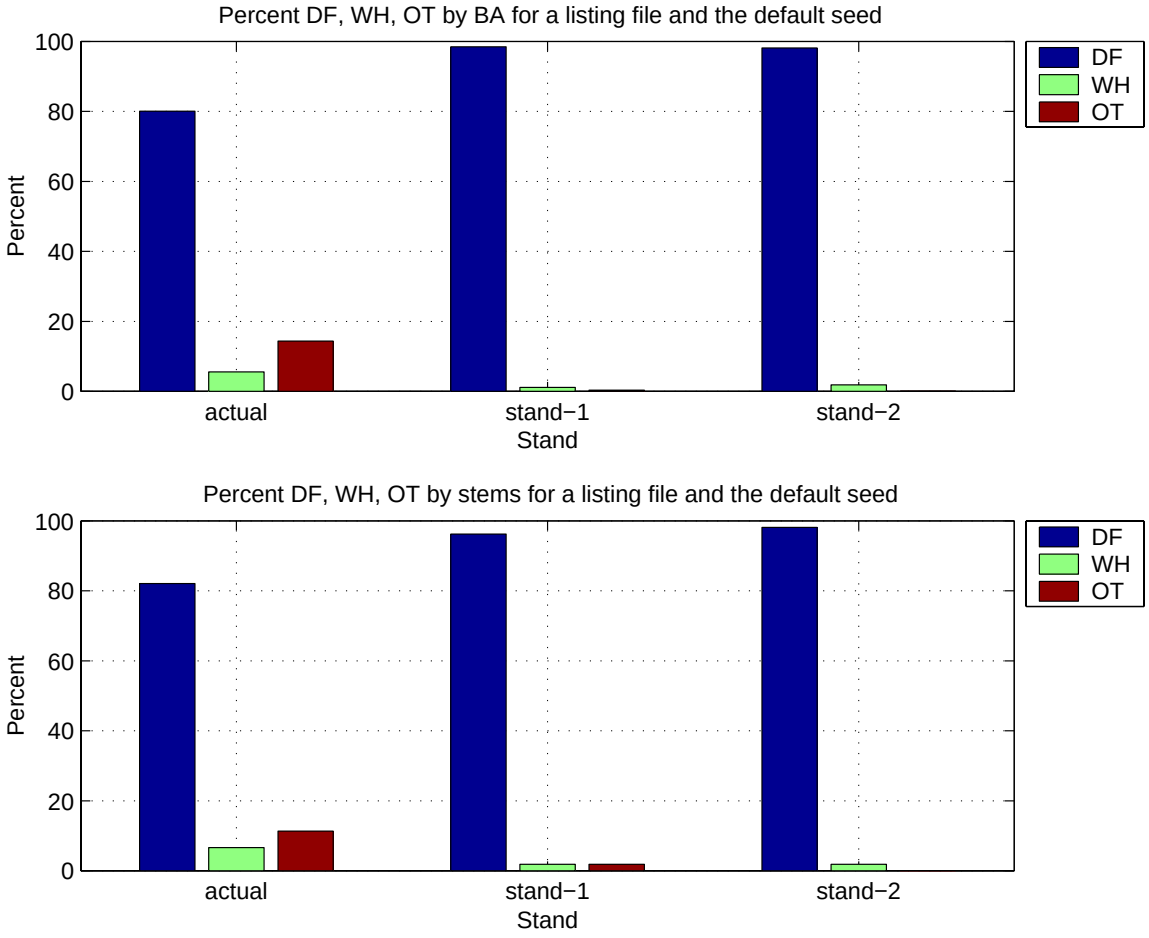


Figure 3.17: Basal area (BA) percentages (top) and stem count percentages (bottom) by species for the untreated sample stand. The plot shows the data for two simulated stands generated simultaneously using the same stand descriptions, a listing file, and the default `TGRAND` seed in Figure 3.15. The actual data are from the untreated sample stand measurement data from `untrt_mf.dat`.

option of TGRAND. We generate the random stand measurement files u_rmflm1.dat and u_rmflm2.dat in this manner, but do not create a second set of random stand measurement files, since the procedure is identical, with the only change being to use a different seed.

The different sets of random stand measurement files may be created in several different ways. First, the files could be created in the current directory and then moved to different directories for each set of simulated stands. Second, segregate or copy the set of stand description files and the listing file *a priori* to separate directories and run TGRAND using the -random_seed option with *unique* seeds in each directory. This permits using *exactly* the same stand description files and listing file, they were copied to the different directories, and simply running TGRAND in each directory. Third, independent sets of stand description files and associated listing files may be used. In this case, each set of stand description files would be designed to write their associated generated random stand measurement files to separate directories.

The first approach is simplest and uses the least storage. The second approach is almost as simple as the first, but uses more storage because copies of the same stand description files are kept in each directory. The third approach is the most difficult, keeping distinct sets of stand description files, with each set writing its output random stand measurement files to a separate directory.

Figure 3.18 presents the command line and the interactive output generated by TGRAND. Note the use of the TGRAND options -random_seed and the -lf, indicating that we are specifying a seed and that the file lm_list.dat is a listing file not a stand description file. Figure 3.16 clearly demonstrates that the two simulated stands are different. The plot symbols for the two simulated stands in the height-diameter plot are not coincident, thus the simulated stands contain different tree lists. Figure 3.20 shows that the species compositions for the two simulated stands are also different. The two simulated stands are both different from the stands generated in cases (1) and (3), but the first stand is identical to that generated in case (2) sing the initial random

```
PROMPT> tgrand -path ../../../../MyData -random_seed 1 ...
   -lf lm_list.dat tgdb1r00
Processing description file:  u_dflm1.dat
   Creating random measurement file: u_rmflm1.dat
Processing description file:  u_dflm2.dat
   Creating random measurement file: u_rmflm2.dat
```

Figure 3.18: Generate two simulated, untreated stands using the same stand description, a listing file, and an initial random number seed defined by the `-random_seed` option of `TGRAND`. The random stand measurement files, `u_rmflm1.dat` and `u_rmflm2.dat`, are created simultaneously from the stand description files `u_dflm1.dat` and `u_dflm2.dat` using the listing file `lm_list.dat`. The two stand description files differ only in the names of their respective output files and are otherwise identical, see Table 3.2 for the file naming conventions. Ellipses, . . . , indicate that a command line is continued on the next line.

number seed was `1` in both cases. This may easily be seen from an examination of the appropriate figures. Again, the simulated stands appear to be fairly reasonable when compared to the actual stand measurement data from the sample untreated stand.

We now know how to generate similar simulated stands, or sets of simulated stands, without generating identical stands, or sets of stands: use a the `-random_seed` option with `TGRAND`, use a listing file, or using a combination of both. Thus, we are able to populate forest stands with trees. Suppose, however, that we want to generate a simulated stand that agrees to within $\pm 10\%$ of the values of one or more stand attributes, e.g., QMD, mean diameter, mean height, or top height. This is a perfectly reasonable thing to want to do, and we describe the general method and an efficient approach to addressing this problem in the next section.

### *3.1.3 Generating a stand to match specific stand attributes*

Having seen in the precious section how `TGRAND` may be used to generate similar simulated stands, it seems reasonable to wonder whether it is possible to generate a simulated stand that matches a set of stand attribute values to some predetermined tolerance, e.g. $\pm 10\%$. At this time, `TGRAND` does not directly support this capability.

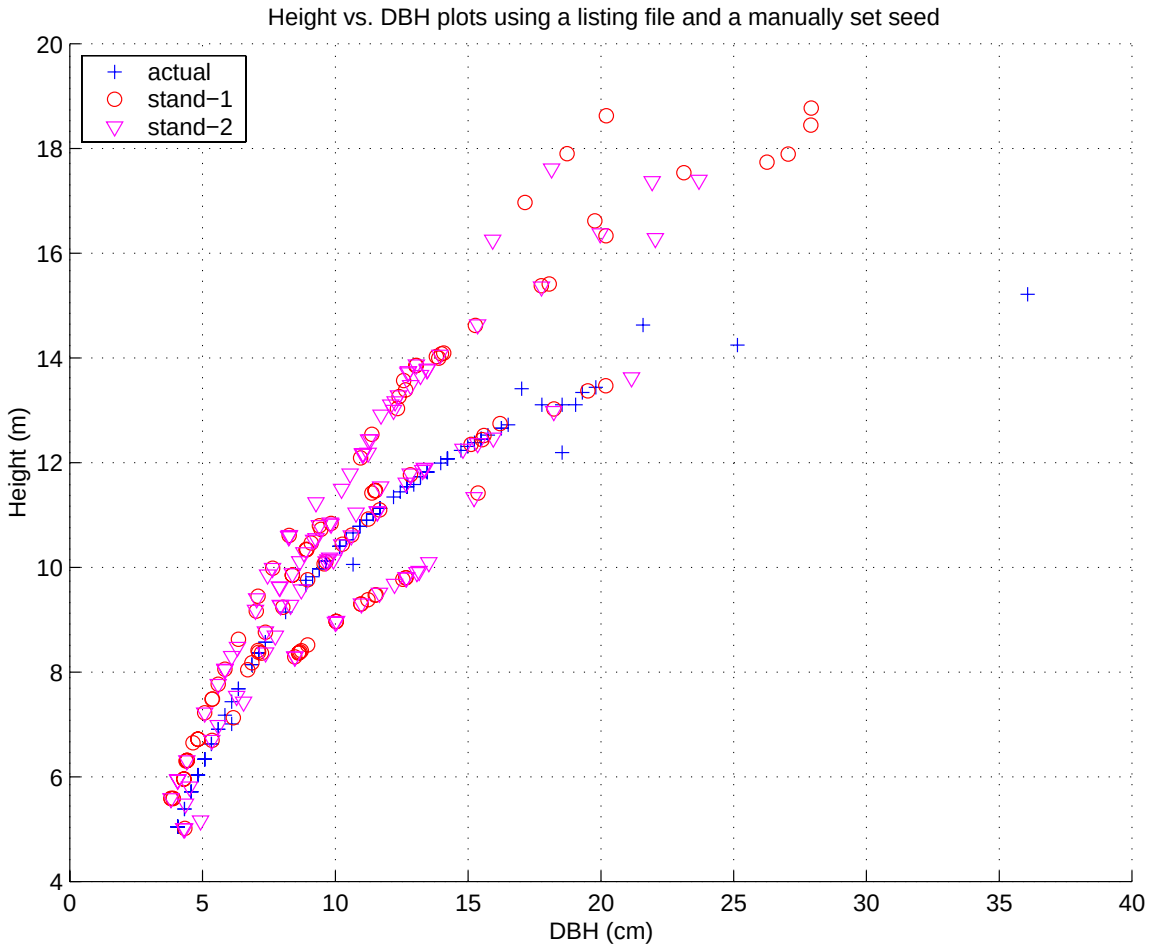


Figure 3.19: Height-diameter plots for the untreated sample stand. The plot shows the data for the two simulated stands generated simultaneously using the same stand descriptions, a listing file, and an initial seeds defined using the `-random_seed` option of `TGRAND` in Figure 3.18. The actual data are from the untreated sample stand measurement data from `untrt_mf.dat`.

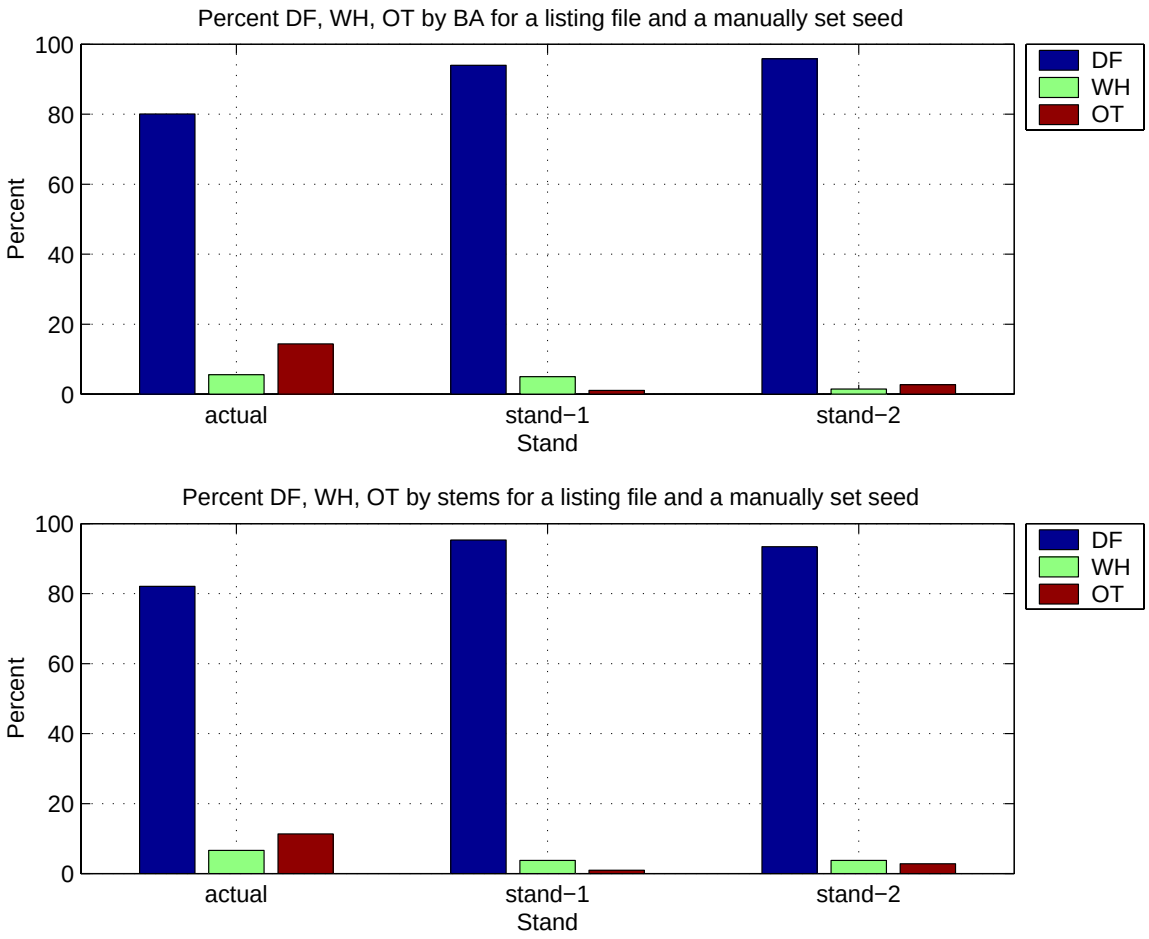


Figure 3.20: Basal area (BA) percentages (top) and stem count percentages (bottom) by species for the untreated sample stand. The plot shows the data for two simulated stands generated simultaneously using the same stand descriptions, a listing file, and an initial seed defined using the `-random_seed` option of `TGRAND` in Figure 3.18. The actual data are from the untreated sample stand measurement data from `untrt_mf.dat`.

However, using TGRAND with a *rejection rule* in the context of a *rejection method*, it is possible to generate stands matching a specified stand description to a reasonable tolerance. Rejection methods provide a set of approaches for generating random numbers meeting prespecified conditions, and are commonly used for stochastic and monte-carlo simulation systems. Therefore using TGRAND with a rejection method to generate simulated stands matching a prespecified stand description is consistent with the underlying tree list generation paradigm of random number generation.

In this section, we first give a very general definition of a rejection method with a simple example. Next, we present the tree list generation database version of a rejection method designed to match a prespecified set of stand attributes. The tree list generation database rejection method is presented first as a general algorithm, which is then specialized to an efficient method using TGRAND and a listing file. Finally, we present two examples demonstrating the use of TGRAND to generate untreated and thinned stands to match a prespecified stand description.

The definition of a rejection method is quite straightforward: the name gives it away. Let $A$ be a set of objects and $B$ be a subset of $A$, $B \subseteq A$, containing objects of interest. Given a method for randomly drawing an object $o$ from the set $A$, and an oracle for determining whether the object $o$ is contained in the set $B$, $o \in B$, a rejection method is defined be the following algorithm. First, select a random object $o$ from the set A. Next, test to see of $o$ is in the set $B$. If $o$ is not in the set $b$, $o \notin B$, reject $o$ and return to the first step. If $o$ is in the set $B$, $o \in B$, accept $o$ and continue.

A lottery may be represented as a rejection method. The set $A$ consists of the all of the possible sets of number combinations that could be played. The set $B$ consists of the numbers picked on each lottery ticket sold prior to the drawing. The method for randomly drawing a set of (possibly) winning numbers, $o$, from the set $A$ varies, but numbered ping-pong balls in a mixer of some kind is common. Once the set of (possibly) winning numbers $o$ is drawn, the oracle for determining whether $o$ is in the set of played numbers $B$ is invoked. If you have a ticket, you look at your lottery

ticket to see if the numbers match those drawn. If not, your pick was rejected. The lottery organization compares the (possibly) winning numbers $o$ to the numbers on the tickets sold, the set $B$, to see if there is a match. If the (possibly) winning numbers $o$ are in set $B$, everyone playing is rejected and there is no jackpot winner. If $o \in B$, there is a very happy ticket holder somewhere.

A rejection method for generating simulated stands or tree lists using a tree list generation database is also straightforward to describe using this notation. Given a desired stand description, the set $A$ is the set of all possible stands that may be generated for that description. The method for randomly selecting one of these stands, $o$ in the notation we are using here, is the program `TGRAND`. The set $B$, containing the stands or tree lists of interest, is defined by specifying a set of tolerances, say $\pm 10\%$ for a set of tree list derived values, e.g., mean DBH, mean height, QMD, or top height. The oracle for determining whether a simulated stand $o$ is contained in the set $B$ is a simple test of whether the set of tree list derived values for this stand are within the acceptable tolerances. The algorithm

1. Create the stand description file for the desired stand and select an initial random seed $r$.

2. Specify the value(s) for the desired set of stand attributes $a$ and determine value(s) for acceptable tolerances $p$. The set $B$ is now defined as those stands having their attributes in the interval(s) $a \pm p$.

3. Generate a simulated stand $o$ using the stand description file created in step 1 and `TGRAND` with the `-random_seed` using the value $r$ as the seed.

4. Compute the set of desired attributes $a_o$ for the generated stand. If $a_o \in B$, $a - p \leq a_o \leq a + p$, stop, we're done. Otherwise, increment the initial random seed, $r = r + 1$ and go to step 3.

is a naive implementation a rejection method to generate a stand matching a set of specified attributes. The `random_seed` option is *necessary* in step 3 because we are reusing the same stand description file.

A more efficient algorithm for generating a matching stand using `TGRAND` is possible. This algorithm processes a group of unique stand description files, containing identical stand descriptions, which create unique random stand measurement files via a listing file and `TGRAND`. In this way, a group of random stands may be generated at one time, and then checked to see if they are acceptable, within the tolerances, to determine if there is a match, in the set $B$ of acceptable stands. If no match is obtained, the process is repeated with a different initial random seed, specified using the `-random_seed` option of `TGRAND`, until at least one stand matches, i.e., is in the set $B$. The batch oriented algorithm is similar to the individual stand algorithm given above and is defined by the following steps.

1. Create $N$ stand description files for the desired stand, and create a listing file containing the names of all the stand description files.

2. Specify the value(s) for the desired set of stand attributes $a$ and determine value(s) for acceptable tolerances $p$. The set $B$ is now defined as those stands having their attributes in the interval(s) $a \pm p$.

3. Generate the simulated stands $o_i$, $i = 1, 2, \ldots, N$ using the stand listing file created in step 1 and `TGRAND`. If step 4 has failed at least once, we need to use the `-random_seed` option of `TGRAND` with the random seed value $r$.

4. Compute the set of desired stand attributes $a_{o_i}$ for the $N$ generated stands. If, for some $k$, $1 \leq k \leq N$, $a_{o_k} \in B$, $a - p \leq a_{o_k} \leq a + p$, stop, there is at least one acceptable stand and we're done. If there is no such $k$, we rejected all of the generated stands, and this is the first time this step has failed, set $r = 1$ and

proceed to step 3. If this is not the first time this step has failed, increment the initial random seed, $r = r + 1$ and go to step 3.

If the number of stand description files, and hence the random stand measurement files, $N$, is large enough *and* there is enough data near the desired stand description, it seems likely that a match could be obtained in the first batch. This batch oriented algorithm is the one we use in the examples where we generate untreated and thinned stands to match a set of prespecified stand attributes for a given stand description.

In this example we want to generate a random stand measurement file that matches, to some prespecified accuracy, a set of attributes for a particular stand description. We will demonstrate that it is possible to generate matching stands with the tree list generation database `tgdb1r00` to match an untreated stand and a thinned stand. The sample description files `untrt_df.dat`, for the untreated stand, and `thin_df.dat`, for the thinned stand, are used as the base stand description files for these examples. The associated sample stand measurement files `untrt_mf.dat` and `untrt_mf.dat`, respectively, are also used in the example to derive the desired set of stand attributes that are to be matched. For the untreated stand, we want to generate a stand that is within 10% of the values derived from `untrt_mf.dat`, and for the thinned stand we want to generate a stand that is within 5% of the values derived from `thin_mf.dat`. The different levels of agreement are used because the untreated stands will have more inherent variability than the thinned stands, so a larger acceptance tolerance seems reasonable for these stands. The attributes of interest for determining a matching stand are: mean DBH, mean height, QMD, top height, and stand basal area. These five attributes are used to determine a match for both the untreated stand and the thinned stand, and a generated stand is only considered to be a match if it *simultaneously* satisfies the tolerances for all five of these attributes.

As mentioned, the batch processing based algorithm was used for these examples. The setup steps for the untreated example and the thinned example are identical,

Table 3.3: File names used to generate a set of similar simulated stands from the same stand description using `TGRAND`. All of the stand description files `u_df*.dat` are identical except for their output random stand measurement file names. The `*` is replaced with the three digit numbers in the range `001` to `100` to obtain the unique stand description file and random stand measurement file names.

| Treatment | File type | File name(s) |
|---|---|---|
| Untreated | Sample stand measurement file | `u_mf.dat` |
| | Stand description file | `u_df*.dat` |
| | Random stand measurement file | `u_rmf*.dat` |
| Thinned | Sample stand measurement file | `t_mf.dat` |
| | Stand description file | `t_df*.dat` |
| | Random stand measurement file | `t_rmf*.dat` |

so only those for the untreated example are given in detail. First, we created $N = 100$ copies of the sample stand description file `untrt_df.dat`, `u_df*.dat`, where `*` is replaced with the three digit numbers `001` through `100`. Each of these files was edited to produce an output random stand measurement file having a similar name, replacing the `df` with `rmf` for the output file, see Table 3.3. No other changes to the stand description files were made. A copy of the sample stand measurement file `untrt_mf` was also made, `u_mf.dat`, to conform to the naming conventions used for the example. Finally, a listing file `u_list.dat` containing the names of all of the stand description files, one per line, was made. We are now ready to generate the group of 100 simulated untreated stands. For the thinned stands, simply change the `u` to a `t` and follow the steps just described.

A complete description for the untreated and thinned stand examples, the command lines used, the interactive results, and a set of figures displaying the results follows. For each treatment we will also provide a table of values for the sample stands and the *best* matching stand. The files used for this example are contained in the directory `MyHome/MyExamp/using/generate/matching`, and from this directory the path to the tree list generation database is `../../../../MyData`.

Figure 3.21 presents the command line and the interactive output generated by

```
PROMPT> tgrand -path ../../../../MyData -lf u_list.dat tgdb1r00
Processing description file:  u_df001.dat
   Creating random measurement file: u_rmf001.dat
Processing description file:  u_df002.dat
   Creating random measurement file: u_rmf002.dat
                                .
                                .
                                .
Processing description file:  u_df100.dat
   Creating random measurement file: u_rmf100.dat
```

Figure 3.21: Generate 100 simulated, similar untreated stands. The stands generated will be used to find an untreated stand matching the mean DBH, mean height, QMD, top height, and basal area per hectare of the sample untreated stand `u_mf.dat` to within $\pm 10\%$.

`TGRAND` and the listing file `u_list.dat` to generate the 100 similar untreated random stand measurement files that will be used to find a close match to the sample untreated stand measurement file `u_mf.dat`. We are seeking a simulated stand the matches the five stand attributes: mean DBH, mean height, QMD, top height, and stand basal area, simultaneously to within $\pm 10\%$,

After the random stan measurement files were created, we extracted the tree measurement data from the untreated sample stand `u_mf.dat`, the actual data for this example, obtaining the tree data $T_a^u$ and all of the simulated random stand measurement files `u_rmf*.dat`, obtaining the tree data $T_i^u$, $i = 1, 2, \ldots, 100$. From these tree data sets we then compute the five desired matching attributes, $M_a^u$, for the actual stand, and $M_i^u$, $i = 1, 2, \ldots, 100$ for the simulated stands. Percent difference vectors for the five matching attributes were computed next

$$P_i^u = \frac{(M_a^u - M_i^u)}{M_a^u}$$

and then sorted in increasing order based on their magnitudes. Finally, the simulated stand having the smallest magnitude vector of percent differences, was selected as the *best* untreated matching stand from the 100 simulated stands. If the best stand is within $\pm 10\%$ in each of its matching attributes, we are done, otherwise we need

to repeat the process used to generate the simulated stands using the `-random_seed` option of `TGRAND` to get a *different* set of 100 simulated stands.

Three out of the 100 simulated untreated stands, or 3%, agreed with the untreated sample stand to within ±10% simultaneously, for the five matching attributes and the first 100 generated stands. If we were willing to accept four out of five matching attributes, we obtain agreement for 54 out of 100 stands, or 54%. This result was quite surprising, and demonstrates the robustness of the tree list generation database methodology. Since we got three matches, we take the best one as our untreated matching stand.

Table 3.4 presents values of the five stand attributes used to determine an untreated matching stand for the actual stand, the best simulated stand, and the percent differences for the best simulated stand. Notice, in particular, that the best simulated stand has similar variation to the actual stand, as indicated by the DBH and height standard deviations. The DBH standard deviations are very similar, with the value for the best simulated stand being within 5% of the value for the actual stand. The height standard deviations are about 16% different, with the best simulated stand being larger. This occurs because the best simulated stand has larger trees than the actual stand for the larger diameters, increasing the variability, see Figure 3.22.

Figure 3.22 presents height-diameter plots of the actual stand and the best simulated stand, and Figure 3.23 presents the species composition of the actual stand and the best simulated stand. The best simulated stand does appear to be fairly reasonable when compared to the actual stand measurement data from the sample untreated stand, though possibly having taller trees for the larger diameters. The tree species composition also appears reasonable, although it was not considered in the stand matching process, that is, we were not trying to obtain agreement for the species composition as well.

The procedures for obtaining the best thinned stand were identical to those used to obtain the best untreated stand with two exceptions: the thinned sample stand data

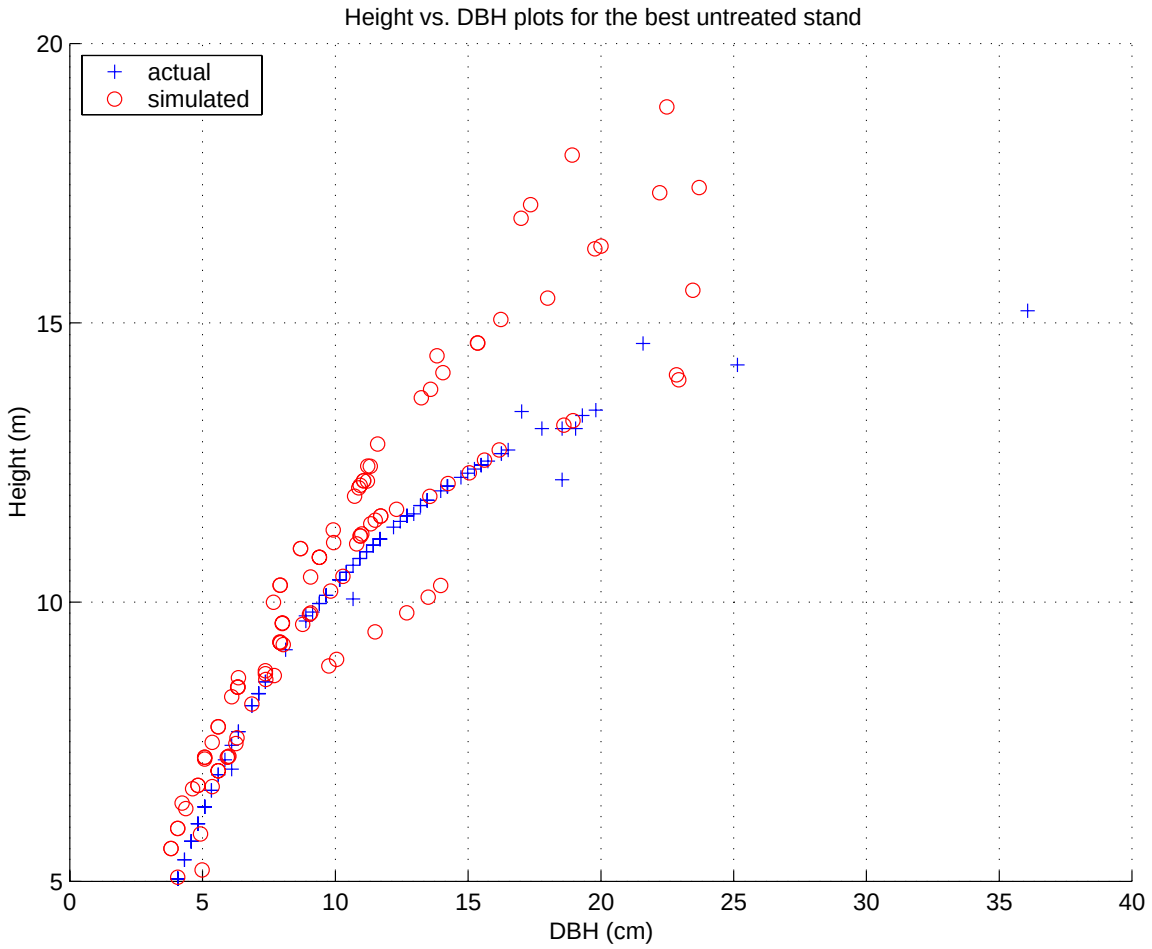


Figure 3.22: Height-diameter plots for the untreated sample stand. The plot shows the data for the sample stand and the best simulated, untreated stand simultaneously matching the mean DBH, mean height, QMD, top height, and basal area per hectare of sample stand. Table 3.4 presents a numerical summary of the agreement in these attributes. The actual data are from the untreated sample stand measurement data in the file `u_mf.dat`.

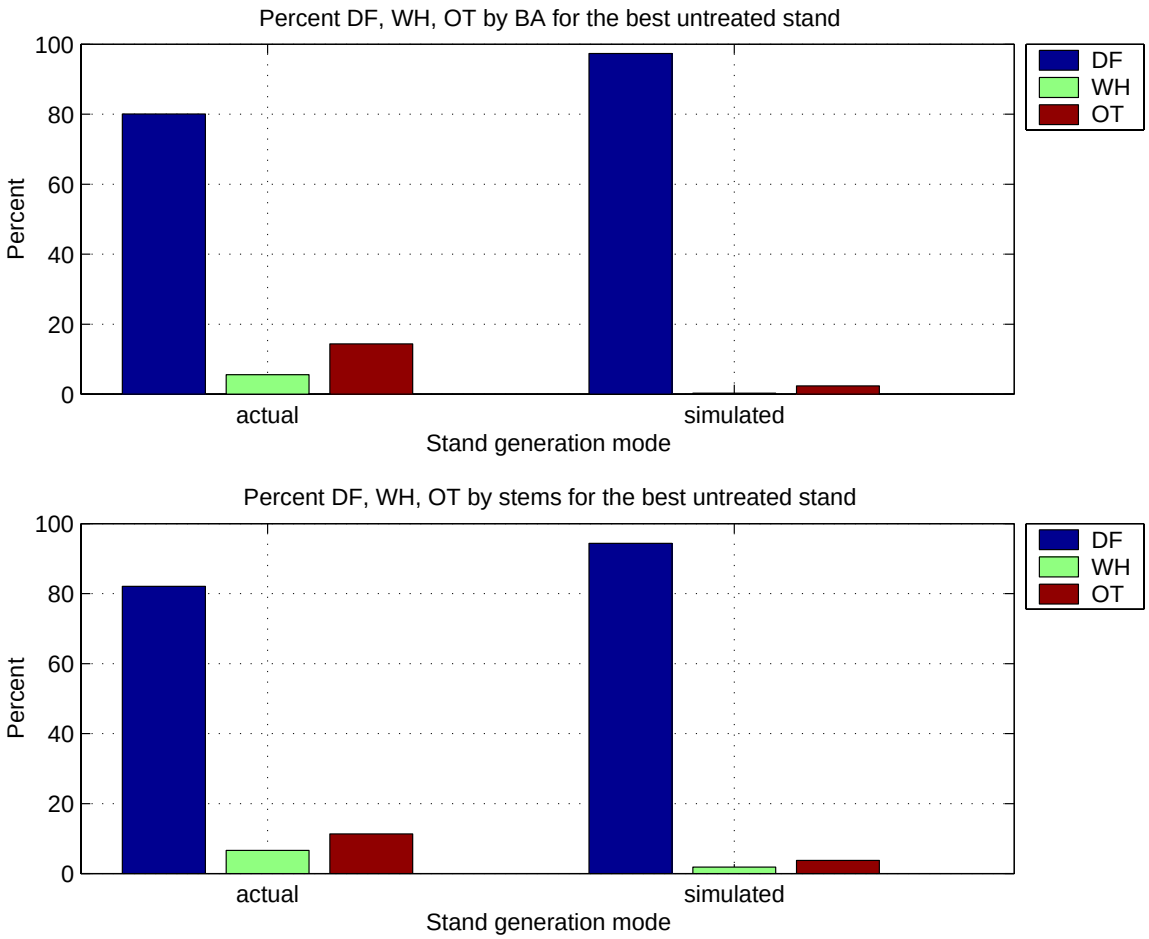


Figure 3.23: Basal area (BA) percentages (top) and stem count percentages (bottom) by species for the untreated sample stand. The plot shows the data for the sample stand and the best simulated, untreated stand simultaneously matching the mean DBH, mean height, QMD, top height, and basal area per hectare of sample stand. The actual data are from the untreated sample stand measurement data in the file `u_mf.dat`.

Table 3.4: Actual and simulated values for the best matching untreated stand. An asterisk * indicates that the value for that attribute was *not* one of those that was to be matched. They are presented to show that the generated stand or tree list may also match other properties of an actual stand. The percent differences were calculated as (actual − simulated)/actual for each attribute.

| Attribute | Actual | Simulated | % difference |
|---|---|---|---|
| Mean DBH(cm) | 10.56 | 10.46 | 1.01 |
| DBH st. dev. (cm)* | 5.33 | 5.07 | 4.82 |
| Mean height (m) | 9.73 | 10.54 | -8.41 |
| Height st. dev. (m)* | 2.73 | 3.18 | -16.12 |
| QMD(cm) | 11.82 | 11.61 | 1.76 |
| Top height (m) | 14.38 | 15.26 | -6.13 |
| Basal area ($m^2ha^{-1}$) | 28.73 | 27.98 | 2.59 |

were used as the actual data, file `t_mf.dat`, and a tolerance of ±5% was used as the acceptance criterion rather than ±10%. The simulated thinned stands were generated as in Figure 3.21 by replacing `u_` with `t_` in the figure. 31 out of the 100 simulated thinned stands, or 31%, agreed with the untreated sample stand to within ±5% simultaneously, for the five matching attributes and the first 100 generated stands. If we were willing to accept a criterion of ±10% for the five matching attributes, we obtain agreement for 80 out of 100 stands, or 80%. Since we got 31 matches, we take the best one as our thinned matching stand.

Table 3.5 presents values of the five stand attributes used to determine an thinned matching stand for the actual stand, the best simulated stand, and the percent differences for the best simulated stand. Notice, in particular, that the best simulated stand has similar variation to the actual stand, as indicated by the DBH and height standard deviations. The DBH standard deviations are very similar, having a value that is 10% smaller than the value for the actual stand. The height standard deviations are about 3% different, with the best simulated stand being larger. Overall, the average agreement between the actual stand and the simulated stand appears quite good.

Table 3.5: Actual and simulated values for the best matching thinned stand. An asterisk * indicates that the value for that attribute was *not* one of those that was to be matched. They are presented to show that the generated stand or tree list may also match other properties of an actual stand. The percent differences were calculated as (actual − simulated)/actual for each attribute.

| Attribute | Actual | Simulated | % difference |
|---|---|---|---|
| Mean DBH(cm) | 31.02 | 31.25 | -0.74 |
| DBH st. dev. (cm)* | 7.84 | 7.00 | 10.61 |
| Mean height (m) | 30.14 | 29.27 | 2.88 |
| Height st. dev. (m)* | 2.75 | 2.90 | -5.43 |
| QMD(cm) | 31.98 | 32.02 | -0.11 |
| Top height (m) | 32.35 | 32.11 | 0.74 |
| Basal area ($m^2ha^{-1}$) | 903.76 | 905.68 | -0.21 |

Figure 3.24 presents height-diameter plots of the actual stand and the best thinned simulated stand, and Figure 3.25 presents the species composition of the actual stand and the best thinned simulated stand. The best simulated stand does appear to be fairly reasonable when compared to the actual stand measurement data from the sample untreated stand, though possibly having shorter trees for the middle diameters. The taller trees in the actual data may, in part, be an artifact of the estimated height-diameter curve used to fill in unmeasured tree heights, and not necessarily be representative of the actual stand. The strong agreement in the range and distribution of tree diameters, as indicated by the mean and standard deviation, between the actual stand and the best simulated stand provides an indication that when real data are available the tree list generation methodology works well.

Next, we address the issues of generating simulated stands for different stand descriptions. The procedures are the same as those demonstrated for generating similar stands. The random number generation issues are also no different than those described for generating similar stands. We briefly describe the appropriate procedures in the next section

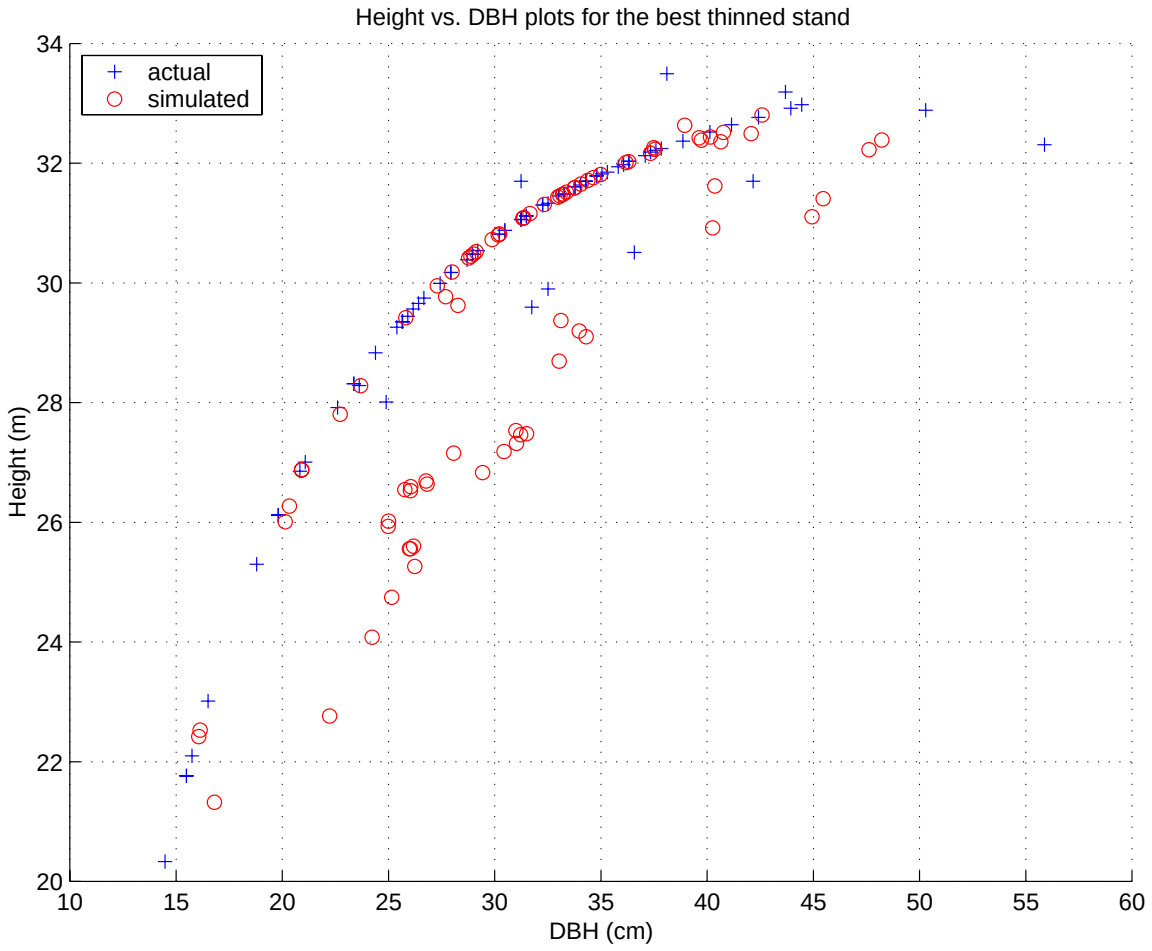


Figure 3.24: Height-diameter plot for the best matching thinned stand. The plot shows the data for the sample stand and the best simulated, thinned stand simultaneously matching the mean DBH, mean height, QMD, top height, and basal area per hectare of sample stand. Table 3.5 presents a numerical summary of the agreement in these attributes. The actual data are from the untreated sample stand measurement data in the file `t_mf.dat`.

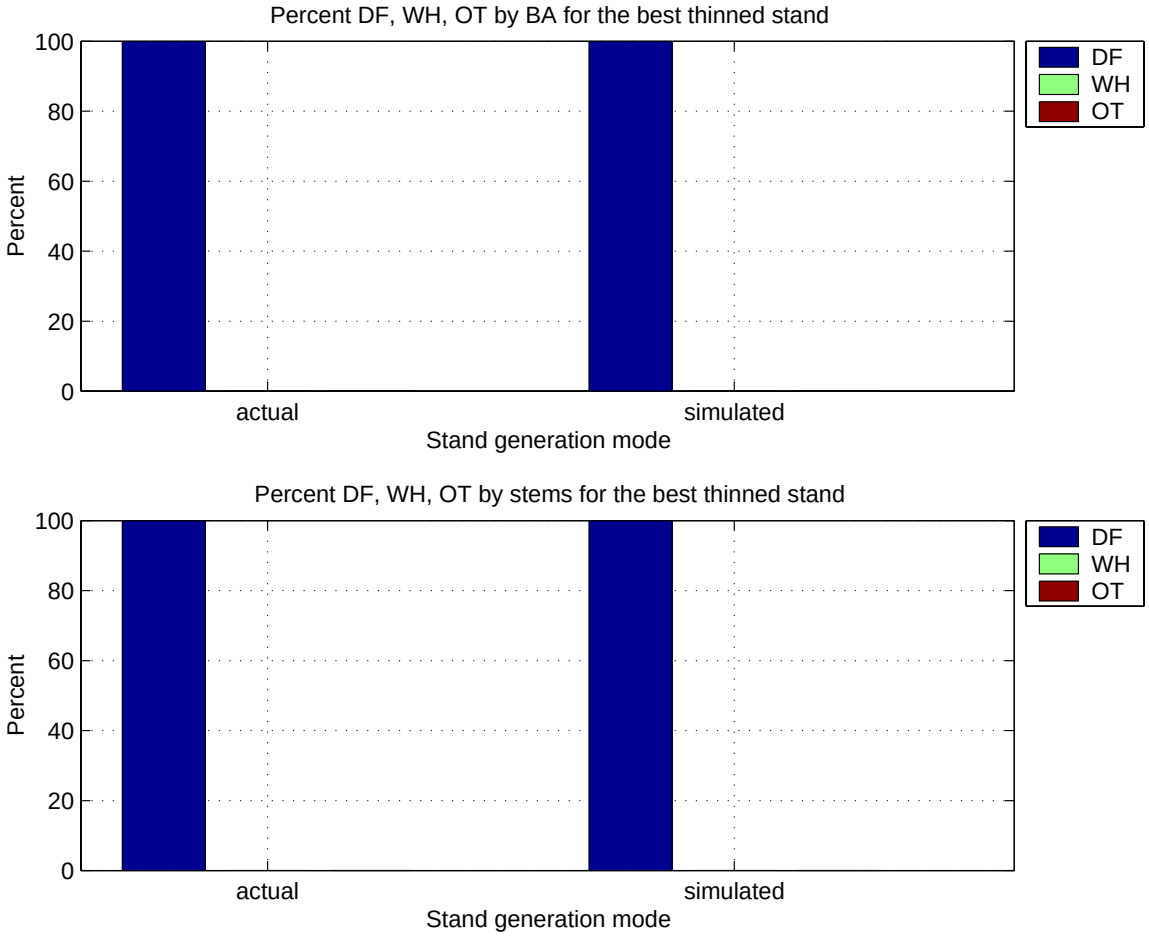


Figure 3.25: Basal area (BA) percentages (top) and stem count percentages (bottom) by species for the best thinned sample stand. The plot shows the data for the sample stand and the best simulated, thinned stand simultaneously matching the mean DBH, mean height, QMD, top height, and basal area per hectare of sample stand. The actual data are from the thinned sample stand measurement data in the file `t_mf.dat`.

```
PROMPT> tgrand -path ../../../../MyData -lf listfile.dat tgdb1r00
Processing description file:  untrt_df.dat
   Creating random measurement file: untrt.rmf
Processing description file:  thin_df.dat
   Creating random measurement file: thin.rmf
```

Figure 3.26: Generate an untreated stand and a thinned stand simultaneously using a listing file with TGRAND. The two listing file contains the names of the two sample stand description files, untrt_df.dat and thin_df.dat, which generate the random stand measurement files untrt.rmf .and thin.rmf

### *3.1.4 Generating different simulated stands*

To generate simulated stands from different stand descriptions you simply use TGRAND with the individual stand description files, or with a listing file containing the names of the stand description files, to create the desired random stand measurement files. There are no new issues or concerns for this tree list generation scenario, but the random number generation issues discussed earlier still apply. Specifically, the TGRAND option -random_seed should be used with a unique random seed for each stand description file. Further, if different sets of simulated stands are desired for the same set of stand descriptions, the -random_seed option should be used with a listing file to efficiently generate the different sets of simulated stands.

We present a single example here to demonstrate this use of TGRAND. We want to create random stand measurement file for the two sample stand description files untrt_df.dat and thin_df.dat. The random stand measurement files created will be untrt.rmf and thin.rmf. We create a listing file listfile.dat containing the names of the two stand description files, one per line, and then use this file with TGRAND. Figure 3.26 presents the command line and the interactive output for this example. The files used are in the directory MyHome/MyExamp/using/generate/matching, and the path to the tree list generation database is ../../../../MyData.

We have just completed an overview of how to generate simulated stands or tree lists using TGRAND and a tree list generation database. Next we describe how new data

may be added to an existing tree list generation database using `TGADD` to augment a database, extending the amount of data for a particular stand condition or adding data for a new stand condition for which tree lists are desired.

### *3.2 Adding new stand measurements to a tree list generation database*

One of the key features of the tree list generation database is the ability to add new stand measurement data, using the program `TGADD`, to an existing database as the data become available. This permits the augmentation of a tree list generation database with data to close gaps in its coverage, to add new stand conditions, or to add new treatments, provided the treatments are available in the tree list generation database software being used. The ease with which data may be added to a tree list generation database makes this method of generating tree lists a powerful addition to a growth and yield modeling toolbox.

This section presents examples demonstrating the two ways that new stand measurement data may be added to a tree list generation database. New stand measurement data, stored in tree list generation database stand measurement files, may be added one file at a time or as a group of files. The latter method is significantly more efficient if many files are to be added to a tree list generation database.

The following examples modify the tree list generation database `tgdb1r00`. If you do *not* want to modify your installation of this tree list generation database, make a copy before working through the examples and apply the examples to the copy.

**Before adding any files to a tree list generation database make a back-up copy of the existing database.** When new stand measurement files are added to a tree list generation database they are added *live*, that is, the database is updated for each file. It is, therefore, possible to corrupt an existing tree list generation database when adding one file or a set of files.

### *3.2.1 Adding a single stand measurement file*

The TGADD program is used to add stand measurement files to a tree list generation database. The stand measurement files must be correctly formatted with valid content in order for TGADD to process them and add them to a tree list generation database. TGADD will work seamlessly with stand measurement files having different treatment types, so long as the treatment types are valid and available in the version of the tree list generation database being used, that is, so long as the treatment types used are supported.

In this example, we add the two sample stand measurement files untrt_mf.dat and thin_mf.dat to the tree list generation database tgdb1r00. Before adding any files to the tree list generation database tgdb1r00 we must first *unlock* it using the program TGUNLOCK, see the next section, Section 3.3. The tree list generation database tgdb1r00 is initially locked to prevent the accidental addition of stand measurement data. Before unlocking the database, we try to add a file using TGADD Figure 3.27 to show the error that is reported by the program. We then unlock the database and proceed to add the two files. Figure 3.28 demonstrate adding individual stand measurement files to a tree list generation database using TGADD, presenting the command lines used and the resulting interactive output.

There are no directly visible changes to the tree list generation database tgdb1r00 caused by the addition of a file, except that the sizes of the files comprising the database may change. The program TGSUMRY, see Section 4.1, may be used to create a before-and-after picture of the tree list generation database contents. The files used for this example are contained in the directory MyHome/MyExamp/using/adding/single, and the path to the tree list generation database is ../../../../MyData.

```
PROMPT> tgadd -path ../../../../MyData -mf untrt_mf.dat tgdb1r00

=======================================================================

Toolkit version:  N0048

TGADD(DATABASELOCKED)

The tree list generation database 'tgdb1r00' located in the directory
'..\..\..\..\MyData' is currently locked.  In order to add data you
must first unlock the tree list generation database.

A traceback follows.  The name of the highest level module is first.
TGADD

=======================================================================

PROMPT> tgunlock -path ../../../../MyData tgdb1r00
The tree list generation database is now unlocked.
```

Figure 3.27: The tree list generation database `tgdb1r00` is initially *locked* to prevent the accidental addition of data. `TGADD` is used to try and add the stand measurement file `untrt_mf.dat`, which fails, reporting that the database is locked. We then unlock the tree list generation database using `TGUNLOCK` so that we may add the two sample stand measurement files, see Figure 3.28.

```
PROMPT> tgadd -path ../../../../MyData -mf untrt_mf.dat tgdb1r00
Processing measurement file: untrt_mf.dat

PROMPT> tgadd -path ../../../../MyData -mf thin_mf.dat tgdb1r00
Processing measurement file: thin_mf.dat
```

Figure 3.28: Add individual stand measurement files to a tree list generation database. The sample untreated and thinned sample stand measurement files `untrt_mf.dat` and `thin_mf.dat` are added separately to the tree list generation database `tgdb1r00`, using `TGADD`.

### *3.2.2 Adding multiple stand measurement file*

The TGADD program is used to add multiple stand measurement files to a tree list generation database using a listing file. When adding multiple stand measurement files it is imperative that the files be correctly formatted with valid content in order for TGADD to successfully process them. TGADD when used with a listing file works seamlessly with stand measurement files having different treatment types, so long as the treatment types are valid and available in the version of the tree list generation database being used, that is, so long as the treatment types used are supported. This means that a single listing file may contain the names of stand measurement files for any supported tree list generation database treatment.

In this example, we add the two sample stand measurement files untrt_mf.dat and thin_mf.dat to the tree list generation database tgdb1r00. Before adding any files to the tree list generation database tgdb1r00 we must first *unlock* it using the program TGUNLOCK, see the next section, Section 3.3. The tree list generation database tgdb1r00 is initially locked to prevent the accidental addition of stand measurement data. Figure 3.29 demonstrates adding multiple stand measurement files to a tree list generation database using TGADD, presenting the command line used and the resulting interactive output. The listing file contains the names of the two sample stand measurement files in alphabetical order, one file name per line.

There is no directly visible change to the tree list generation database tgdb1r00 caused by the addition of stand measurement files, except that the sizes of the files comprising the database may change. The program TGSUMRY, see Section 4.1, may be used to create a before-and-after picture of the contents of a tree list generation database. The files used for this example are contained in the directory MyHome/MyExamp/using/adding/multiple, and the path to the tree list generation database is ../../../../MyData.

```
PROMPT> tgadd -path ../../../../MyData -lf listfile.dat tgdb1r00
Processing measurement file: thin_mf.dat
Processing measurement file: untrt_mf.dat
```

Figure 3.29: Add multiple stand measurement files to a tree list generation database simultaneously using a listing file. The sample untreated and thinned sample stand measurement files `untrt_mf.dat` and `thin_mf.dat` are added simultaneously, to the tree list generation database `tgdb1r00` using `TGADD` and the listing file `listfile.dat`.

### *3.3 Protecting a tree list generation database*

The tree list generation database has a straightforward locking mechanism to protect the database from accidental additions of stand measurement data. The locking mechanism works like a toggle: the database is either locked or it is unlocked, and using the appropriate program on a tree list generation database changes, or toggles, the state, changing it from locked to unlocked or vice versa. Running `TGLOCK` on a locked database or `TGUNLOCK` on an unlocked database has no effect on the database, but a brief message informing you of the current database state is displayed as in Figure 3.30.

Each time the locked/unlocked state of the tree list generation database is changed two events occur. First, the current date and time are associated with the change in the locked/unlocked state. Second, the modification date of the tree list generation database is changed to the current date and time. This provides a mechanism for distinguishing between additions of new stand measurement data and a change in the locked/unlocked state. If the modification time and the locked/unlocked time are identical, the tree list generation database was just locked. If they are different, then new data have been added.

New stand measurement data may be added to an unlocked tree list generation database, but *not* to a locked tree list generation database. Attempting to add stand measurement data to a locked tree list generation database will result in `TGADD` displaying an error message indicating that the database is locked and that it may not

```
PROMPT> tglock -path ../../../MyData tgdb1r00
   The tree list generation database was already locked.

PROMPT> tgunlock -path ../../../MyData tgdb1r00
   The tree list generation database is now unlocked.

PROMPT> tgunlock -path ../../../MyData tgdb1r00
   The tree list generation database was already unlocked.

PROMPT> tglock -path ../../../MyData tgdb1r00
   The tree list generation database is now locked.
```

Figure 3.30: This example takes the initially locked tree list generation database `tgdb1r00`, tries to lock it with `TGLOCK`, which displays a brief message but has no other effect. `TGUNLOCK` is then used to unlock the database, also displaying a brief message indicating the change in state. `TGUNLOCK` is tried on the previously unlocked database to display its message, but has no other effect. Finally `TGLOCK` is used to relock the database and display its message about the change in state.

be modified until it is unlocked. No modifications are made to the database if this occurs.

The tree list generation database `tgdb1r00` is initially in a *locked* state, implying that no new stand measurement files may be added without first *unlocking* the database. `tgdb1r00` was distributed in a locked state to prevent the accidental addition of data. Figure 3.30 demonstrates the programs `TGLOCK` and `TGUNLOCK`, showing the interactive messages displayed by each program, depending on the locked or unlocked state of a tree list generation database when they are used. There are no files other than the tree list generation database `tgdb1r00` used for these examples, but they were run in the directory `MyHome/MyExamp/using/generate/protect`, and from this directory the path to the tree list generation database is `../../../MyData`.

# Chapter 4

# TREE LIST GENERATION DATABASE UTILITIES

This chapter describes and demonstrates the tree list generation database utility programs. These programs allow you to discover the structure of a tree list generation database, obtain the data coverage for a tree list generation database, create a new tree list generation database, and modify the structure of an existing tree list generation database.

We begin with `TGSUMRY` the utility used to obtain a summary of a tree list generation database. The summary contains information about the structure of the tree list generation database, including the number of treatments available, the types of treatments, and the numbers of source stands, the of histogram bins, the trees, for each treatment. In addition, the summary provides the available stand index parameters for each treatment and the specific stand index parameters used for a specific tree list generation database. `TGSUMRY` may be though of as providing a concise description of a particular tree list generation database.

Next, we describe the program `TGSCATRP`. `TGSCATRP` is used to create *scatter plot files*. These files contain the stand index parameter values for the centers of the multidimensional histogram bins comprising the tree list generation database components associated with each treatment. The scatter plot file is a comma delimited file that may be used with plotting software to obtain an indication of the data coverage for a treatment. `TGSCATRP` may be thought of as providing a detailed summary of a treatment within a tree list generation database.

Finally, the programs `TGNEW` and `TGXPLODE` are described. These programs are used to create a new tree list generation database and to convert an existing tree list

generation database into its component parts. This latter capability is important for two reasons. First, `TGXPLODE` allows the recovery of the data sets used to construct a tree list generation database. Second, when `TGXPLODE` is used in conjunction with `TGNEW`, the structure of a tree list generation database may be modified, by changing the stand index parameters used or the histogram bin widths, *without the loss of any data*.

The examples rely on the installation of the tree list generation database as described in Section 1.3. In particular, we assume a home directory `/MyHome` containing the subdirectories `/MyHome/MyData`, `/MyHome/MyDocs`, `/MyHome/MyExe`, and `/MyHome/MyExamp`. These subdirectories contain the tree list generation database `tgdb1r00` and other miscellaneous files, the tree list generation database documentation, the executable files, and the examples for this tutorial broken down by topic, respectively. The directory `/MyHome/MyExe` is assumed to be on the executable search path to provide easy access to the programs for the examples. We assume that the examples for this chapter, from the `utils` directory that is contained in the file `tutexamp.zip`, have been extracted and placed into the installation directory `/MyHome/MyExamp/utils`.

The examples all stand alone, that is, they are all performed in separate subdirectories within the directory `/MyHome/MyExamp/utils` and they have no overlapping dependencies. The name of the specific subdirectory within `/MyHome/MyExamp/utils` containing each example is mentioned with their respective descriptions. The examples are intended to be performed *live* by typing the appropriate commands exactly as presented in this tutorial. All of the examples have been tested in exactly this manner, so they are known to work.

### *4.1 Summarizing a tree list generation database*

The TGSUMRY program is used to obtain a concise summary of the contents and structure of a tree list generation database. The summary of a tree list generation database produced by TGSUMRY includes information describing the number of available treatments, the stand index parameters and their attributes used for indexing each treatment, the locked or unlocked status, and the date and time of last modification. The summary provides a convenient means by which the characteristics of a tree list generation database may be determined. The stand index parameter information from a tree list generation database summary allows the straightforward identification of the minimal set of stand attributes for a particular treatment that must be defined in a stand description file to generate a simulate stand or tree list.

In this example, we use TGSUMRY to obtain a summary of the tree list generation database tgdb1r00. Figure 4.1 demonstrates the use of TGSUMRY to obtain a summary of the tree list generation database tgdb1r00, presenting the command lines used and the resulting interactive output. Figure 4.2 presents the summary generated by the TGSUMRY program and placed into the file sumry.dat by the example.

There are no changes made to the tree list generation database tgdb1r00 by TGSUMRY. There are no files used for this example, other than the tree list generation database tgdb1r00, but an output file sumry.dat is created by the example. The example was performed in the directory MyHome/MyExamp/utils/summary, and the path to the tree list generation database is from this directory is ../../../MyData.

The next section provides a means for obtaining a more detailed summary of a tree list generation database. The summary consists of a list of the multidimensional histogram bin center values. This information may be sued to obtain an idea of the data coverage contained within a tree list generation database.

```
PROMPT> tgsumry -path ../../../mydata -file sumry.dat tgdb1r00
Writing information to file: sumry.dat
```

Figure 4.1: Summarizing a tree list generation database. The TGSUMRY program is used to obtain a concise summary of the tree list generation database tgdb1r00. The summary information is saved to the file sumry.dat by using the TGSUMRY option -file, otherwise the summary information would have been displayed on the terminal screen. Figure 4.2 presents the contents of the output file sumry.dat.

## *4.2 Determining the data coverage in a tree list generation database*

At times it will be necessary to obtain a detailed summary of the data coverage contained within a tree list generation database, for example, to identify gaps in the coverage for particular stand conditions or treatments. The program TGSCATRP provides the means for extracting a detailed summary of the data coverage from a tree list generation database. The data coverage summary produced by TGSCATRP may, therefore, be used as an additional tool to aid in determining the data collection needs for a particular forested region.

The detailed summary of the data coverage created by TGSCATRP, called a *scatter plot file*, is a comma delimited text file containing the list of values for the multi-dimensional histogram bin centers comprising the stand index for a particular tree list generation database. The files produced by TGSCATRP may be used with other software to produce scatter plots of the histogram bin dimensions to display the data coverage.

In this example, we use TGSCATRP to create scatter plot files for the untreated and thinned treatments contained in the tree list generation database tgdb1r00. Figure 4.3 demonstrates the use of TGSCATRP, presenting the command lines used and the resulting interactive output. Two scatter plot files are created by this example, u_scatrp.dat and t_scatrp.dat, for the untreated stands and thinned stands, respectively.

```
Tree list generation database summary.

Summary created on:  08/11/2001 at 15:45:49.38

Tree list generation database path: ..\..\..\mydata
Tree list generation database name: tgdb1r00

Tree list generation database created on : 06/27/2000 at 17:17:27.82
Tree list generation database modified on: 06/28/2000 at 13:06:12.85

Tree list generation database lock status     : locked
Tree list generation database lock/unlock date: 06/28/2000 at 13:06:12.85

The units used in this summary are metric.

Number of treatments: 2

Treatment : UNTREATED

   Number of index parameters  : 7

   Number of tree data buckets      : 3139
   Number of source files           : 5209
   Number of tree records           : 573036

   Index parameters and parameter types:

   Parameter Name                   Used    Type      Width
   -------------------------------- ------- --------- ----------
   TREATMENT_TYPE                   YES          N/A         N/A
   SITE_INDEX_50                    YES     INTERVAL        3.00
   STAND_TOTAL_AGE                  YES     INTERVAL        4.00
   STAND_ORIGIN                     YES        FIXED         N/A
   MEAN_DBH                         NO      INTERVAL        4.00
   MEAN_HEIGHT                      NO      INTERVAL        3.00
   QMD                              YES     INTERVAL        4.00
   TOP_HEIGHT                       NO      INTERVAL        3.00
   STAND_DENSITY                    YES     INTERVAL      200.00
   STAND_BASAL_AREA                 NO      INTERVAL        1.50
   PCT_DF_STEMS                     NO      INTERVAL       20.00
   PCT_WH_STEMS                     NO      INTERVAL       20.00
   PCT_OT_STEMS                     NO      INTERVAL       20.00
   PCT_DF_BASAL_AREA                NO      INTERVAL       20.00
   PCT_WH_BASAL_AREA                NO      INTERVAL       20.00
   PCT_OT_BASAL_AREA                NO      INTERVAL       20.00
   STAND_TYPE                       YES        FIXED         N/A
   NUMBER_OF_SPECIES                NO      INTERVAL        5.00
```

Figure 4.2: `TGSUMRY` output generated for the tree list generation database `tgdb1r00` by the example in Figure 4.1 that created the file `sumry.dat`.

```
Treatment : THINNED

   Number of index parameters  : 12

   Number of tree data buckets      : 3716
   Number of source files           : 4440
   Number of tree records           : 237245

   Index parameters and parameter types:

   Parameter Name                   Used    Type      Width
   -------------------------------- ------- --------- ----------
   TREATMENT_TYPE                   YES          N/A         N/A
   SITE_INDEX_50                    YES     INTERVAL        3.00
   STAND_TOTAL_AGE                  YES     INTERVAL        4.00
   STAND_ORIGIN                     YES        FIXED         N/A
   NUMBER_OF_THINNINGS              YES        FIXED         N/A
   PRE_THIN_DENSITY                 YES     INTERVAL      200.00
   PRE_THIN_BASAL_AREA              YES     INTERVAL        1.50
   PCT_STEMS_REMOVED                NO      INTERVAL       10.00
   PCT_BASAL_AREA_REMOVED           YES     INTERVAL       10.00
   YEARS_SINCE_TREATMENT            YES     INTERVAL        4.00
   MEAN_DBH                         NO      INTERVAL        4.00
   MEAN_HEIGHT                      NO      INTERVAL        3.00
   QMD                              YES     INTERVAL        4.00
   TOP_HEIGHT                       NO      INTERVAL        3.00
   STAND_DENSITY                    YES     INTERVAL      200.00
   STAND_BASAL_AREA                 NO      INTERVAL        1.50
   PCT_DF_STEMS                     NO      INTERVAL       20.00
   PCT_WH_STEMS                     NO      INTERVAL       20.00
   PCT_OT_STEMS                     NO      INTERVAL       20.00
   PCT_DF_BASAL_AREA                NO      INTERVAL       20.00
   PCT_WH_BASAL_AREA                NO      INTERVAL       20.00
   PCT_OT_BASAL_AREA                NO      INTERVAL       20.00
   STAND_TYPE                       YES        FIXED         N/A
   NUMBER_OF_SPECIES                NO      INTERVAL        4.00
```

Figure 4.2: (continued)

```
PROMPT> tgscatrp -path ../../../MyData -treatment untreated ...
   -file u_scatrp.dat tgdb1r00
Writing scatter plot data to file: u_scatrp.dat

PROMPT> tgscatrp -path ../../../MyData -treatment thinned ...
   -file t_scatrp.dat tgdb1r00
Writing scatter plot data to file: t_scatrp.dat
```

Figure 4.3: Obtaining the data coverage for a tree list generation database. The TGSCATRP program is used to obtain a detailed summary of the data coverage in the tree list generation database tgdb1r00. The summary information is saved to a file for each treatment using the TGSCATRP option -file. The data coverage for the untreated stands is saved to the file u_scatrp.dat, and for thinned stands to the file t_scatrp.dat. If the data coverage information were not saved to a file it would have been displayed on the terminal screen for each command line.

TGSCATRP makes no changes to the tree list generation database tgdb1r00, and there are no input files used for this example, other than the tree list generation database tgdb1r00. The two output files u_scatrp.dat and t_scatrp.dat are created in the current directory by the command lines in the example, Figure 4.3. The example was performed in the directory MyHome/MyExamp/utils/summary, and the path to the tree list generation database is from this directory is ../../../MyData.

Figure 4.4 through Figure 4.7 present a straightforward summary of the data coverage for untreated stands contained in the tree list generation database tgdb1r00. The data used to generate the figures was obtained from the scatter plot file u_scatrp.dat created by the example in Figure 4.3. Figure 4.4 presents the percentages of the multidimensional histogram bin centers, or the stand index, by stand type and stand origin. The stand types are PURE_DF, PURE_WH, DF_DOMINANT, WH_DOMINANT, and MIXTURE. These stand types are based on a basal area criterion: stands having at least 80% of their basal area in Douglas-fir or western hemlock are considered pure for that species, stands having at least 50% and less than 80% of their basal area in Douglas-fir or western hemlock are considered to be dominant for that species, and stands having less than 50% of their basal area in Douglas-fir or western hemlock are consid-

ered to be mixtures. Stand origins are either NATURAL or PLANTED. The stand origin NATURAL is comprised of stands with unknown origin, which were presumed to be natural regeneration, and stands known to have been naturally regenerated.

Figure 4.4 clearly indicates that pure Douglas-fir stands are the most abundant stand type in the tree list generation database tgdb1r00, comprising approximately 60% of the multidimensional histogram bin centers in the stand index. Pure western hemlock and Douglas-fir dominant stands follow, both comprising approximately 15% of the stand index. Western hemlock dominant stands and mixed stands comprise the rest of the stand index, splitting the remaining 10% of the stand index. This composition is representative of the commercial stand compositions found in the Pacific Northwest, and is indicative of the industrial forestry nature of the data used to construct the tree list generation database tgdb1r00.

Figure 4.4 also indicates that naturally regenerated stands are the most abundant in the tree list generation database tgdb1r00, comprising 80% of the stand index. There are two reasons for this. First, there is a historical bias in the data used to construct the tree list generation database tgdb1r00. Planting as a preferred stand regeneration method is a relatively recent phenomenon, natural regeneration being favored historically for economic reasons. Second, many stands in the data used to construct the tree list generation database tgdb1r00 were of unknown stand origin. These stands were thought to be predominantly naturally regenerated, as was the custom at the time, and were therefore classified as natural stands for lack of a better alternative. As new stands are added to the tree list generation database tgdb1r00 the distribution of stand types should begin to favor planted stands over naturally regenerated stands.

Figure 4.5 through Figure 4.7 present scatter plots derived from the data contained in the scatter plot file u_scatrp.dat produced by the commands executed in Figure 4.3. The figures present, respectively, plots of QMD *vs.* stand age, QMD *vs.* stand density, and site index at 50 years *vs.* stand age for the stand types PURE_DF,

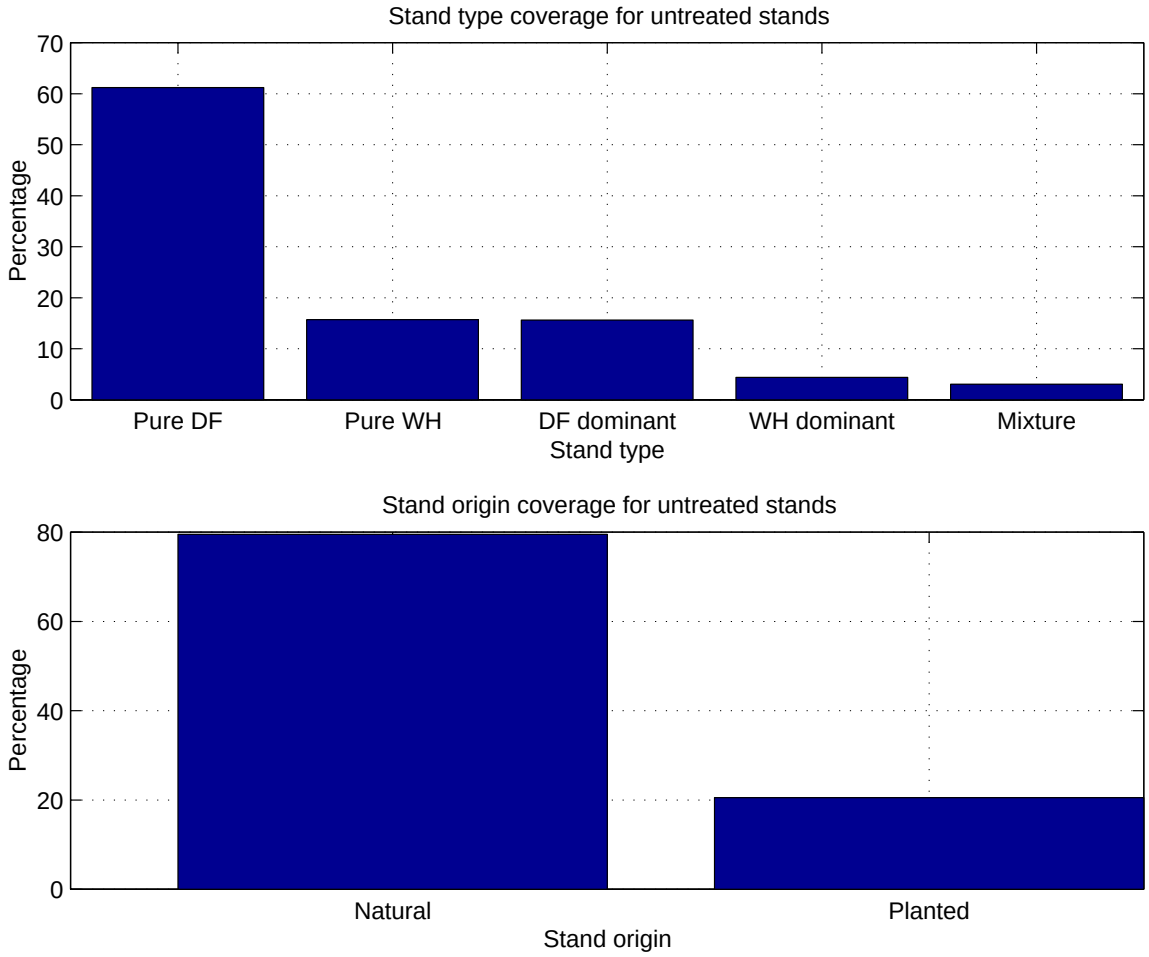


Figure 4.4: Stand type (top) and stand origin (bottom) summary for untreated stands in the tree list generation database `tgdb1r00`. The data used to create this figure were obtained from the scatter plot file `u_scatrp.dat` created by the example in Figure 4.3.

PURE_WH, DF_DOMINANT, WH_DOMINANT for the thinned stands in the tree list generation database tgdb1r00. The stand type MIXTURE was omitted since it comprises a small percentage of the data in the tree list generation database tgdb1r00 used to generate the figures. Notice in each figure that as we progress from the pure Douglas-fir stands, subfigure A, to the western hemlock dominant stands subfigure D, the coverage becomes more sparse and gaps become evident. This is, again, a reflection of the composition of the tree list generation database tgdb1r00 which is predominantly comprised of pure Douglas-fir stands, with smaller components of Douglas-fir dominant stands, pure western hemlock stands, and western hemlock dominant stands.

Figure 4.8 through Figure 4.14 present a straightforward summary of the data coverage for thinned stands contained in the tree list generation database tgdb1r00. The data used to generate the figures was obtained from the scatter plot file t_scatrp.dat created by the example in Figure 4.3. Figure 4.8 presents the percentages of the multidimensional histogram bin centers, or the stand index, by stand type and stand origin. The stand types are PURE_DF, PURE_WH, DF_DOMINANT, WH_DOMINANT, and MIXTURE. These stand types are based on a basal area criterion: stands having at least 80% of their basal area in Douglas-fir or western hemlock are considered pure for that species, stands having at least 50% and less than 80% of their basal area in Douglas-fir or western hemlock are considered to be dominant for that species, and stands having less than 50% of their basal area in Douglas-fir or western hemlock are considered to be mixtures. Stand origins are either NATURAL or PLANTED. The stand origin NATURAL is comprised of stands with unknown origin, which were presumed to be natural regeneration, and stands known to have been naturally regenerated.

Figure 4.9 through Figure 4.11 present scatter plots derived from the data contained in the scatter plot file t_scatrp.dat produced by the commands executed in Figure 4.3. The figures present, respectively, plots of QMD *vs.* stand age, QMD *vs.* stand density, and site index at 50 years *vs.* stand age for the stand types PURE_DF, PURE_WH, DF_DOMINANT, WH_DOMINANTfor the thinned stands in the tree list generation

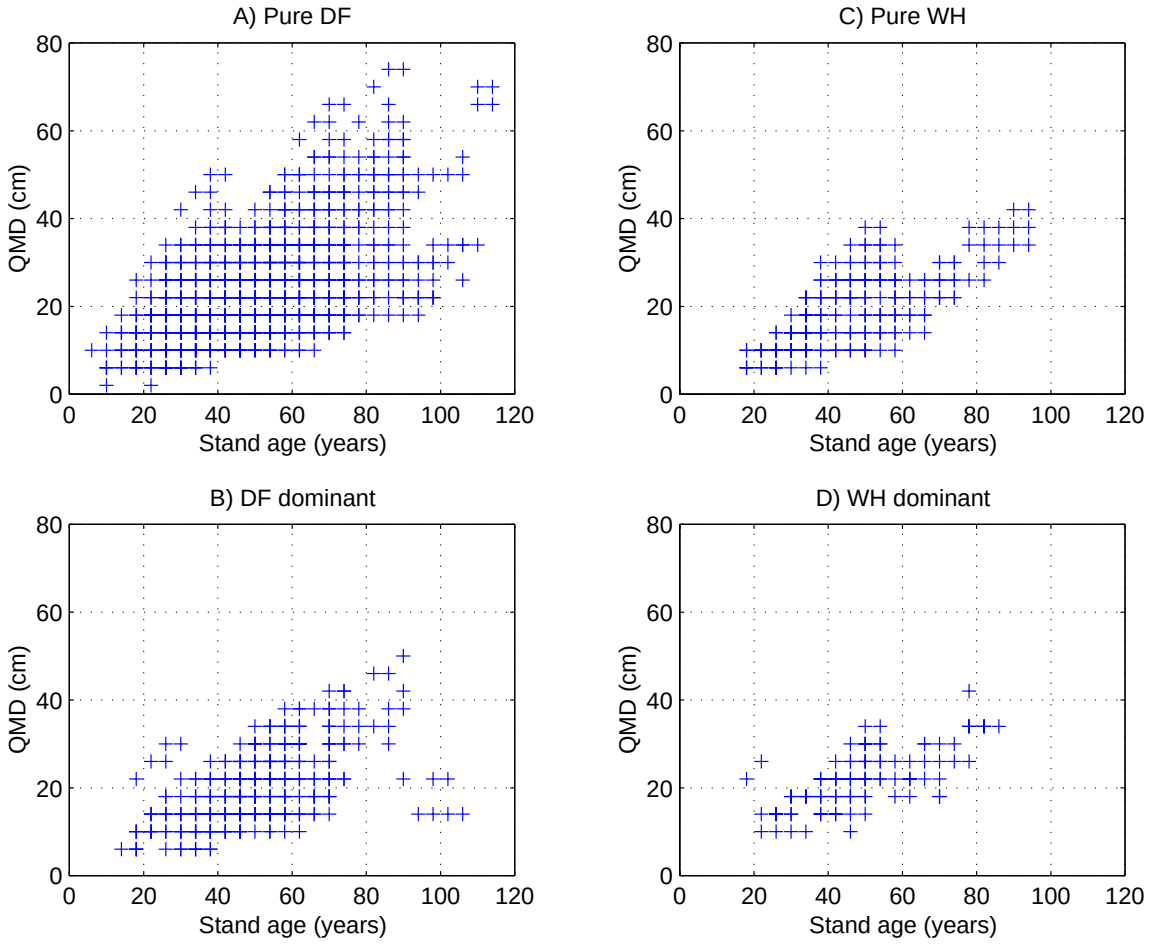


Figure 4.5: QMD *vs.* stand age data coverage by stand type for untreated stands and the tree list generation database `tgdb1r00`. The plots are for `PURE_DF` (A), `DF_DOMINANT` (B), `PURE_WH` (C), and `WH_DOMINANT` (D). The stand type `MIXTURE` is not shown since it comprises a small percentage of the tree list generation database `tgdb1r00`.

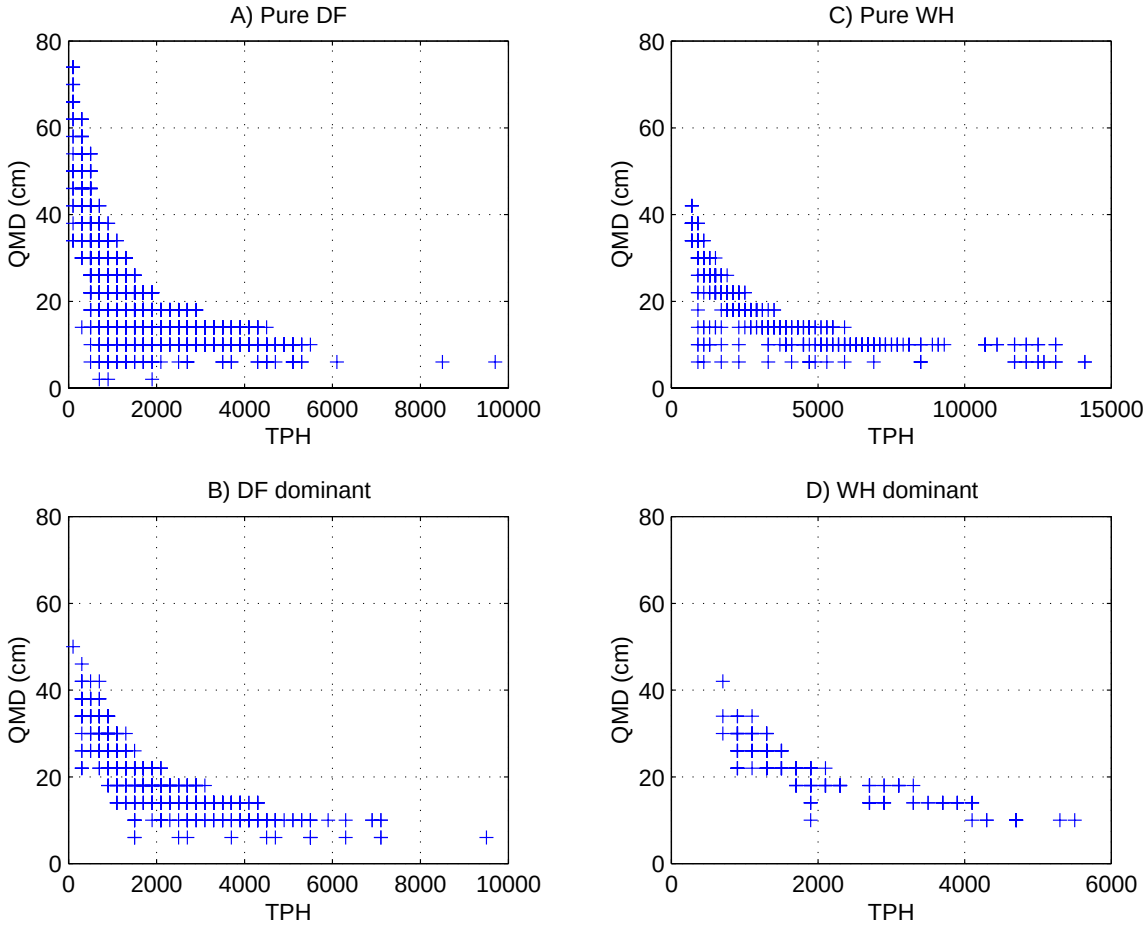


Figure 4.6: QMD *vs.* stand density data coverage by stand type for untreated stands and the tree list generation database `tgdb1r00`. The plots are for `PURE_DF` (A), `DF_DOMINANT` (B), `PURE_WH` (C), and `WH_DOMINANT` (D). The stand type `MIXTURE` is not shown since it comprises a small percentage of the tree list generation database `tgdb1r00`.

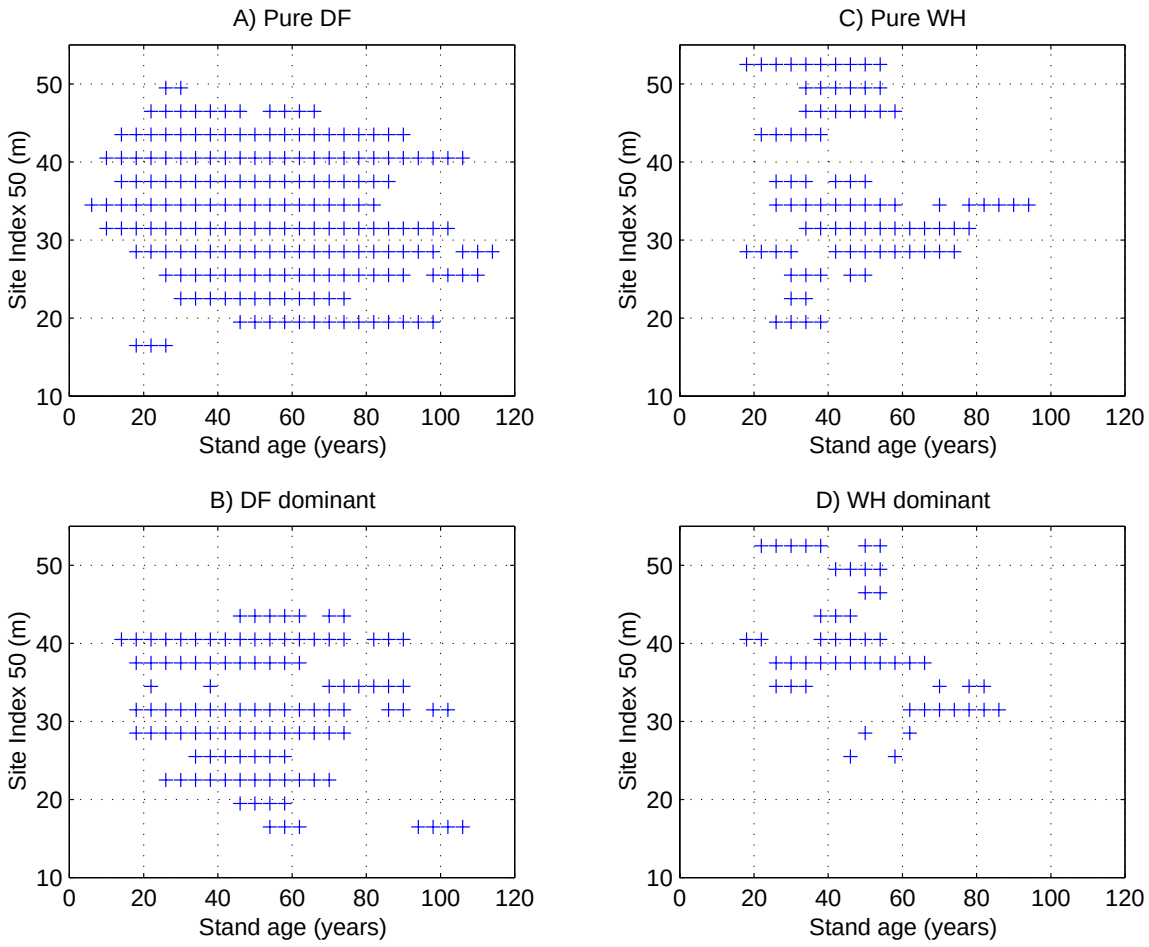


Figure 4.7: Site index *vs.* stand age data coverage by stand type for untreated stands and the tree list generation database `tgdb1r00`. The plots are for `PURE_DF` (A), `DF_DOMINANT` (B), `PURE_WH` (C), and `WH_DOMINANT` (D). The stand type `MIXTURE` is not shown since it comprises a small percentage of the tree list generation database `tgdb1r00`.

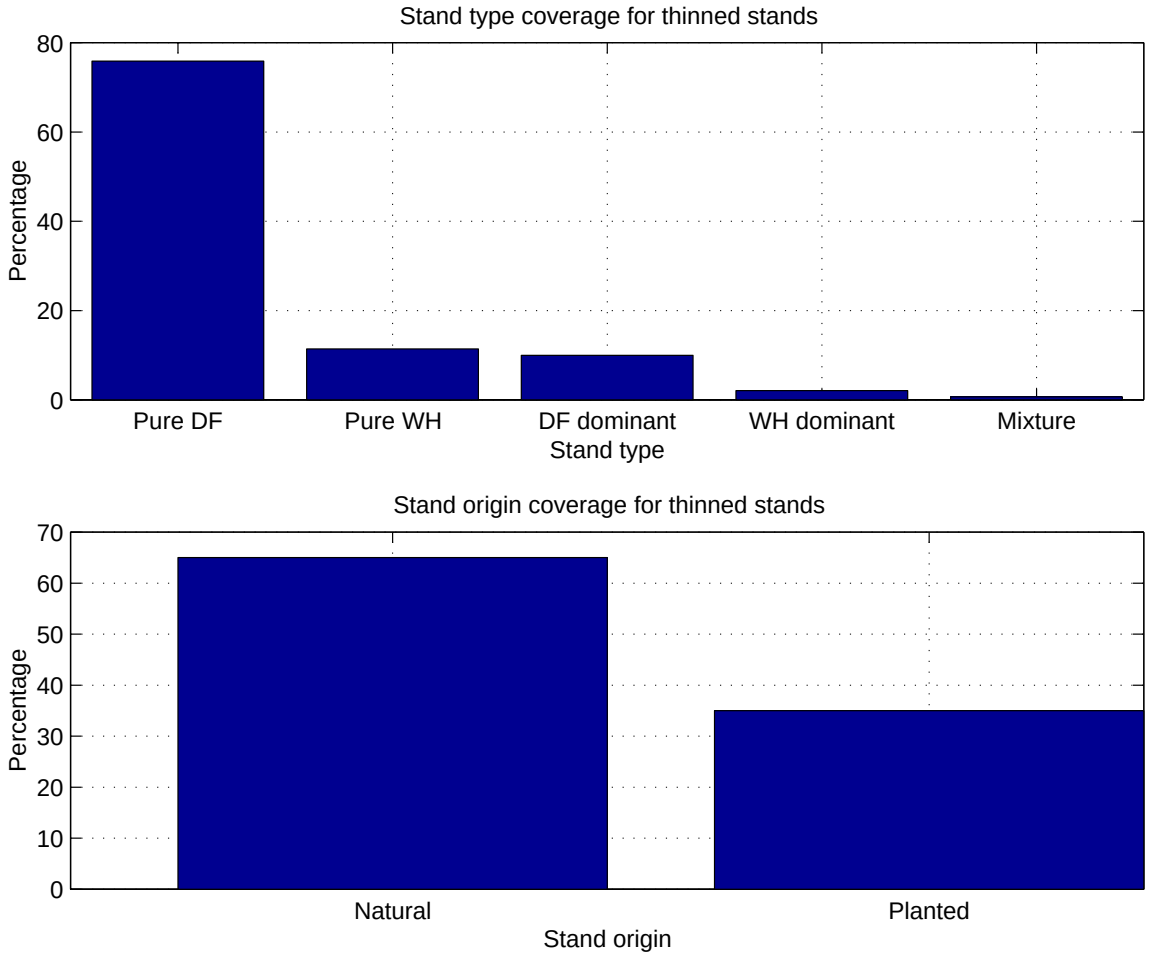


Figure 4.8: Stand type (top) and stand origin (bottom) summary for thinned stands in the tree list generation database `tgdb1r00`. The data used to create this figure were obtained from the scatter plot file `t_scatrp.dat` created by the example in Figure 4.3.

database tgdb1r00. The stand type MIXTURE was, again, omitted since it comprises a small percentage of the data in the tree list generation database tgdb1r00 used to generate the figures. Notice in each figure that as we progress from the pure Douglas-fir stands, subfigure A, to the western hemlock dominant stands subfigure D, the coverage becomes more sparse and gaps become evident. This is, again, a reflection of the composition of the tree list generation database tgdb1r00 which is predominantly comprised of pure Douglas-fir stands, with smaller components of Douglas-fir dominant stands, pure western hemlock stands, and western hemlock dominant stands. These data coverage patterns are identical to those observed for the untreated stands.

Figure 4.12 through Figure 4.14 present scatter plots derived from the data contained in the scatter plot file t_scatrp.dat produced by the commands executed in Figure 4.3. The figures present, respectively, plots of percent of basal area removed *vs.* pre thin stand density, percent of basal area removed *vs.* stand age, and number of thinnings *vs.* stand age for the stand types PURE_DF, PURE_WH, DF_DOMINANT, WH_DOMINANT for the thinned stands in the tree list generation database tgdb1r00. These figures represent the data coverage for the thinning treatments, or the range of possible thinning treatments for each stand type, represented in the tree list generation database tgdb1r00.

We now know how to summarize a tree list generation database and to determine the data coverage for the treatments within a tree list generation database. Next we demonstrate how a new tree list generation database is created.

## *4.3 Creating a tree list generation database*

A tree list generation database must be created before it can be used. The program TGADD is used to create a new, initially empty tree list generations database. Stand measurement files may be added to a new tree list generation database using TGADD, after which the database may be used to generate simulated stands or tree lists with

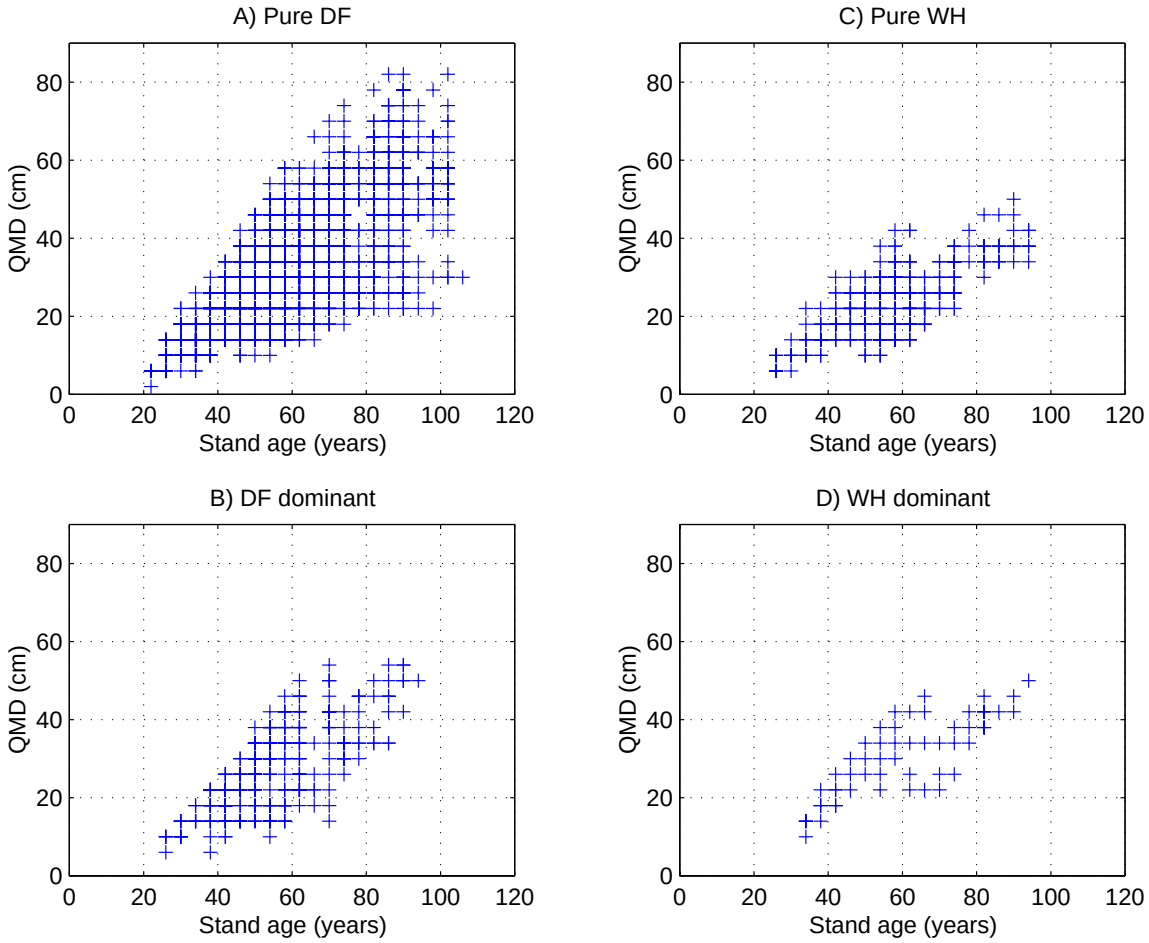


Figure 4.9: QMD *vs.* stand age data coverage by stand type for thinned stands and the tree list generation database `tgdb1r00`. The plots are for `PURE_DF` (A), `DF_DOMINANT` (B), `PURE_WH` (C), and `WH_DOMINANT` (D). The stand type `MIXTURE` is not shown since it comprises a small percentage of the tree list generation database `tgdb1r00`.

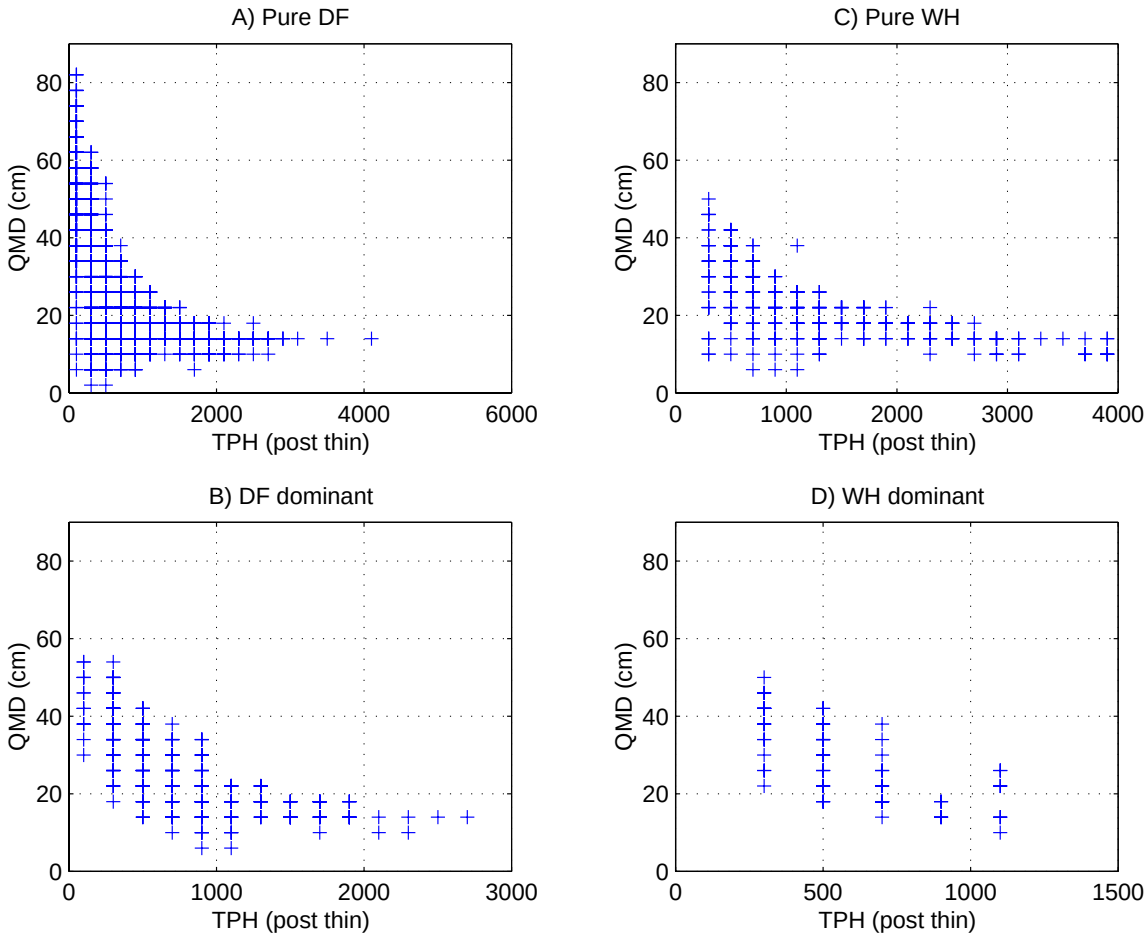


Figure 4.10: QMD *vs.* stand density data coverage by stand type for thinned stands and the tree list generation database `tgdb1r00`. The plots are for `PURE_DF` (A), `DF_DOMINANT` (B), `PURE_WH` (C), and `WH_DOMINANT` (D). The stand type `MIXTURE` is not shown since it comprises a small percentage of the tree list generation database `tgdb1r00`.

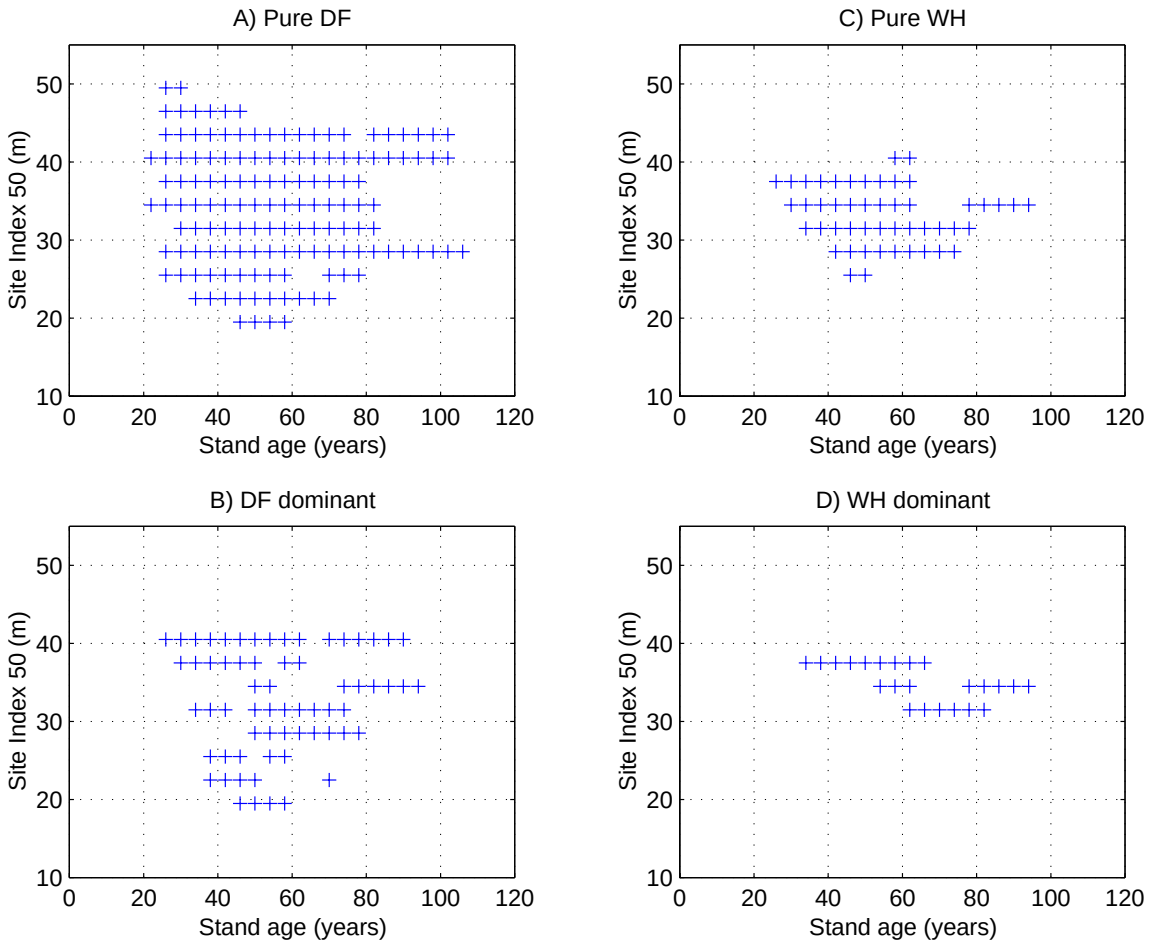


Figure 4.11: Site index *vs.* stand age data coverage by stand type for thinned stands and the tree list generation database `tgdb1r00`. The plots are for `PURE_DF` (A), `DF_DOMINANT` (B), `PURE_WH` (C), and `WH_DOMINANT` (D). The stand type `MIXTURE` is not shown since it comprises a small percentage of the tree list generation database `tgdb1r00`.

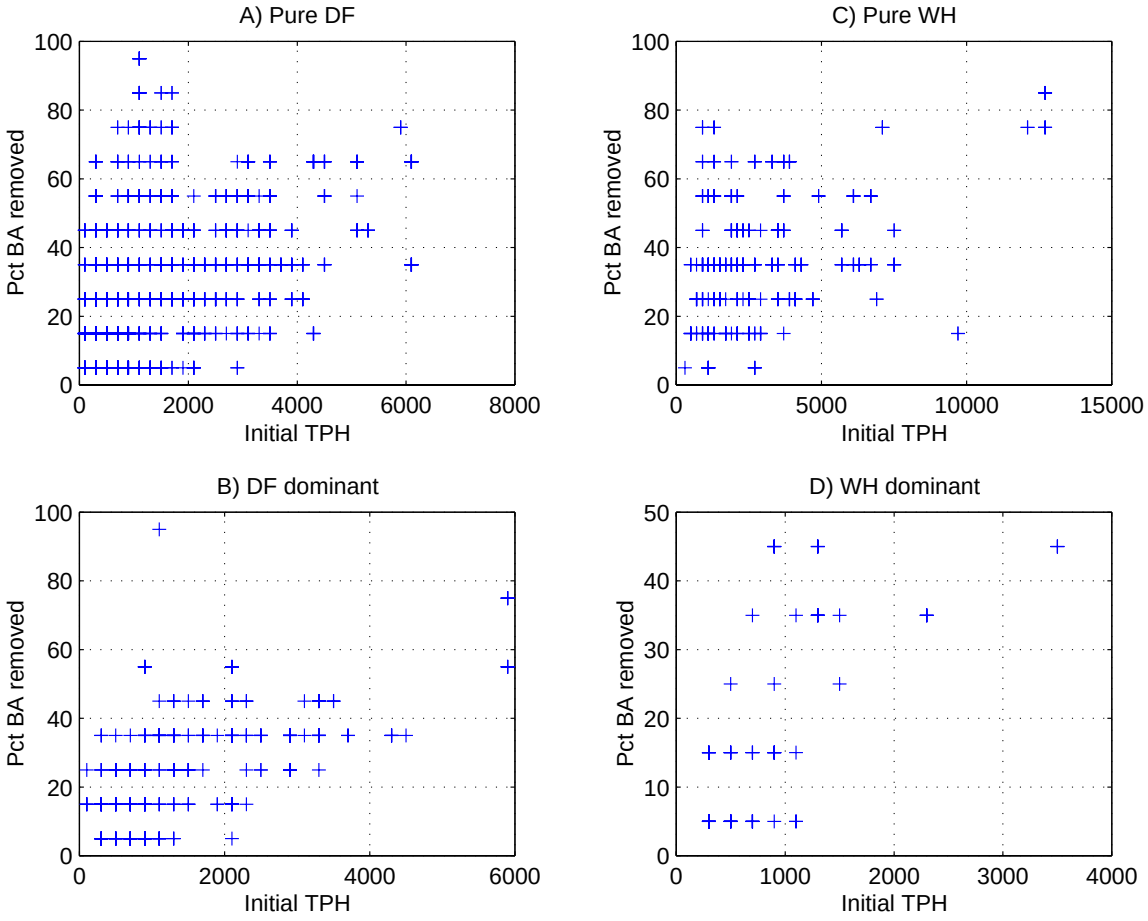


Figure 4.12: Percent of BA removed *vs.* prethin stand density data coverage by stand type for thinned stands and the tree list generation database `tgdb1r00`. The plots are for `PURE_DF` (A), `DF_DOMINANT` (B), `PURE_WH` (C), and `WH_DOMINANT` (D). The stand type `MIXTURE` is not shown since it comprises a small percentage of the tree list generation database `tgdb1r00`.

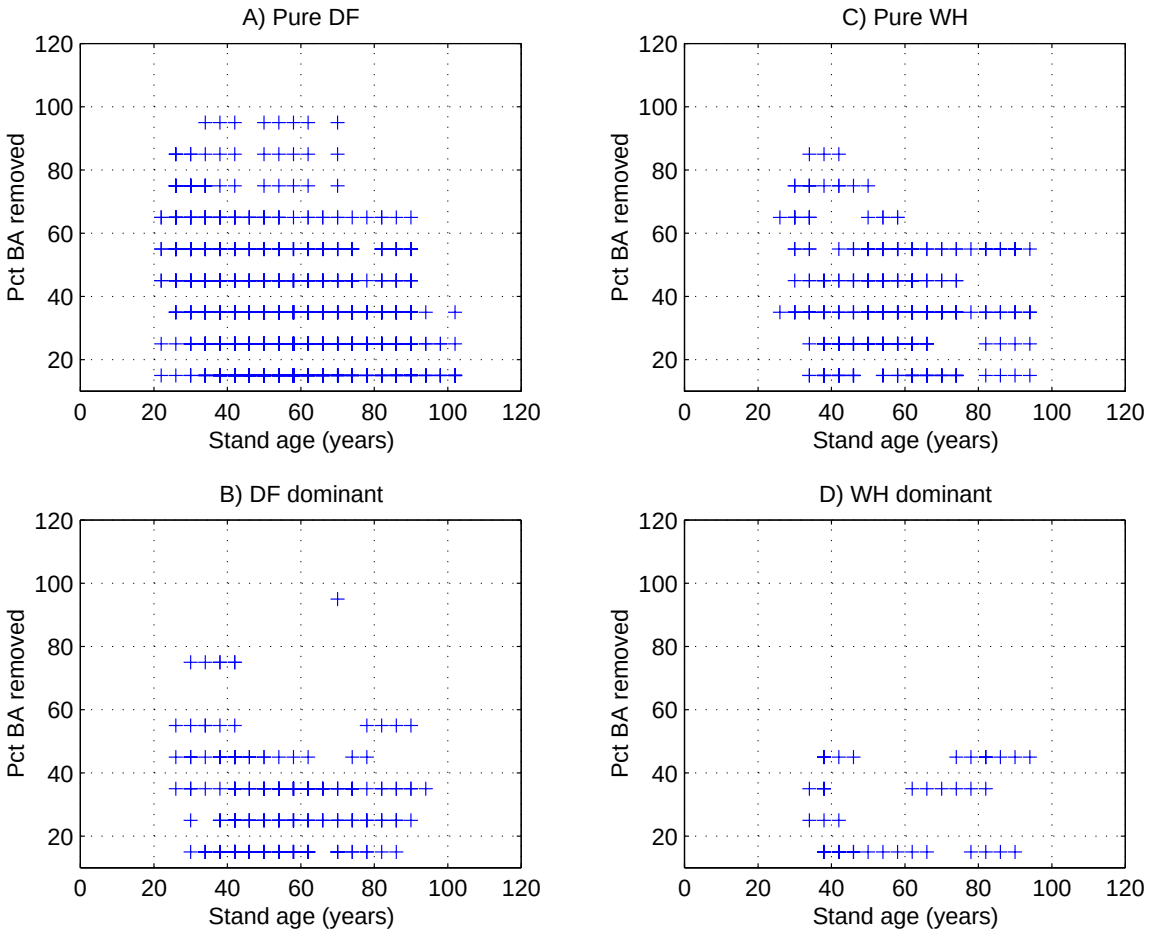


Figure 4.13: Percent of BA removed *vs.* stand age data coverage by stand type for thinned stands and the tree list generation database `tgdb1r00`. The plots are for `PURE_DF` (A), `DF_DOMINANT` (B), `PURE_WH` (C), and `WH_DOMINANT` (D). The stand type `MIXTURE` is not shown since it comprises a small percentage of the tree list generation database `tgdb1r00`.

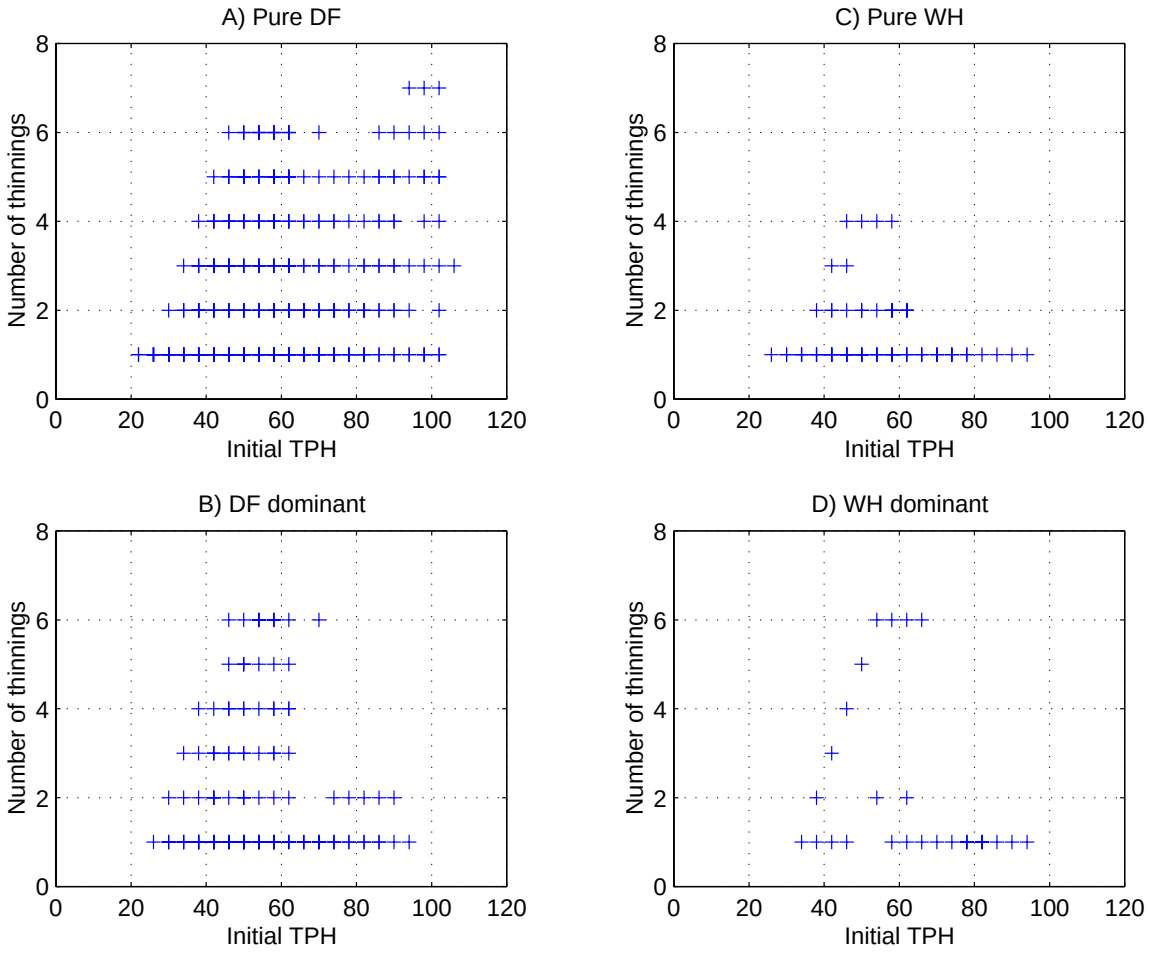


Figure 4.14: Number of thinnings *vs.* stand age data coverage by stand type for thinned stands and the tree list generation database `tgdb1r00`. The plots are for `PURE_DF` (A), `DF_DOMINANT` (B), `PURE_WH` (C), and `WH_DOMINANT` (D). The stand type `MIXTURE` is not shown since it comprises a small percentage of the tree list generation database `tgdb1r00`.

TGRAND. Two files are required to create a tree list generation database, a *schema file* and a *species mapping file*. The schema file defines the structure of the tree list generation database, specifying the treatments, stand index parameters for each treatment, and stand index parameter attributes, including the multidimensional histogram bin widths used. The species mapping file defines a mapping from integer ID codes, used internally by the tree list generation database software, to short and long names for the tree species.

In this example, we create a new tree list generation database, `new_tgdb`, using the sample schema file, `schema.dat`, and the sample species mapping file `spmap.dat` obtained with a tree list generation database distribution. The sample schema file and the sample species mapping file are the files used to create the tree list generation database `tgdb1r00`. We demonstrate two methods for creating a new tree list generation database. The first creates the tree list generation database `new_tgdb` in the current directory, and the second creates a database with the same name in a different directory, specifically the directory `MyHome/Mydata`. In each example the schema file and the species mapping file are assumed to be in the current directory.

Figure 4.15 demonstrates the use of `TGNEW` to create the tree list generation database `new_tgdb` in the current directory, using a schema file and a species mapping file that are also in the current directory. Figure 4.16 demonstrates the use of `TGNEW` to create the tree list generation database `new_tgdb` in the directory `MyHome/MyData`. Both figures present the command lines used and the interactive output produced by `TGADD`. The schema file `schema.dat` and the species mapping file `spmap.dat` need not be in the current directory; full or relative paths to these files may be used as part of the file names. The examples assume that they are in the current directory for convenience.

`TGSUMRY` may be used to obtain a summary of the new tree list generation database `new_tgdb`. The summary will indicate the dat and time that the tree list generation database was created, the treatments and their respective stand index parameters, and

```
PROMPT> tgnew new_tgdb schema.dat spmap.dat
Creating the tree list generation database: new_tgdb
in the current directory.
The tree list generation database was created successfully.
```

Figure 4.15: Creating a tree list generation database in the current directory. A new tree list generation database, `new_tgdb`, is created in the current directory using the sample schema file `schema.dat` and the sample species mapping file `spmap.dat`.

```
PROMPT> tgnew -path ../../../MyData new_tgdb schema.dat spmap.dat
Creating the tree list generation database: new_tgdb
in the directory:  ..\..\..\MyData.
The tree list generation database was created successfully.
```

Figure 4.16: Creating a tree list generation database in a different directory. A new tree list generation database, `new_tgdb`, is created in a different directory using the sample schema file `schema.dat` and the sample species mapping file `spmap.dat` contained in the current directory. The new tree list generation database is created in the directory `MyHome/Mydata`.

that there are no stand measurement files, histogram bins, or trees in the database. Once a tree list generation database is created `TGADD` may be used to add stand measurement files, after which `TGRAND` may be used to generate simulated stands or tree lists using the new database.

The sample schema file and the sample species mapping file used in this example are contained in the directory `MyHome/MyExamp/utils/creating`, which is also the directory where the examples were performed. The path to the directory `MyHome/Mydata` from this directory is `../../../MyData`.

Having seen how a tree list generation database is created using `TGNEW` and `TGADD`, we next consider the problem of modifying the structure of a tree list generation database. A modification may be necessary in two circumstances. First, if different integer ID codes are desired for the tree species. Second, if the set of stand index parameters, or the characteristics of the current stand index parameters, are to be changed. In the next section we describe the procedures for modifying the structure

of a tree list generation database.

### *4.4 Modifying a tree list generation database*

The structure and tree species definitions for a tree list generation database may be modified. There are two sets of circumstances where the modification of a tree list generation database is warranted. First, it may be desirable or beneficial to use a different set of stand index parameters, or have different index parameter characteristics for the current set of stand index parameters, in a tree list generation database. For example, the existing database may use QMD as a stand index parameter and you want to use mean DBH, or you want 2 cm QMD classes rather than the 4 cm QMD classes in `tgdb1r00`. Second, if a tree list generation database is to be coupled with a growth and yield simulator, it may be desirable to have the species codes, in particular the integer ID codes, used in the growth and yield simulator also used for the tree species within a tree list generation database. The tree list generation database uses short, two to six characters, names for the tree species as well as integer ID codes, so it may not be necessary to modify the species mapping. The issues is addressed for completeness. In the remainder of this section we demonstrate how an existing tree list generation database may be modified using `TGXPLODE`, a text editor, `TGNEW`, and `TGADD`.

To modify a tree list generation database four programs must be used; a text editor and three of the tree list generation database programs. Two of the tree list generation database programs have already been introduced, `TGNEW` for creating a tree list generation database, and `TGADD` for adding stand measurement files to a tree list generation database. The third tree list generation database program is `TGXPLODE`, and it is used to convert an existing tree list generation database into its component parts, a schema file, a species mapping file, and a set of stand measurement files for each treatment.

To modify a tree list generation database takes four steps. First, convert an existing tree list generation database into its component parts using `TGXPLODE`. Second, edit the extracted schema file, or create a new schema file, to reflect the new tree list generation database structure. If desired, the species mapping file may be modified at this time as well. Third, create a new tree list generation database using `TGNEW` and the new schema file, and species mapping file if it was also modified. Finally, add all of the stand measurement files that were extracted in the first step to the new tree list generation database using `TGADD`. That's all there is to it. This procedure allows the complete reuse of the data contained in an existing tree list generation database when constructing a new database, and demonstrates one of the primary characteristics of the data based tree list generation scheme: ease of data reuse.

In this example, we want to modify the tree list generation database `tgdb1r00` so that it will index each stand measurement file, that is, only one stand will be associated with each multidimensional histogram bin. This would then permit simulated stands to be generated from the true nearest neighbor stands, rather than the nearest histogram bin centers and their associated stands. This is easily accomplished by assigning a very small histogram bin width to the continuous stand index parameters, those having an `INTERVAL` type, in the schema file. A value of 0.0001 was used in this example.

To begin, we extract all of the component files from the tree list generation database `tgdb1r00` into the directory `extract` using `TGXPLODE`. See Figure 4.17 for the command line used and the interactive output. We obtain 4440 thinned stand measurement files and 5209 untreated stand measurement files. Next, we copy the schema file `schema.dat` and the species mapping file `spmap.dat` from the directory `extract` to the current directory. This is done to avoid modifying the original extracted copies in case we make an error. Now, edit the schema file `schema.dat`, changing all of the interval widths, indicated by the `INTERVAL_WIDTH` keyword, to a value of 0.001, and then save the file. At this point, we have a schema file and a species mapping file, and

we create a new tree list generation database `new_tgdb` using `TGNEW`. See Figure 4.18 for the command line used an the interactive output. Finally, we create a listing file containing the names of all of the extracted stand measurement files and use this with `TGADD` to populate the database with the existing stands. See Figure 4.19 for the command line used and the interactive output. The tree list generation database `new_tgdb` may now be used to generate tree lists with `TGRAND`.

The modified schema file and the extracted components of the tree list generation database `tgdb1r00` used or created for this example are contained in the directory `MyHome/MyExamp/utils/modify`, which is also the directory where the examples were performed. The tree list generation database components were first extracted from `tgdb1r00`, and were then used to create a new tree list generation database `new_tgdb` in the same directory by following the steps described above. The path to the directory `MyHome/Mydata` from this directory is `../../../MyData`.

```
PROMPT> tgxplode -path ../../../../MyData tgdb1r00
extract
Exploding the tree list generation database:  tgdb1r00
Located in directory:  ..\..\..\MyData
To the destination directory:  extract

Creating file:  extract\u\00000001.dat
Creating file:  extract\u\00000002.dat
Creating file:  extract\u\00000003.dat
                      .
                      .
                      .
Creating file:  extract\u\00005207.dat
Creating file:  extract\u\00005208.dat
Creating file:  extract\u\00005209.dat
Creating file:  extract\t\00000001.dat
Creating file:  extract\t\00000002.dat
Creating file:  extract\t\00000003.dat
                      .
                      .
                      .
Creating file:  extract\t\00004438.dat
Creating file:  extract\t\00004439.dat
Creating file:  extract\t\00004440.dat
```

Figure 4.17: Extracting the components of a tree list generation database using `TGXPLODE`. The components of the tree list generation database, the schema file, the species mapping file, and the stand measurement files for each treatment, `tgdb1r00` are extracted and placed into the directory `extract`. These files may then be used to construct a new tree list generation database.

```
PROMPT> tgnew new_tgdb schema.dat spmap.dat
Creating the tree list generation database: new_tgdb
in the current directory.
The tree list generation database was created successfully.
```

Figure 4.18: Creating the modified tree list generation database in the current directory using `TGNEW`. A new tree list generation database, `new_tgdb`, is created in the current directory using the modified schema file `schema.dat` and the species mapping file `spmap.dat` that were extracted from the tree list generation database `tgdb1r00`.

```
PROMPT> tgadd -lf filelist.dat new_tgdb
Exploding the tree list generation database:  tgdb1r00
Located in directory:  ..\..\..\MyData
To the destination directory:  extract

Processing measurement file:  extract\t\00000001.dat
Processing measurement file:  extract\t\00000002.dat
Processing measurement file:  extract\t\00000003.dat
                       .
                       .
                       .
Processing measurement file:  extract\t\00004438.dat
Processing measurement file:  extract\t\00004439.dat
Processing measurement file:  extract\t\00004440.dat
Processing measurement file:  extract\u\00000001.dat
Processing measurement file:  extract\u\00000002.dat
Processing measurement file:  extract\u\00000003.dat
                       .
                       .
                       .
Processing measurement file:  extract\u\00005207.dat
Processing measurement file:  extract\u\00005208.dat
Processing measurement file:  extract\u\00005209.dat
```

Figure 4.19: Adding the stand measurement files extracted from the tree list generation database `tgdb1r00` to the new tree list generation database `new_tgdb`. This is the last step in modifying a tree list generation database and reusing the original stand measurement data. The listing file `filelist.dat` contains the names, including the relative paths, to the extracted stand measurement files in the directory `extract`.