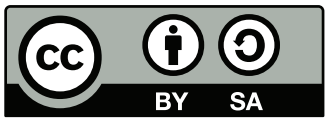
The Sorted Pulse Data Software Library.

Pete Bunting

Documentation and Examples of using SPDLib.

October 27, 2013

Aberystwyth University



Acknowledgements

Authors

Peter Bunting

Dr Pete Bunting joined the Institute of Geography and Earth Sciences (IGES), Aberystwyth University, in September 2004 for his Ph.D. where upon completion in the summer of 2007 he received a lectureship in remote sensing and GIS. Prior to joining the department, Peter received a BEng(Hons) in software engineering from the department of Computer Science at Aberystwyth University. Pete also spent a year working for Landcare Research in New Zealand before rejoining IGES in 2012 as a senior lecturer in remote sensing.

Contact Details

EMail: pfb@aber.ac.uk

Senior Lecturer in Remote Sensing
Institute of Geography and Earth Sciences
Aberystwyth University
Aberystwyth
Ceredigion
SY23 3DB
United Kingdom

Table of Contents

List of Figures

List of Tables

Part I

Background

Chapter 1

Introduction

1.1 SPDLib Version

These notes were created with the following version of SPDLib:

```
spdversion SPDLib 3.0.0, Copyright (C) 2012 Sorted Pulse Library (SPD)
This program comes with ABSOLUTELY NO WARRANTY. This is free software,
and you are welcome to redistribute it under certain conditions; See
website (http://www.spdlib.org). Bugs are to be reported on the trac
or directly to spdlib-develop@lists.sourceforge.net
Mercurial Version: 180
SPD IO Library Version: 3.0.0
SPD Library Version: 3.0.0
spdversion - end
```

1.2 Background

The Sorted Pulse Data (SPD) software library (SPDLib; Bunting et al., 2013a) is a set of open source software tools for processing laser scanning data (i.e., LiDAR), including data captured from airborne and terrestrial platforms and provides an implementation of the SPD file format (Bunting et al., 2013b). The aim of the software is provide a set of tools to process these data. The software has grown out of our needs for LiDAR processing and has continued to grow. One of the

key features which differentiates this software from other LiDAR software is the ability to process and store waveform datasets alongside traditional discrete return data.

1.3 The SPDLib File Format

The software library and tools presented in this paper are built on top of the SPD file format (Bunting et al., 2013b), which has been designed specifically for the storage of LiDAR waveform and discrete return data acquired by TLS, ALS and space borne systems, and includes support for multiple wavelengths within a single file. The format uses a pulse-based structure as opposed to a solely point-based structure, where pulses contain all the information associated with a transmitted pulse from the sensor. This includes transmitted and received waveforms and the discrete returns, determined by Gaussian decomposition of digitised waveforms or by the sensor hardware using proprietary methods. The SPD format also supports 2D spatial indexing of the pulses, where pulses can be referenced using cartesian, spherical or polar coordinate systems and projections. These indexes can be used to significantly speed up data processing whilst allowing the data to be appropriately projected. They are particularly useful when analysing and interpreting TLS data. The format is defined within a HDF5 file, which provides a number of benefits that include broad support across a wide range of platforms and architectures and support for file compression.

For full details see Bunting et al. (2013b).

1.4 The KEA Image File Format

The KEA image format is a new image file format proposed by (Bunting and Gillingham, 2013) and integrated into GDAL as a driver. It the software can be downloaded from `https://bitbucket.org/chchrsc/kealib` and many of the examples which use raster images will use the KEA file format. If you do not have KEA installed then change the format string to HFA for Erdas Imagine (*.img)

files or ENVI for the ENVI file format or GTIFF for a geotiff file. All the image formats supported by your installation of GDAL are supported by SPDLib.

1.5 Comparison to Other Software

This section is an attempted to provide some kind of comparison of the functionality within SPDLib to other popular packages. I am not familiar with either LASTools or FUSION and this material is based on what I've found in the FUSION and LASTools user manuals and websites. If there are errors or software functionality has changed please tell me so I can correct those errors.

Table 1.1: Comparison of the functionality within SPDLib to FUSION and LASTools. (See `http://www.gdal.org/formats_list.html` for list of GDAL supported formats)

Function	FUSION	LASTools	SPDLib
License	BSD	LGPL / Commercial	GPL3 / MIT-X
Source Code	Closed	Mixed Open & Closed	Open
Import LAS	Yes	Yes	Yes
Import ASCII	Yes	Yes	Yes
Raster Formats	ASCII / Binary	ASCII / GTIFF	All GDAL Formats
Subset Point Cloud	Yes		Yes
Canopy Height Metrics	Yes	Yes	Yes
Point Cloud Z Metrics	No	No	Yes
Intensity Metrics	No	No	Yes
Combined Metrics	No	No	Yes
Metrics Calculator	No	No	Yes
Classification of Ground	Yes	Yes	Yes
Merge Point Clouds	Yes		Yes
Interpolate Surfaces	Yes	Yes	Yes
Surfaces Volumes	Yes	No	No
Assign RGB Values	No	No	Yes
Generate Contours	No	Yes	No
Tile Point Cloud	No	Yes	Yes
Mosaic Rasters	Yes	No	Yes
3D Data Viewer	Yes	Yes	Yes
Website Generation	Yes	No	No
Data Indexing	No	Yes	Yes
Pulses Data model	No	No	Yes
Waveform Processing	No	No	Yes

Chapter 2

Installing SPDLib

The notes below hopefully provide some useful details on the process for installing SPDLib. These notes are intended for people compiling the software on a UNIX platform such as Mac OSX, Linux or Solaris (these are the platforms on which the software has been tested). See the end of the chapter for Windows info, Section 2.6.

To compile the software (and the pre-requisites) you will need a C++ compiler, we use the GNU GCC (`http://gcc.gnu.org`) compilers but the software has also been tested and compiles without a problem using the SunPro compiler on Solaris and the Intel x86 compilers.

You will also need to have mercurial (`http://mercurial.selenic.com`) installed to download the latest version of the SPDLib source code, cmake `http://www.cmake.org` to configure the source code before compilation.

2.1 Getting the SPDLib Source Code

The SPDLib source code is hosted within a Mercurial repository on bitbucket (`https://bitbucket.org/petebunting/spdlib`). To clone the source code into a folder spdlib run the following command

```
hg clone https://bitbucket.org/petebunting/spdlib spdlib
```

2.2 Compiling SPDLib

If libraries are not installed within /usr/local then the path needs to be specified using the variables available on CMake listed below.

```
cmake -D CMAKE_INSTALL_PREFIX=/usr/local \
-D HDF5_INCLUDE_DIR=/usr/local/include \
-D HDF5_LIB_PATH=/usr/local/lib \
-D LIBLAS_INCLUDE_DIR=/usr/local/include \
-D LIBLAS_LIB_PATH=/usr/local/lib\
-D GSL_INCLUDE_DIR=/usr/local/include \
-D GSL_LIB_PATH=/usr/local/lib \
-D CGAL_INCLUDE_DIR=/usr/local/include \
-D CGAL_LIB_PATH=/usr/local/lib \
-D BOOST_INCLUDE_DIR=/usr/local/include \
-D BOOST_LIB_PATH=/usr/local/lib \
-D GDAL_INCLUDE_DIR=/usr/local/include \
-D GDAL_LIB_PATH=/usr/local/lib \
-D XERCESC_INCLUDE_DIR=/usr/local/include \
-D XERCESC_LIB_PATH=/usr/local/lib \
-D GMP_INCLUDE_DIR=/usr/local/include \
-D GMP_LIB_PATH=/usr/local/lib \
-D MPFR_INCLUDE_DIR=/usr/local/include \
-D MPFR_LIB_PATH=/usr/local/lib \
-D CMAKE_VERBOSE_MAKEFILE=ON \
.

make

make install
```

2.3 Pre-requisites

The SPDLib software library has a number of software prerequisites, which are required to built the software.

During the development process, to date, the following libraries have been included:

- Boost (`http://www.boost.org`) ≥ Version 1.49
- HDF5 (`http://www.hdfgroup.org`) ≥ Version 1.8.2
- GNU Scientific Library (GSL; `http://www.gnu.org/software/gsl`) ≥ Version 1.14
- Xerces-C (`http://xerces.apache.org/xerces-c`) ≥ Version 3.1.1
- GDAL/OGR (`http://www.gdal.org`) ≥ Version 1.7
- LibLAS (`http://www.liblas.org`) ≥ Version 1.6
- CGAL (`http://www.cgal.org`) ≥ Version 3.8

Boost

The Boost libraries provide a set of software utilities, which form extensions to the C++ standard library (STL). There are alternatives to the individual components of boost, but boost provides a single installation, simplifying the installation process for the user, and the project conducts regular code review ensuring quality. Within SPDLib the integer numeric types definitions, python interface, type casting, file system and text processing components are used.

HDF5

The HDF5 software library, provided by the HDF group, is used to read and write HDF5 files. Using the software library provided by the HDF5 group ensures full compatibility with the HDF5 standard and optimal input / output (I/O) performance. Many systems used for scientific computing have the HDF5 libraries installed but when used with SPDLib the optional C++ libraries needs to build and the zlib library is also required.

GSL

The GSL library is provided under the terms of the GNU GSL3 license and provides numerous functions for mathematical operations as well as C struct representations

for mathematical vectors and matrices. Where possible the mathematical operations required within SPDLib are provided by calling functions within GSL.

Xerces-C

The ability to parser XML is required for the metrics command and the Xerces-C parser is used.

GDAL/OGR

The GDAL/OGR library provides a common interface to read and write image and vector files in the formats commonly used within the remote sensing community. The GDAL library is provided under the MIT-X license and as such has been included within numerous open source and commercial software (e.g., QGIS and ArcMap), it is the only open source software of its type and has a large support community.

LibLAS

When SPDLib was first developed the libLAS software was the only open source library available for reading and writing the LAS file format. It is recommended that LibLAS is built with laszip support to allow compressed LAS files to be read and written.

CGAL

The CGAL library provides support for numerous computation geometry functions and algorithms. Within SPDLib CGAL has been used to generate triangulations and natural neighbour interpolation, used as the default interpolation technique throughout the library. CGAL was used as the project provides excellent documentation (The CGAL Project, 2012) and very good computational performance, important for operations such as interpolation.

2.4 Installing Pre-requisites From Source

2.5 Using the Ubuntu Package Manager

Please note that packages within the manager are always changing and there these notes may not be up to date but provide a guideline and things to look out for.

2.6 Compiling on Windows

There is nothing that should stop you from compiling SPDLib on Windows but we have not tested it so there would probably be some issues which would need working through to make it happen. This is something we are keen to see happen but we do not have any experience in development under Windows or using Visual Studio. If you interested we would happily work with anyone wanting to tackle this issue.

The other options are to install using Cygwin (`http://www.cygwin.com`) or to install Linux through virtualisation. VirtualBox (`https://www.virtualbox.org`) is a free and open source virtualisation package with wide support for multiple platforms, including Windows. The UNIX installation instructions should be followed once you have one of these setup.

Part II

Command Line Tools

Chapter 3

Data Management Commands

3.1 Convert File Formats – spdtranslate

The spdtranslate command is one of the key commands associated with SPDLib as it allows for the conversion between the various supported file formats, while it also supports coordinate system conversion and the definition of the data origin (mainly for TLS data).

Although, the most common use of this tool is converting data to the SPD format. The simplest command for converting to UPD (SPD with a spatial index) is shown below, where the input and output file formats have been specified alongside the field used to attribute each pulse with a location to be used if the pulses where later index (i.e., into an SPD file).

```
spdtranslate --if LAS --of UPD -x FIRST_RETURN -i 44123A1305_ALL.las \
-o 44123A1305_ALL.spd
```

If you wish to explicitly define the projection of the SPD file then use the –output_proj switch to specify a text file containing the OGC WKT string representing the projection.

```
spdtranslate --if LAS --of UPD -x FIRST_RETURN --input_proj ./NZTM2000.wkt \
-i 44123A1305_ALL.las -o 44123A1305_ALL.spd
```

To convert data to the SPD format, where data is indexed on to a 10 m grid the

following command is the simplest form.

```
spdtranslate --if LAS --of SPD -x FIRST_RETURN -b 10 -i 44123A1305_ALL.las \
-o 44123A1305_ALL_10m.spd
```

3.1.1 Memory Requirements

When converting to an UPD very little memory is required as only a few pulses are held in memory at any one time (to be exact it is the number of pulses which corresponds to the UPD/SPD compress block size, see header info but probably 1000 pulses), this is because no sorting of the pulses is required. On the other hand when generating an SPD file the data needs to be spatially sorted, making no assumptions of the order the data is being read (such as it is in time sequential order). Therefore, in the command given above the whole file is read into memory and sorted into the spatial grid before being written output the file. This requires that you have sufficient memory to store the whole dataset and index data structure in memory. If you do not have sufficient memory to complete this operation the file needs to tiled, into blocks which are small enough to fit into memory. Allowing the SPD file to be built in stages, naturally this is slower but once completed it is very fast to make spatial selections within the file and other processing steps (i.e., classification and interpolations) can be applied to the whole file with only a relatively small memory footprint (Figure 3.1).

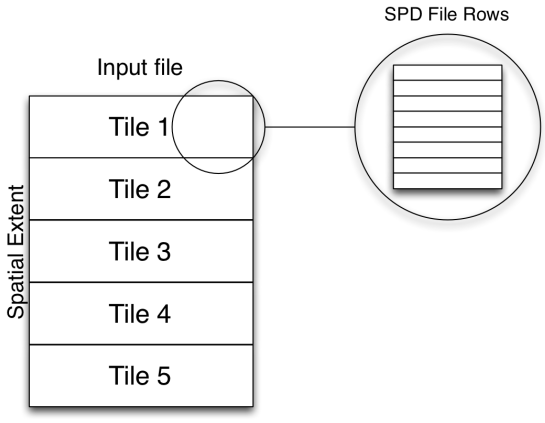


Figure 3.1: spdtranslate tiling.

The option to select tiling the file to disk while building the SPD file is –temppath which is path and base file name while the tiles will be written. Additionally, the option –numofrows may need to be specified (default is 25), this parameter specifies the number of rows of the final SPD file are written to each temporary tile (note the tile height in metres (or where ever units the data is projected) is the binsize * numofrows). Finally, the –numofcols option maybe set where datasets are very wide such that the tiles are not the full width of the output file, by default this is set to 0. Where this option is used the resulting SPD file with a non-sequential rather than a sequential file generated by the other commands. This means the data on disk is not order left-to-right top-to-bottom or top-left to bottom-right, which has some performance benefits. See (Bunting et al., 2013b) for more details.

When running multiple conversions at the same time it is worth noting that the temp file path, or at least the base name, need to be different to avoid over writing the files of the other jobs.

```
spdtranslate --if LAS --of SPD -b 10 -x FIRST_RETURN \
--temppath ./tmp/44123A1305 --numofrows 50 -i 44123A1305_ALL.las \
-o 44123A1305_ALL_10m.spd

spdtranslate --if LAS --of SPD -b 10 -x FIRST_RETURN \
--temppath ./tmp/44123A1305 --numofrows 50 --numofcols 50 \
-i 44123A1305_ALL.las -o 44123A1305_ALL_10m.spd
```

Therefore, when this command runs the following steps will be completed:

1. Convert input (LAS) file to an unsorted SPD (UPD) file so the extent is calculated and pulses built – if the file is already an SPD file this step does not take place.

2. Tile the SPD file into individual unsorted SPD files base on the –numofrows and –numofcols parameters.

3. Spatial sorted (grid) each tile in memory and write to the output SPD file.

3.1.2 Supported File Formats

The file formats supported are the ones which we have so far required for our research therefore the current level of support is not intended to encompass all the available formats but it is relatively easy to add support for new formats within the C++ code so further formats will be added as and when we need them or on request from others. Therefore, if the format you require is not currently supported please let us know and we can look into adding it.

Importers

- SPD
- LAS
- ASCII
- ASCIIPULSEROW
- ASCIIMULTILINE
- FWF_DAT
- DECOMPOSED_DAT
- DECOMPOSED_COO

LAS

The LAS reader is via LibLAS (urlhttp://www.liblas.org) and therefore only supports the discrete return (LAS 1.2) data. A python script is available for reading LAS 1.3 waveform data but this is currently experimental.

ASCII

The ASCII format requires a schema, written in XML, to be supplied. The parse experts a single return per line and the resulting pulses will only contain a single

return.

The following are some examples of schema's for common ASCII formats.

A schema for the PTS format:

```
<?xml version="1.0" encoding="UTF-8" ?>
<line delimiter=" " comment="#" ignorelines="0" >
    <field name="X" type="spd_double" index="0" />
    <field name="Y" type="spd_double" index="1" />
    <field name="Z" type="spd_float" index="2" />
    <field name="AMPLITUDE_RETURN" type="spd_uint" index="3" />
    <field name="RED" type="spd_uint" index="4" />
    <field name="GREEN" type="spd_uint" index="5" />
    <field name="BLUE" type="spd_uint" index="6" />
</line>
```

A schema for the iSite PTS format:

```
<?xml version="1.0" encoding="UTF-8" ?>
<line delimiter=" " comment="#" ignorelines="0" >
    <field name="X" type="spd_double" index="0" />
    <field name="Y" type="spd_double" index="1" />
    <field name="Z" type="spd_float" index="2" />
    <field name="RED" type="spd_uint" index="3" />
    <field name="GREEN" type="spd_uint" index="4" />
    <field name="BLUE" type="spd_uint" index="5" />
    <field name="AMPLITUDE_RETURN" type="spd_uint" index="6" />
</line>
```

A schema for the XYZ format:

```
<?xml version="1.0" encoding="UTF-8" ?>
<line delimiter="," comment="#" ignorelines="0" >
    <field name="X" type="spd_double" index="0" />
    <field name="Y" type="spd_double" index="1" />
    <field name="Z" type="spd_float" index="2" />
</line>
```

A schema for the XYZI format:

```
<?xml version="1.0" encoding="UTF-8" ?>
<line delimiter="," comment="#" ignorelines="0" >
    <field name="X" type="spd_double" index="0" />
```

```
    <field name="Y" type="spd_double" index="1" />
    <field name="Z" type="spd_float" index="2" />
    <field name="AMPLITUDE_RETURN" type="spd_uint" index="3" />
</line>
```

ASCIIPULSEROW

ASCIIMULTILINE

FWF_DAT

DECOMPOSED_DAT

DECOMPOSED_COO

Exporters

SPDLib currently provides 3 formats to which data can be exported:

- SPD
- LAS
- ASCII

LAS

LAS 1.2 files are exported through the libLAS (urlhttp://www.liblas.org) library so as with the importer only discrete return data are supported.

ASCII

The ASCII format depends on the data type (Discrete Return, Decompose Returns or Waveform Data) but the columns are named and match on to the SPD fields.

3.1.3 Examples

Converting Between Coordinate Systems

With all spatial projected data the ability to record the projection and datum and convert between different projections is an important step. To store coordinate system information SPDLib uses the well-known text (WKT) format as defined by the OGC. A large database of WKT strings for coordinate systems worldwide is available from `http://spatialreference.org`. Additionally, the spdproj command can be used to find the OCG WKT string from an existing image, SPD file or to convert a proj4 string.

Within the spdtranslate command the option –input_proj is used to specify the coordinate system of the inputed file and will written to the output file if defined. The input to the –input_proj option is the file path to a text file with the OGC WKT string for the projection being used. Below is the WKT string for UTM Zone 55 South:

```
PROJCS["UTM Zone 55, Southern Hemisphere",GEOGCS["WGS 84",DATUM[
"WGS_1984",SPHEROID["WGS 84",6378137,298.257223563,AUTHORITY[
"EPSG","7030"]],TOWGS84[0,0,0,0,0,0,0],AUTHORITY["EPSG","6326"]],PRIMEM[
"Greenwich",0,AUTHORITY["EPSG","8901"]],UNIT["degree",0.0174532925199433,
AUTHORITY["EPSG","9108"]],AUTHORITY["EPSG","4326"]],PROJECTION[
"Transverse_Mercator"],PARAMETER["latitude_of_origin",0],PARAMETER[
"central_meridian",147],PARAMETER["scale_factor",0.9996],PARAMETER[
"false_easting",500000],PARAMETER["false_northing",10000000],UNIT["Meter",1]]
```

To convert the coordinate system the –convert_proj and –output_proj options need to be used where the –output_proj option requires the path to another OGC WKT file, specifying the output coordinate system. When converting between coordinate systems the GDAL OGR library (urlhttp://www.gdal.org/ogr) is used, which if available makes use of the proj4 library (urlhttp://trac.osgeo.org/proj/; it is recommend that GDAL/OGR are built with this library for coordinate conversion) and therefore the restrictions, in terms of coordinate systems support, of these libraries apply. In particular, if you wish to convert datums you need to ensure you have the appropriate datum shift files installed within proj4. Although, the spdelevation command can be used if a datum shift grid is available as an image

or can be converted to an image.

The following command provides an example where the coordinate system (UK Ordnance Survey national grid) has been specified when converting data to an SPD file.

```
spdtranslate --if ASCIIMULTILINE --of UPD --input_proj osgb36.wkt \
-x FIRST_RETURN -i LDR-GB08_12-2009152a09.txt -o LDR-GB08_12-2009152a09.spd
```

while the following command converts from lat/long to UTM 55s while reading the file and converting to SPD.

```
spdtranslate --if DECOMPOSED_DAT --of SPD --convert_proj --input_proj GDA94_Deg.wkt \
--output_proj GDA94_UTM55.wkt -xFIRST_RETURN -b 1 -i p137_ahd.dat \
-o p137_ahd_UTM_1m.spd
```

Defining the Top-Left corner of the index

Another option when creating an SPD is to define the top-left corner of the SPD index, which can be used to put the LiDAR dataset onto the same grid as another dataset (e.g., image or another LiDAR scene), useful for change detection or fusing datasets. spdtranslate provides three options to enable this, –defineTL, –tlx and –tly. Where –tlx and –tly take double values specifying the coordinate, an example is given below:

```
spdtranslate --if UPD --of SPD -b 25 --defineTL --tlx 534700 --tly 7203900 \
--temppath ./tmp/ --numofrows 200 -i apr2dc_injunePSUs_20090412_AHD.spd \
-o apr2dc_injunePSUs_20090412_AHD_25m.spd
```

Indexing using Spherical Coordinate System

For airborne data indexing the pulses using a cartesian coordinate system is highly appropriate and suits almost all types of processing which might be required. For some terrestrial laser scanning data, acquired within a spherical coordinate system (Figure 3.2) it is more appropriate to store and process the data in this form, for example to monitor change (i.e., movement).

To index with a spherical coordinate system the –spherical option just needs to be

Figure 3.2: An example of a terrestrial laser scanning scene indexed using spherical coordinate system; the overview image from the SPD Points Viewer.

specified while if the origin of the dataset has not be specified the –defineOrigin, –Ox, –Oy and –Oz options are required to specify origin from which the azimuth, zenith and range values for each return and pulse can be calculated. An example of this is given below, note the binsize is specified in radians for a spherical index.

```
spdtranslate --if LAS --of SPD --input_proj utm55s_wgs84.wkt --defineOrigin\
--Ox 562197.7 --Oy 7183563.8 --Oz 593.5  --spherical --binsize 0.005 \
-i 58_forest_registered.las -o 58_forest_registered_spherical.spd
```

Indexing into a Hemispherical Image

Alternatively, it can also be useful to index the dataset into a hemisphere (Figure 3.3). The options for this are also identical to those outlined above when indexing for an unwrapped spherical coordinates, as shown below. Note the binsize is specified in radians for a hemispherical index.

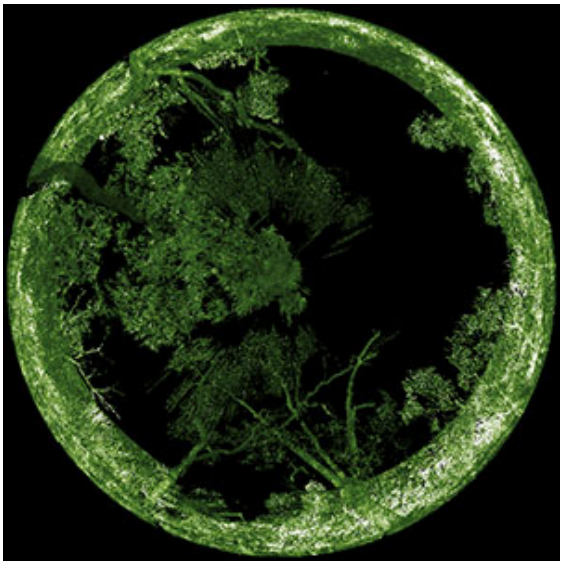


Figure 3.3: An example of a terrestrial laser scanning scene indexed into a hemisphere; the image shown is the mean intensity as generated using spdmetrics.

```
spdtranslate --if LAS --of SPD --input_proj utm55s_wgs84.wkt --defineOrigin \
--Ox 562197.7 --Oy 7183563.8 --Oz 593.5 --hemispherical --binsize 0.005 \
-i 58_forest_registered.las -o 58_forest_registered_ hemispherical.spd
```

3.1.4 Help File

```
spdtranslate SPDLib 3.0.0, Copyright (C) 2012 Sorted Pulse Library (SPD)
This program comes with ABSOLUTELY NO WARRANTY. This is free software,
and you are welcome to redistribute it under certain conditions; See
website (http://www.spdlib.org). Bugs are to be reported on the trac
or directly to spdlib-develop@lists.sourceforge.net

USAGE:

   spdtranslate  -o <String> -i <String> [--pulseversion <unsigned int>]
                 [--pointversion <unsigned int>] [--wavenoise <float>]
                 [--Oz <float>] [--Oy <double>] [--Ox <double>]
                 [--defineOrigin] [--tly <double>] [--tlx <double>]
                 [--defineTL] [--keeptemp] [--convert_proj] [--output_proj
                 <string>] [--input_proj <string>] [-b <float>] [-c
                 <unsigned int>] [-r <unsigned int>] [-s <string>] [-t
                 <string>] [--scan] [--polar] [--spherical] [--wavebitres
                 <8BIT|16BIT|32BIT>] [-x <FIRST_RETURN|LAST_RETURN
                 |START_WAVEFORM|END_WAVEFORM|ORIGIN|MAX_INTENSITY
                 |UNCHANGED>] --of <SPD|UPD|ASCII|LAS|LAZ> --if <SPD
                 |ASCIIPULSEROW|ASCII|FWF_DAT|DECOMPOSED_DAT|LAS|LASNP
                 |LASSTRICT|DECOMPOSED_COO|ASCIIMULTILINE> [--] [--version]
                 [-h]


Where:

   -o <String>,  --output <String>
     (required)  The output file.

   -i <String>,  --input <String>
     (required)  The input file.

   --pulseversion <unsigned int>
     Specify the pulse version to be used within the SPD file (Default: 2)
```

```
--pointversion <unsigned int>
  Specify the point version to be used within the SPD file (Default: 2)

--wavenoise <float>
  Waveform noise threshold (Default 0)

--Oz <float>
  Origin Z coordinate

--Oy <double>
  Origin Y coordinate

--Ox <double>
  Origin X coordinate.

--defineOrigin
  Define the origin coordinate for the SPD.

--tly <double>
  Top left Y coordinate for defining the SPD file index.

--tlx <double>
  Top left X coordinate for defining the SPD file index.

--defineTL
  Define the top left (TL) coordinate for the SPD file index

--keeptemp
  Keep the tempory files generated during the conversion.

--convert_proj
  Convert file buffering to disk

--output_proj <string>
  WKT string representing the projection of the output file

--input_proj <string>
  WKT string representing the projection of the input file

-b <float>,  --binsize <float>
```

```
  Bin size for SPD file index (Default 1)

-c <unsigned int>,  --numofcols <unsigned int>
  Number of columns within a tile (Default 0), using this option
  generats a non-sequencial SPD file.

-r <unsigned int>,  --numofrows <unsigned int>
  Number of rows within a tile (Default 25)

-s <string>,  --schema <string>
  A schema for the format of the file being imported (Note, most
  importers do not require a schema)

-t <string>,  --temppath <string>
  A path were temporary files can be written too

--scan
  Index the pulses using a scan coordinate system

--polar
  Index the pulses using a polar coordinate system

--spherical
  Index the pulses using a spherical coordinate system

--wavebitres <8BIT|16BIT|32BIT>
  The bit resolution used for storing the waveform data (Default: 32BIT)

-x <FIRST_RETURN|LAST_RETURN|START_WAVEFORM|END_WAVEFORM|ORIGIN
   |MAX_INTENSITY|UNCHANGED>,  --indexfield <FIRST_RETURN|LAST_RETURN
   |START_WAVEFORM|END_WAVEFORM|ORIGIN|MAX_INTENSITY|UNCHANGED>
  The location used to index the pulses (Default: UNCHANGED)

--of <SPD|UPD|ASCII|LAS|LAZ>
  (required)  Format of the output file (Default SPD)

--if <SPD|ASCIIPULSEROW|ASCII|FWF_DAT|DECOMPOSED_DAT|LAS|LASNP|LASSTRICT
   |DECOMPOSED_COO|ASCIIMULTILINE>
  (required)  Format of the input file (Default SPD)

--,  --ignore_rest
```

```
    Ignores the rest of the labeled arguments following this flag.

  --version
    Displays version information and exits.

  -h,  --help
    Displays usage information and exits.


  Convert between file formats: spdtranslate
```

3.2 Subset Data – spdsubset

The spdsubset command supports the sub-setting of any of the supported input formats. The command has two modes either a polygon (as a shapefile) can be provided and the input file will be subset to the polygon (using the –shpfile parameter) or the individual dimensions of the dataset can be specified and the file cut to the parameters specified. Note the parameters not specified will be replaced by the largest or smallest (depending on parameter) value in the file.

3.2.1 Subset to shapefile

```
spdsubset --shpfile polygonshp.shp -i inputLiDAR.spd -o outputLiDAR.spd
```

3.2.2 Subset to bounding box

```
spdsubset --xmin 500000 --xmax 501000 --ymin 7200000 --ymax 721000 \
-i inputLiDAR.spd -o outputLiDAR.spd
```

3.2.3 Subset to bounding volume

```
spdsubset --xmin 500000 --xmax 501000 --ymin 7200000 --ymax 721000 \
--zmin 250 --zmax 350 -i inputLiDAR.spd -o outputLiDAR.spd
```

3.2.4 Subset to bounding volume - using height above ground

Note, that to use this command the height field in the SPD/UPD file needs to have previously been defined (see spddefheight).

```
spdsubset --xmin 500000 --xmax 501000 --ymin 7200000 --ymax 721000 \
--hmin 0 --hmax 50 --height -i inputLiDAR.spd -o outputLiDAR.spd
```

3.2.5 Subset to minimum Y

```
spdsubset --inputformat SPD --outputformat SPD --ymin 7200000 \
-i inputLiDAR.spd -o outputLiDAR.spd
```

3.2.6 Subset to maximum Z

```
spdsubset --zmax 500 -i inputLiDAR.spd -o outputLiDAR.spd
```

3.2.7 Help File

```
spdsubset SPDLib 3.0.0, Copyright (C) 2012 Sorted Pulse Library (SPD)
This program comes with ABSOLUTELY NO WARRANTY. This is free software,
and you are welcome to redistribute it under certain conditions; See
website (http://www.spdlib.org). Bugs are to be reported on the trac
or directly to spdlib-develop@lists.sourceforge.net

USAGE:

   spdsubset  [--num <uint_fast32_t>] [--start <uint_fast32_t>] [--shpfile
              <string>] [--ignorez] [--ignorerange] [--txtfile <string>]
              [--spherical] [--height] [--ranmax <double>] [--ranmin
              <double>] [--zenmax <double>] [--zenmin <double>] [--azmax
              <double>] [--azmin <double>] [--hmax <double>] [--hmin
              <double>] [--zmax <double>] [--zmin <double>] [--ymax
              <double>] [--ymin <double>] [--xmax <double>] [--xmin
              <double>] [--] [--version] [-h] <string> ...
```

```
Where:

   --num <uint_fast32_t>
     Number of pulses to be exported

   --start <uint_fast32_t>
     First pulse in the block

   --shpfile <string>
     A shapefile to which the dataset should be subsetted to

   --ignorez
     Defining that Z should be ignored when subsetting using a text file.

   --ignorerange
     Defining that range should be ignored when subsetting using a text
     file.

   --txtfile <string>
     A text containing the extent to which the file should be cut to.

   --spherical
     Subset a spherically indexed SPD file.

   --height
     Threshold the height of each pulse (currently only valid with SPD to
     SPD subsetting)

   --ranmax <double>
     Maximum range threshold

   --ranmin <double>
     Minimum range threshold

   --zenmax <double>
     Maximum zenith threshold

   --zenmin <double>
     Minimum zenith threshold

   --azmax <double>
```

```
  Maximum azmuth threshold

--azmin <double>
  Minimum azimuth threshold

--hmax <double>
  Maximum Height threshold

--hmin <double>
  Minimum Height threshold

--zmax <double>
  Maximum Z threshold

--zmin <double>
  Minimum Z threshold

--ymax <double>
  Maximum Y threshold

--ymin <double>
  Minimum Y threshold

--xmax <double>
  Maximum X threshold

--xmin <double>
  Minimum X threshold

--,  --ignore_rest
  Ignores the rest of the labeled arguments following this flag.

--version
  Displays version information and exits.

-h,  --help
  Displays usage information and exits.

<string>  (accepted multiple times)
  (required)  File names for the input and output files
```

```
Subset point cloud data: spdsubset
```

3.3 Merging Data Files – spdmerge

3.3.1 Help File

```
spdmerge SPDLib 3.0.0, Copyright (C) 2012 Sorted Pulse Library (SPD)
This program comes with ABSOLUTELY NO WARRANTY. This is free software,
and you are welcome to redistribute it under certain conditions; See
website (http://www.spdlib.org). Bugs are to be reported on the trac
or directly to spdlib-develop@lists.sourceforge.net

USAGE:

   spdmerge  -o <String> [-s <std::string>] [--classes <uint_fast16_t>] ...
             [--returnIDs <uint_fast16_t>] ...  [--source] [--ignorechecks]
             [-c] [-r <std::string>] [-p <std::string>] [--wavebitres <8BIT
             |16BIT|32BIT>] [-x <FIRST_RETURN|LAST_RETURN|START_WAVEFORM
             |END_WAVEFORM|ORIGIN>] -f <SPD|ASCIIPULSEROW|ASCII|FWF_DAT
             |DECOMPOSED_DAT|LAS|ASCIIMULTILINE> [--] [--version] [-h]
             <std::string> ...


Where:

   -o <String>,  --output <String>
     (required)  The output SPD file.

   -s <std::string>,  --schema <std::string>
     A schema for the format of the file being imported (Note, most
     importers do not require a schema)

   --classes <uint_fast16_t>  (accepted multiple times)
     Lists the classes for the files listed.

   --returnIDs <uint_fast16_t>  (accepted multiple times)
     Lists the return IDs for the files listed.
```

```
--source
  Set source ID for each input file

--ignorechecks
  Ignore checks between input files to ensure compatibility

-c,  --convert_proj
  Convert file buffering to disk

-r <std::string>,  --output_proj <std::string>
  WKT std::string representing the projection of the output file

-p <std::string>,  --input_proj <std::string>
  WKT std::string representing the projection of the input file

--wavebitres <8BIT|16BIT|32BIT>
  The bit resolution used for storing the waveform data (Default: 32BIT)

-x <FIRST_RETURN|LAST_RETURN|START_WAVEFORM|END_WAVEFORM|ORIGIN>,
   --indexfield <FIRST_RETURN|LAST_RETURN|START_WAVEFORM|END_WAVEFORM
   |ORIGIN>
  The location used to index the pulses

-f <SPD|ASCIIPULSEROW|ASCII|FWF_DAT|DECOMPOSED_DAT|LAS|ASCIIMULTILINE>,
   --inputformat <SPD|ASCIIPULSEROW|ASCII|FWF_DAT|DECOMPOSED_DAT|LAS
   |ASCIIMULTILINE>
  (required)  Format of the input file

--,  --ignore_rest
  Ignores the rest of the labeled arguments following this flag.

--version
  Displays version information and exits.

-h,  --help
  Displays usage information and exits.

<std::string>  (accepted multiple times)
  (required)  The list of input files
```

```
Merge compatable files into a single non-indexed SPD file: spdmerge
```

3.4 Extract Data – spdextract

3.4.1 Help File

```
spdextract SPDLib 3.0.0, Copyright (C) 2012 Sorted Pulse Library (SPD)
This program comes with ABSOLUTELY NO WARRANTY. This is free software,
and you are welcome to redistribute it under certain conditions; See
website (http://www.spdlib.org). Bugs are to be reported on the trac
or directly to spdlib-develop@lists.sourceforge.net

USAGE:

   spdextract  -o <String> -i <String> [--return <ALL|FIRST|LAST|NOTFIRST
               |FIRSTLAST>] [--class <unsigned int>] [-c <unsigned int>]
               [-r <unsigned int>] [--] [--version] [-h]


Where:

   -o <String>,  --output <String>
     (required)  The output SPD file.

   -i <String>,  --input <String>
     (required)  The input SPD file.

   --return <ALL|FIRST|LAST|NOTFIRST|FIRSTLAST>
     The return(s) of interest

   --class <unsigned int>
     Class of interest (Ground == 3; Not Ground == 104)

   -c <unsigned int>,  --blockcols <unsigned int>
     Number of columns within a block (Default 0) - Note values greater
     than 1 result in a non-sequencial SPD file.
```

```
   -r <unsigned int>,  --blockrows <unsigned int>
     Number of rows within a block (Default 100)

   --,  --ignore_rest
     Ignores the rest of the labeled arguments following this flag.

   --version
     Displays version information and exits.

   -h,  --help
     Displays usage information and exits.


   Extract returns and pulses which meet a set of criteria: spdextract
```

3.5 Copying Data – spdcopy

3.5.1 Help File

```
spdcopy SPDLib 3.0.0, Copyright (C) 2012 Sorted Pulse Library (SPD)
This program comes with ABSOLUTELY NO WARRANTY. This is free software,
and you are welcome to redistribute it under certain conditions; See
website (http://www.spdlib.org). Bugs are to be reported on the trac
or directly to spdlib-develop@lists.sourceforge.net

USAGE:

   spdcopy  -o <String> -i <String> [-c <unsigned int>] [-r <unsigned int>]
            [--] [--version] [-h]


Where:

   -o <String>,  --output <String>
     (required)  The output file.

   -i <String>,  --input <String>
     (required)  The input file.
```

```
   -c <unsigned int>,  --blockcols <unsigned int>
     Number of columns within a block (Default 0) - Note values greater
     than 1 result in a non-sequencial SPD file.

   -r <unsigned int>,  --blockrows <unsigned int>
     Number of rows within a block (Default 100)

   --,  --ignore_rest
     Ignores the rest of the labeled arguments following this flag.

   --version
     Displays version information and exits.

   -h,  --help
     Displays usage information and exits.


   Makes a copy of an indexed SPD file: spdcopy
```

3.6 Overlap – spdoverlap

3.6.1 Help File

```
spdoverlap SPDLib 3.0.0, Copyright (C) 2012 Sorted Pulse Library (SPD)
This program comes with ABSOLUTELY NO WARRANTY. This is free software,
and you are welcome to redistribute it under certain conditions; See
website (http://www.spdlib.org). Bugs are to be reported on the trac
or directly to spdlib-develop@lists.sourceforge.net

USAGE:

   spdoverlap  {-c|-s} [-o <String>] [--] [--version] [-h] <string> ...


Where:

   -c,  --cartesian
```

```
    (OR required)  Find cartesian overlap.
        -- OR --
  -s,  --spherical
    (OR required)  Find spherical overlap.


  -o <String>,  --output <String>
    The output file.

  --,  --ignore_rest
    Ignores the rest of the labeled arguments following this flag.

  --version
    Displays version information and exits.

  -h,  --help
    Displays usage information and exits.

  <string>  (accepted multiple times)
    (required)  File names for the output (if required) and input files


  Calculate the overlap between UPD and SPD files: spdoverlap
```

3.7 Registration – spdwarp

3.7.1 Help File

```
spdwarp SPDLib 3.0.0, Copyright (C) 2012 Sorted Pulse Library (SPD)
This program comes with ABSOLUTELY NO WARRANTY. This is free software,
and you are welcome to redistribute it under certain conditions; See
website (http://www.spdlib.org). Bugs are to be reported on the trac
or directly to spdlib-develop@lists.sourceforge.net

USAGE:

  spdwarp  {--shift|--warp} -o <String> -i <String> [-c <unsigned int>]
           [-r <unsigned int>] [-y <float>] [-x <float>] [-g
```

```
            <std::string>] [--order <unsigned int>] [-p <>] [-t <POLYNOMIAL
            |NEAREST_NEIGHBOR|TRIANGULATION>] [--] [--version] [-h]


Where:

   --shift
     (OR required)  Apply a linear shift to the SPD file.
         -- OR --
   --warp
     (OR required)  Apply a nonlinear warp to the SPD file defined by a set
     of GCPs.


   -o <String>,  --output <String>
     (required)  The output SPD file.

   -i <String>,  --input <String>
     (required)  The input SPD file.

   -c <unsigned int>,  --blockcols <unsigned int>
     Number of columns within a block (Default 0) - Note values greater
     than 1 result in a non-sequencial SPD file.

   -r <unsigned int>,  --blockrows <unsigned int>
     Number of rows within a block (Default 100)

   -y <float>,  --yshift <float>
     SHIFT: The y shift in the units of the dataset (probably metres).

   -x <float>,  --xshift <float>
     SHIFT: The x shift in the units of the dataset (probably metres).

   -g <std::string>,  --gcps <std::string>
     WARP: The path and file name of the gcps file.

   --order <unsigned int>
     POLY TRANSFORM (Default=3): The order of the polynomial fitted.

   -p <>,  --pulsewarp <>
     WARP (Default=PULSE_IDX): The eastings and northings used to calculate
```

```
     the warp. ALL_RETURNS recalculates the offsets for each X,Y while
     PULSE_IDX and PULSE_ORIGIN use a single offset for the whole pulse.

   -t <POLYNOMIAL|NEAREST_NEIGHBOR|TRIANGULATION>,  --transform <POLYNOMIAL
      |NEAREST_NEIGHBOR|TRIANGULATION>
     WARP (Default=POLYNOMIAL): The transformation model to be fitted to
     the GPCs and used to warp the data.

   --,  --ignore_rest
     Ignores the rest of the labeled arguments following this flag.

   --version
     Displays version information and exits.

   -h,  --help
     Displays usage information and exits.


   Interpolate a raster elevation surface: spdwarp
```

3.8 Tiling Data

3.8.1 Defining Tiles – spddeftiles

Help File

```
spddeftiles SPDLib 3.0.0, Copyright (C) 2012 Sorted Pulse Library (SPD)
This program comes with ABSOLUTELY NO WARRANTY. This is free software,
and you are welcome to redistribute it under certain conditions; See
website (http://www.spdlib.org). Bugs are to be reported on the trac
or directly to spdlib-develop@lists.sourceforge.net

USAGE:

   spddeftiles  {-t|-e} [-i <String>] [-o <String>] [--ymax <double>]
                [--xmax <double>] [--ymin <double>] [--xmin <double>]
                [--overlap <double>] [--ysize <double>] [--xsize <double>]
                [--] [--version] [-h]
```

```
Where:

   -t,  --tiles
     (OR required)  Define a set of tiles for a region.
         -- OR --
   -e,  --extent
     (OR required)  Calculate the extent of a set of files.


   -i <String>,  --input <String>
     Input file listing the set of input files (--extent).

   -o <String>,  --output <String>
     Output XML file defining the tiles (--tiles).

   --ymax <double>
     Y max (in units of coordinate systems) of the region to be tiled
     (--tiles).

   --xmax <double>
     X max (in units of coordinate systems) of the region to be tiled
     (--tiles).

   --ymin <double>
     Y min (in units of coordinate systems) of the region to be tiled
     (--tiles).

   --xmin <double>
     X min (in units of coordinate systems) of the region to be tiled
     (--tiles).

   --overlap <double>
     Size (in units of coordinate systems) of the overlap for tiles
     (Default 100) (--tiles).

   --ysize <double>
     Y size (in units of coordinate systems) of the tiles (Default 1000)
     (--tiles).
```

```
   --xsize <double>
     X size (in units of coordinate systems) of the tiles (Default 1000)
     (--tiles).

   --,  --ignore_rest
     Ignores the rest of the labeled arguments following this flag.

   --version
     Displays version information and exits.

   -h,  --help
     Displays usage information and exits.


   Tools for defining a set of tiles: spddeftiles
```

3.8.2 Performing Tiling – spdtiling

Help File

```
spdtiling SPDLib 3.0.0, Copyright (C) 2012 Sorted Pulse Library (SPD)
This program comes with ABSOLUTELY NO WARRANTY. This is free software,
and you are welcome to redistribute it under certain conditions; See
website (http://www.spdlib.org). Bugs are to be reported on the trac
or directly to spdlib-develop@lists.sourceforge.net

USAGE:

   spdtiling  {--all|--extract|--extractcore} [-u] [-d] [-c <Unsigned int>]
              [-r <Unsigned int>] -i <String> -o <String> -t <String> [--]
              [--version] [-h]


Where:

   --all
     (OR required)  Create all tiles.
         -- OR --
   --extract
```

```
  (OR required)  Extract an individual tile as specified in the XML
  file.
      -- OR --
--extractcore
  (OR required)  Extract the core of a tile as specified in the XML
  file.


-u,  --updatexml
  Update the tiles XML file.

-d,  --deltiles
  Remove tiles which have no data.

-c <Unsigned int>,  --col <Unsigned int>
  The column of the tile to be extracted (--extract).

-r <Unsigned int>,  --row <Unsigned int>
  The row of the tile to be extracted (--extract).

-i <String>,  --input <String>
  (required)  A text file with a list of input files, one per line.
  (--extractcore expects a single input SPD file)

-o <String>,  --output <String>
  (required)  The base path for the tiles. (--extractcore expects a
  single output SPD file)

-t <String>,  --tiles <String>
  (required)  XML file defining the tile regions

--,  --ignore_rest
  Ignores the rest of the labeled arguments following this flag.

--version
  Displays version information and exits.

-h,  --help
  Displays usage information and exits.
```

```
   Tools for tiling a set of SPD files using predefined tile areas:
   spdtiling
```

3.8.3 Mosaicing Results – spdtileimg

Help File

```
spdtileimg SPDLib 3.0.0, Copyright (C) 2012 Sorted Pulse Library (SPD)
This program comes with ABSOLUTELY NO WARRANTY. This is free software,
and you are welcome to redistribute it under certain conditions; See
website (http://www.spdlib.org). Bugs are to be reported on the trac
or directly to spdlib-develop@lists.sourceforge.net

USAGE:

   spdtileimg  {-m|-c} [-i <String>] [-t <String>] [-o <String>] [-w
               <String>] [-r <double>] [-f <String>] [--] [--version] [-h]


Where:

   -m,  --mosaic
     (OR required)  Mosaic the images (within the input list) together.
         -- OR --
   -c,  --clump
     (OR required)  Create a clumps image specifying the location of the
     tiles.


   -i <String>,  --input <String>
     The text file with a list of input files.

   -t <String>,  --tiles <String>
     The input XML file defining the tiles.

   -o <String>,  --output <String>
     The output image.

   -w <String>,  --wkt <String>
```

```
    A file containing the WKT string representing the projection (--clump
    only).

-r <double>,  --resolution <double>
    The output image pixel size (--clump only).

-f <String>,  --format <String>
    The output image format.

--,  --ignore_rest
    Ignores the rest of the labeled arguments following this flag.

--version
    Displays version information and exits.

-h,  --help
    Displays usage information and exits.


Tools for mosaicing raster results following tiling: spdtileimg
```

Chapter 4

Utility Commands

4.1 SPDLib Version – spdversion

4.1.1 Help File

```
spdversion SPDLib 3.0.0, Copyright (C) 2012 Sorted Pulse Library (SPD)
This program comes with ABSOLUTELY NO WARRANTY. This is free software,
and you are welcome to redistribute it under certain conditions; See
website (http://www.spdlib.org). Bugs are to be reported on the trac
or directly to spdlib-develop@lists.sourceforge.net

USAGE:

   spdversion  [--] [--version] [-h] <string> ...


Where:

   --,  --ignore_rest
     Ignores the rest of the labeled arguments following this flag.

   --version
     Displays version information and exits.

   -h,  --help
```

```
     Displays usage information and exits.

   <string>  (accepted multiple times)
     File names for the input files


   Prints version information: spdversion
```

4.2 File Summary – spdinfo

4.2.1 Help File

```
spdinfo SPDLib 3.0.0, Copyright (C) 2012 Sorted Pulse Library (SPD)
This program comes with ABSOLUTELY NO WARRANTY. This is free software,
and you are welcome to redistribute it under certain conditions; See
website (http://www.spdlib.org). Bugs are to be reported on the trac
or directly to spdlib-develop@lists.sourceforge.net

USAGE:

   spdinfo  [--] [--version] [-h] <string> ...


Where:

   --,  --ignore_rest
     Ignores the rest of the labeled arguments following this flag.

   --version
     Displays version information and exits.

   -h,  --help
     Displays usage information and exits.

   <string>  (accepted multiple times)
     Input file
```

```
Print header info for an SPD File: spdinfo
```

4.3 Find WKT Projections – spdproj

4.3.1 Help File

```
USAGE:

   spdproj  {--proj4 <string>|--proj4pretty <string>|--image <string>
            |--imagepretty <string>|--spd <string>|--spdpretty <string>
            |--epsg <int>|--epsgpretty <int>|--shp <string>|--shppretty
            <string>} [--] [--version] [-h]


Where:

   --proj4 <string>
     (OR required)  Enter a proj4 string (to print WKT)
         -- OR --
   --proj4pretty <string>
     (OR required)  Enter a proj4 string (to print Pretty WKT)
         -- OR --
   --image <string>
     (OR required)  Print the WKT string associated with the input image.
         -- OR --
   --imagepretty <string>
     (OR required)  Print the WKT (to print Pretty WKT) string associated
     with the input image.
         -- OR --
   --spd <string>
     (OR required)  Print the WKT string associated with the input spd
     file.
         -- OR --
   --spdpretty <string>
     (OR required)  Print the WKT (to print Pretty WKT) string associated
     with the input spd file.
         -- OR --
   --epsg <int>
```

```
    (OR required)  Print the WKT string associated with the EPSG code
    provided.
        -- OR --
  --epsgpretty <int>
    (OR required)  Print the WKT (to print Pretty WKT) string associated
    with the EPSG code provided.
        -- OR --
  --shp <string>
    (OR required)  Print the WKT string associated with the input ESRI
    shapefile.
        -- OR --
  --shppretty <string>
    (OR required)  Print the WKT (to print Pretty WKT) string associated
    with the input ESRI shapefile.


  --,  --ignore_rest
    Ignores the rest of the labeled arguments following this flag.

  --version
    Displays version information and exits.

  -h,  --help
    Displays usage information and exits.


  Print and convert projection strings: spdproj
```

4.4 Errors with LAS files – spdlastest

4.4.1 Help File

```
spdlastest SPDLib 3.0.0, Copyright (C) 2012 Sorted Pulse Library (SPD)
This program comes with ABSOLUTELY NO WARRANTY. This is free software,
and you are welcome to redistribute it under certain conditions; See
website (http://www.spdlib.org). Bugs are to be reported on the trac
or directly to spdlib-develop@lists.sourceforge.net
```

```
USAGE:

   spdlastest  {-p|-c|-f} -i <String> [-n <unsigned int>] [-s <unsigned
               int>] [--] [--version] [-h]


Where:

   -p,  --print
     (OR required)  Print a selction of pulses from LAS file.
         -- OR --
   -c,  --count
     (OR required)  Count the number of pulses in LAS file.
         -- OR --
   -f,  --notfirst
     (OR required)  Print the returns which start a pulse with point IDs
     greater than 1


   -i <String>,  --input <String>
     (required)  The input SPD file.

   -n <unsigned int>,  --number <unsigned int>
     Number of pulses to be printed out (Default 10)

   -s <unsigned int>,  --start <unsigned int>
     Starting pulse index (Default 0)

   --,  --ignore_rest
     Ignores the rest of the labeled arguments following this flag.

   --version
     Displays version information and exits.

   -h,  --help
     Displays usage information and exits.


   Print data pulses from a LAS file - for debugging: spdlastest
```

4.5 Clear all classifications – spdclearclass

4.5.1 Help File

```
spdclearclass SPDLib 3.0.0, Copyright (C) 2012 Sorted Pulse Library (SPD)
This program comes with ABSOLUTELY NO WARRANTY. This is free software,
and you are welcome to redistribute it under certain conditions; See
website (http://www.spdlib.org). Bugs are to be reported on the trac
or directly to spdlib-develop@lists.sourceforge.net

USAGE:

   spdclearclass  -o <String> -i <String> [-c <unsigned int>] [-r <unsigned
                  int>] [--] [--version] [-h]


Where:

   -o <String>,  --output <String>
     (required)  The output SPD file.

   -i <String>,  --input <String>
     (required)  The input SPD file.

   -c <unsigned int>,  --blockcols <unsigned int>
     Number of columns within a block (Default 0) - Note values greater
     than 1 result in a non-sequencial SPD file.

   -r <unsigned int>,  --blockrows <unsigned int>
     Number of rows within a block (Default 100)

   --,  --ignore_rest
     Ignores the rest of the labeled arguments following this flag.

   --version
     Displays version information and exits.

   -h,  --help
     Displays usage information and exits.
```

```
Clear the classification of an SPD file: spdclearclass
```

4.6 Define RGB Values – spddefrgb

4.6.1 Help File

```
spddefrgb SPDLib 3.0.0, Copyright (C) 2012 Sorted Pulse Library (SPD)
This program comes with ABSOLUTELY NO WARRANTY. This is free software,
and you are welcome to redistribute it under certain conditions; See
website (http://www.spdlib.org). Bugs are to be reported on the trac
or directly to spdlib-develop@lists.sourceforge.net

USAGE:

   spddefrgb  {--define|--stretch} -o <String> --image <String> -i <String>
              [--blue <uint_fast16_t>] [--green <uint_fast16_t>] [--red
              <uint_fast16_t>] [-c <unsigned int>] [-r <unsigned int>]
              [--stddev] [--linear] [--] [--version] [-h]


Where:

   --define
     (OR required)  Define the RGB values on an SPD file from an input
     image.
         -- OR --
   --stretch
     (OR required)  Stretch existing RGB values to a range of 0 to 255.


   -o <String>,  --output <String>
     (required)  The output SPD file.

   --image <String>
     (required)  The input image file.

   -i <String>,  --input <String>
```

```
  (required)  The input SPD file.

--blue <uint_fast16_t>
  Image band for blue channel

--green <uint_fast16_t>
  Image band for green channel

--red <uint_fast16_t>
  Image band for red channel

-c <unsigned int>,  --blockcols <unsigned int>
  Number of columns within a block (Default 0) - Note values greater
  than 1 result in a non-sequencial SPD file.

-r <unsigned int>,  --blockrows <unsigned int>
  Number of rows within a block (Default 100)

--stddev
  Use a linear 2 standard deviation stretch.

--linear
  Use a linear stretch between the min and max values.

--,  --ignore_rest
  Ignores the rest of the labeled arguments following this flag.

--version
  Displays version information and exits.

-h,  --help
  Displays usage information and exits.


Define the RGB values on the SPDFile: spddefrgb
```

4.7 Generate an image mask – spdmaskgen

4.7.1 Help File

```
spdstats SPDLib 3.0.0, Copyright (C) 2012 Sorted Pulse Library (SPD)
This program comes with ABSOLUTELY NO WARRANTY. This is free software,
and you are welcome to redistribute it under certain conditions; See
website (http://www.spdlib.org). Bugs are to be reported on the trac
or directly to spdlib-develop@lists.sourceforge.net

USAGE:

   spdmaskgen  -o <String> -i <String> [-f <string>] [-b <float>] [-c
               <unsigned int>] [-r <unsigned int>] [-p <unsigned int>] [--]
               [--version] [-h]


Where:

   -o <String>,  --output <String>
     (required)  The output SPD file.

   -i <String>,  --input <String>
     (required)  The input SPD file.

   -f <string>,  --format <string>
     Image format (GDAL driver string), Default is ENVI.

   -b <float>,  --binsize <float>
     Bin size for processing and output image (Default 0) - Note 0 will use
     the native SPD file bin size.

   -c <unsigned int>,  --blockcols <unsigned int>
     Number of columns within a block (Default 0) - Note values greater
     than 1 result in a non-sequencial SPD file.

   -r <unsigned int>,  --blockrows <unsigned int>
     Number of rows within a block (Default 100)

   -p <unsigned int>,  --numpulses <unsigned int>
```

```
    Number of pulses for a bin to be included in mask (Default 1)

  --,  --ignore_rest
    Ignores the rest of the labeled arguments following this flag.

  --version
    Displays version information and exits.

  -h,  --help
    Displays usage information and exits.


  Generate a binary mask for the an input SPD File: spdmaskgen
```

4.8 Generate Statistics – spdstats

4.8.1 Help File

```
spdstats SPDLib 3.0.0, Copyright (C) 2012 Sorted Pulse Library (SPD)
This program comes with ABSOLUTELY NO WARRANTY. This is free software,
and you are welcome to redistribute it under certain conditions; See
website (http://www.spdlib.org). Bugs are to be reported on the trac
or directly to spdlib-develop@lists.sourceforge.net

USAGE:

  spdstats  {--image|--overall} -o <String> -i <String> [-f <string>] [-b
            <float>] [-c <unsigned int>] [-r <unsigned int>] [--]
            [--version] [-h]


Where:

  --image
    (OR required)  Create a point / pulses density statistics image
        -- OR --
  --overall
    (OR required)  Create overall statistics for point / pulse density
```

```
-o <String>,  --output <String>
  (required)  The output SPD file.

-i <String>,  --input <String>
  (required)  The input SPD file.

-f <string>,  --format <string>
  Image format (GDAL driver string), Default is ENVI.

-b <float>,  --binsize <float>
  Bin size for processing and output image (Default 0) - Note 0 will use
  the native SPD file bin size.

-c <unsigned int>,  --blockcols <unsigned int>
  Number of columns within a block (Default 0) - Note values greater
  than 1 result in a non-sequencial SPD file.

-r <unsigned int>,  --blockrows <unsigned int>
  Number of rows within a block (Default 100)

--,  --ignore_rest
  Ignores the rest of the labeled arguments following this flag.

--version
  Displays version information and exits.

-h,  --help
  Displays usage information and exits.


Provides statistics the point and pulse density of an SPD file: spdstats
```

4.9 Thin the point cloud – spdthin

4.9.1 Help File

```
spdthin SPDLib 3.0.0, Copyright (C) 2012 Sorted Pulse Library (SPD)
This program comes with ABSOLUTELY NO WARRANTY. This is free software,
and you are welcome to redistribute it under certain conditions; See
website (http://www.spdlib.org). Bugs are to be reported on the trac
or directly to spdlib-develop@lists.sourceforge.net

USAGE:

   spdthin  -o <String> -i <String> [-n <unsigned int>] [-c <unsigned int>]
            [-r <unsigned int>] [--] [--version] [-h]


Where:

   -o <String>,  --output <String>
     (required)  The output SPD file.

   -i <String>,  --input <String>
     (required)  The input SPD file.

   -n <unsigned int>,  --numpulses <unsigned int>
     Number of pulses within the bin (Default 1).

   -c <unsigned int>,  --blockcols <unsigned int>
     Number of columns within a block (Default 0) - Note values greater
     than 1 result in a non-sequencial SPD file.

   -r <unsigned int>,  --blockrows <unsigned int>
     Number of rows within a block (Default 100)

   --,  --ignore_rest
     Ignores the rest of the labeled arguments following this flag.

   --version
     Displays version information and exits.
```

```
-h,  --help
  Displays usage information and exits.


Thin a point cloud to a defined bin spacing: spdthin
```

Chapter 5

Data Processing Tools

5.1 Decompose Waveforms – spddecomp

5.1.1 Help File

```
spddecomp SPDLib 3.0.0, Copyright (C) 2012 Sorted Pulse Library (SPD)
This program comes with ABSOLUTELY NO WARRANTY. This is free software,
and you are welcome to redistribute it under certain conditions; See
website (http://www.spdlib.org). Bugs are to be reported on the trac
or directly to spdlib-develop@lists.sourceforge.net

USAGE:

   spddecomp  -o <String> -i <String> [-a] [-d <float>] [-w
              <uint_fast32_t>] [-e <uint_fast32_t>] [-n] [-t
              <uint_fast32_t>] [-c <unsigned int>] [-r <unsigned int>] [--]
              [--version] [-h]


Where:

   -o <String>,  --output <String>
     (required)  The output file.

   -i <String>,  --input <String>
```

```
  (required)  The input file.

-a,  --all
  Fit all Gaussian at once

-d <float>,  --decay <float>
  Decay value for ignoring ringing artifacts (Default 5)

-w <uint_fast32_t>,  --window <uint_fast32_t>
  Window for the values taken either side of the peak for fitting
  (Default 5)

-e <uint_fast32_t>,  --decaythres <uint_fast32_t>
  Intensity threshold above which a decay function is used (Default 100)

-n,  --noise
  Estimate noise. Only applicable when --all is set. Note an initial
  estimate is required for peak detection (see -t)

-t <uint_fast32_t>,  --threshold <uint_fast32_t>
  Noise threshold below which peaks are ignored (Default: Value in
  pulse->waveNoiseThreshold)

-c <unsigned int>,  --blockcols <unsigned int>
  Number of columns within a block (Default 0) - Note values greater
  than 1 result in a non-sequencial SPD file.

-r <unsigned int>,  --blockrows <unsigned int>
  Number of rows within a block (Default 100)

--,  --ignore_rest
  Ignores the rest of the labeled arguments following this flag.

--version
  Displays version information and exits.

-h,  --help
  Displays usage information and exits.


Decompose full waveform data to create discrete points: spddecomp
```

5.2 Define Height Fields – spddefheight

The spddefheight command is used to define the height field within both the pulse and point fields of the SPD data file. This can be done in two ways, the simplest is the use of a DTM of the same resolution as the SPD file bin size. The disadvantage of using a DTM is that it is in effect using a series of spot heights and this can introduce artefacts. Therefore, interpolating a value for each point/pulse generates a continuous surface reducing any artefacts. The recommended approach is to use Natural Neighbour interpolation, as demonstrated in the paper of (Bater and Coops, 2009).

5.2.1 Without Interpolation, using DTM

Using the DTM method the only parameters are the input file and output files. The raster DTM needs to the same resolution as the SPD grid and it can be any raster format supported by the GDAL library.

```
spddefheight --dtm -i p144_LiDAR_1m_grd.spd -e dtm_1m.kea -o p144_LiDAR_1m_grd_ht.spd
```

5.2.2 Interpolation Mode

The interpolators are the same as those defined within the spdinterp command with the same parameters so look to this command for details on their use and information on the thinning process available using the –thin parameters.

The recommend command using the natural neighbour interpolation algorithm, along with the default parameters, is shown below:

```
spddefheight --interp --in NATURAL_NEIGHBOR -i liukdr_40205250_20090601_1m_grd.spd \
-o liukdr_40205250_20090601_1m_grd_nnht.spd
```

While with the wider range of options to control the block size used for processing (and therefore memory usage) and the thinning process is given below.

```
spddefheight --interp --overlap 20 --block 200 --thin --thinres 0.5 --ptsperbin 1 \
--in NATURAL_NEIGHBOR -i liukdr_40205250_20090601_1m_grd.spd \
-o liukdr_40205250_20090601_1m_grd_nnht.spd
```

5.2.3 Help File

```
spddefheight SPDLib 3.0.0, Copyright (C) 2012 Sorted Pulse Library (SPD)
This program comes with ABSOLUTELY NO WARRANTY. This is free software,
and you are welcome to redistribute it under certain conditions; See
website (http://www.spdlib.org). Bugs are to be reported on the trac
or directly to spdlib-develop@lists.sourceforge.net

USAGE:

   spddefheight  {--interp|--image} -o <String> [-e <String>] -i <String>
                 [--idxres <float>] [--thinres <float>] [--ptsperbin
                 <uint_fast16_t>] [--thin] [--tpsnopts <uint_fast16_t>]
                 [--tpsRadius <float>] [--stdDevRadius <float>]
                 [--largeRadius <float>] [--smallRadius <float>]
                 [--stddevThreshold <float>] [--in <TIN_PLANE
                 |NEAREST_NEIGHBOR|NATURAL_NEIGHBOR|STDEV_MULTISCALE
                 |TPS_RAD|TPS_PTNO>] [-b <float>] [--overlap
                 <uint_fast16_t>] [-c <unsigned int>] [-r <unsigned int>]
                 [--] [--version] [-h]


Where:

   --interp
     (OR required)  Use interpolation of the ground returns to calculate
     ground elevation
         -- OR --
   --image
     (OR required)  Use an image which defines the ground elevation.


   -o <String>,  --output <String>
     (required)  The output file.

   -e <String>,  --elevation <String>
     The input elevation image.

   -i <String>,  --input <String>
     (required)  The input SPD file.
```

```
--idxres <float>
  Resolution of the grid index used for some interpolates

--thinres <float>
  Resolution of the grid used to thin the point cloud

--ptsperbin <uint_fast16_t>
  The number of point allowed within a grid cell following thinning

--thin
  Thin the point cloud when interpolating

--tpsnopts <uint_fast16_t>
  TPS: (TPS_RAD - minimum) Number of points to be used by TPS algorithm

--tpsRadius <float>
  TPS: (TPS_PTNO - maximum) Radius used to retrieve data in TPS
  algorithm

--stdDevRadius <float>
  STDEV_MULTISCALE: Radius used to calculate the standard deviation

--largeRadius <float>
  STDEV_MULTISCALE: Large radius to be used when standard deviation is
  low

--smallRadius <float>
  STDEV_MULTISCALE: Smaller radius to be used when standard deviation is
  high

--stddevThreshold <float>
  STDEV_MULTISCALE: Standard Deviation threshold

--in <TIN_PLANE|NEAREST_NEIGHBOR|NATURAL_NEIGHBOR|STDEV_MULTISCALE
   |TPS_RAD|TPS_PTNO>
  The interpolator to be used.

-b <float>,  --binsize <float>
  Bin size for processing and output image (Default 0) - Note 0 will use
  the native SPD file bin size.
```

```
--overlap <uint_fast16_t>
  Size (in bins) of the overlap between processing blocks (Default 10)

-c <unsigned int>,  --blockcols <unsigned int>
  Number of columns within a block (Default 0) - Note values greater
  than 1 result in a non-sequencial SPD file.

-r <unsigned int>,  --blockrows <unsigned int>
  Number of rows within a block (Default 100)

--,  --ignore_rest
  Ignores the rest of the labeled arguments following this flag.

--version
  Displays version information and exits.

-h,  --help
  Displays usage information and exits.


Define the height field within pulses and points: spddefheight
```

5.3 Shift Elevation (Datum) – spdelevation

5.3.1 Help File

```
spdelevation SPDLib 3.0.0, Copyright (C) 2012 Sorted Pulse Library (SPD)
This program comes with ABSOLUTELY NO WARRANTY. This is free software,
and you are welcome to redistribute it under certain conditions; See
website (http://www.spdlib.org). Bugs are to be reported on the trac
or directly to spdlib-develop@lists.sourceforge.net

USAGE:

   spdelevation  {--constant <double>|--variable <string>} {--add|--minus}
                 -o <String> -i <String> [-c <unsigned int>] [-r <unsigned
                 int>] [--] [--version] [-h]
```

```
Where:

   --constant <double>
     (OR required)  Alter pulse elevation by a constant amount
         -- OR --
   --variable <string>
     (OR required)  Alter pulse elevation by a variable amount defined
     using an image


   --add
     (OR required)  Add offset
         -- OR --
   --minus
     (OR required)  Remove offset


   -o <String>,  --output <String>
     (required)  The output SPD file.

   -i <String>,  --input <String>
     (required)  The input SPD file.

   -c <unsigned int>,  --blockcols <unsigned int>
     Number of columns within a block (Default 0) - Note values greater
     than 1 result in a non-sequencial SPD file.

   -r <unsigned int>,  --blockrows <unsigned int>
     Number of rows within a block (Default 100)

   --,  --ignore_rest
     Ignores the rest of the labeled arguments following this flag.

   --version
     Displays version information and exits.

   -h,  --help
     Displays usage information and exits.
```

```
Alter the elevation of the pulses: spdelevation
```

5.4 Interpolate Raster Surfaces – spdinterp

5.4.1 Help File

```
spdinterp SPDLib 3.0.0, Copyright (C) 2012 Sorted Pulse Library (SPD)
This program comes with ABSOLUTELY NO WARRANTY. This is free software,
and you are welcome to redistribute it under certain conditions; See
website (http://www.spdlib.org). Bugs are to be reported on the trac
or directly to spdlib-develop@lists.sourceforge.net

USAGE:

   spdinterp  {--dtm|--chm|--dsm} {--topo|--height} -o <String> -i <String>
              [--idxres <float>] [--thinres <float>] [--ptsperbin
              <uint_fast16_t>] [--thin] [--tpsnopts <uint_fast16_t>]
              [--tpsRadius <float>] [--stdDevRadius <float>] [--largeRadius
              <float>] [--smallRadius <float>] [--stddevThreshold <float>]
              [--in <TIN_PLANE|NEAREST_NEIGHBOR|NATURAL_NEIGHBOR
              |STDEV_MULTISCALE|TPS_RAD|TPS_PTNO>] [-f <string>] [-b
              <float>] [--overlap <uint_fast16_t>] [-c <unsigned int>] [-r
              <unsigned int>] [--] [--version] [-h]


Where:

   --dtm
     (OR required)  Interpolate a DTM image
         -- OR --
   --chm
     (OR required)  Interpolate a CHM image.
         -- OR --
   --dsm
     (OR required)  Interpolate a DSM image.


   --topo
```

```
  (OR required)  Use topographic elevation
      -- OR --
--height
  (OR required)  Use height above ground elevation.


-o <String>,  --output <String>
  (required)  The output SPD file.

-i <String>,  --input <String>
  (required)  The input SPD file.

--idxres <float>
  Resolution of the grid index used for some interpolates

--thinres <float>
  Resolution of the grid used to thin the point cloud

--ptsperbin <uint_fast16_t>
  The number of point allowed within a grid cell following thinning

--thin
  Thin the point cloud when interpolating

--tpsnopts <uint_fast16_t>
  TPS: (TPS_RAD - minimum) Number of points to be used by TPS algorithm

--tpsRadius <float>
  TPS: (TPS_PTNO - maximum) Radius used to retrieve data in TPS
  algorithm

--stdDevRadius <float>
  STDEV_MULTISCALE: Radius used to calculate the standard deviation

--largeRadius <float>
  STDEV_MULTISCALE: Large radius to be used when standard deviation is
  low

--smallRadius <float>
  STDEV_MULTISCALE: Smaller radius to be used when standard deviation is
  high
```

```
--stddevThreshold <float>
  STDEV_MULTISCALE: Standard Deviation threshold

--in <TIN_PLANE|NEAREST_NEIGHBOR|NATURAL_NEIGHBOR|STDEV_MULTISCALE
   |TPS_RAD|TPS_PTNO>
  The interpolator to be used.

-f <string>,  --format <string>
  Image format (GDAL driver string), Default is ENVI.

-b <float>,  --binsize <float>
  Bin size for processing and output image (Default 0) - Note 0 will use
  the native SPD file bin size.

--overlap <uint_fast16_t>
  Size (in bins) of the overlap between processing blocks (Default 10)

-c <unsigned int>,  --blockcols <unsigned int>
  Number of columns within a block (Default 0) - Note values greater
  than 1 result in a non-sequencial SPD file.

-r <unsigned int>,  --blockrows <unsigned int>
  Number of rows within a block (Default 100)

--,  --ignore_rest
  Ignores the rest of the labeled arguments following this flag.

--version
  Displays version information and exits.

-h,  --help
  Displays usage information and exits.


Interpolate a raster elevation surface: spdinterp
```

5.5 Classify Ground Returns

5.5.1 Progressive Morphology Filter – spdpmfgrd

The spdpmfgrd command is an implementation of the progressive morphological filter algorithm of (Zhang et al., 2003).

The algorithm works by generating an initial minimum return raster surface at the bin resolution of the SPD-file. Circulate morphological operators of a range of scales (starting at –initfilter size and going up in increments of 1 to –maxfilter). At each scale a morphological closing (erosion + dilation) operation is performed. The new height value from the morphological operator is kept if it is above the elevation difference threshold (initialised to –initelev with maximum value of –maxelev) where the elevation difference threshold is increased between threshold using the –slope parameter. Finally, to classify the LiDAR returns a buffer threshold (–grd) is used such that all the point within the buffer or below the surface are classified as ground. Before the classification, by default, a median filter is applied to the final raster surface. The size of the median filter can be specified with the –medianfilter parameter and the use of the median filter can be turned off using the –nomedian filter parameter.

Examples

Default Command

Under most circumstances the default parameters for the algorithm will be fit for purpose and it is recommend that you try these first with the following command. But be careful that the bin size used within SPD is not too large as the processing will be at this resolution (see the next example to change).

```
spdpmfgrd -i liukdr_20090601_1m.spd -o liukdr_20090601_1m_grd.spd
```

Processing Bin Size

The processing bin size can be changed (by default it is the bin size of the SPD file) as shown below:

```
spdpmfgrd -b 1 -i liukdr_20090601_10m.spd -o liukdr_20090601_10m_grd.spd
```

Using Block and Overlap Parameters

Notice that as with the other similar commands the block and overlap sizes can also be adjusted, as shown below:

```
spdpmfgrd --overlap 10 -c 100 -r 100  -i liukdr_20090601_1m.spd \
-o liukdr_20090601_1m_grd.spd
```

Changing the Filter Sizes

```
spdpmfgrd --initfilter 2 --maxfilter 12 -i liukdr_20090601_1m.spd \
-o liukdr_20090601_1m_grd.spd
```

Changing the Elevation Difference Threshold

```
spdpmfgrd --initelev 0.3  --maxelev 5 --slope 0.3 -i liukdr_20090601_1m.spd \
-o liukdr_20090601_1m_grd.spd
```

Change Ground Return Classification Threshold

The ground return threshold is the distance the points being classified as ground can be from the identified ground surface raster.

The default is 30 cm:

```
spdpmfgrd --grd 0.3 -i liukdr_20090601_1m.spd -o liukdr_20090601_1m_grd.spd
```

Here a distance of 1 m is used:

```
spdpmfgrd --grd 1.0 -i liukdr_20090601_1m.spd -o liukdr_20090601_1m_grd.spd
```

Filter points depending on class

The –class option allows the filter to be applied to returns of a particular class (i.e., if you have ground returns classified but it needs tidying up etc). It's useful for TLS as it can take a thick slice with mcc algorithm and then use the PMF algorithm to tidy that result up to get a good overall ground classification.

Help File

```
spdpmfgrd SPDLib 3.0.0, Copyright (C) 2012 Sorted Pulse Library (SPD)
This program comes with ABSOLUTELY NO WARRANTY. This is free software,
and you are welcome to redistribute it under certain conditions; See
website (http://www.spdlib.org). Bugs are to be reported on the trac
or directly to spdlib-develop@lists.sourceforge.net

USAGE:

   spdpmfgrd  -o <String> -i <String> [--gdal <string>] [--class
              <uint_fast16_t>] [--image] [--medianfilter <uint_fast16_t>]
              [--nomedian] [--grd <float>] [--maxelev <float>] [--initelev
              <float>] [--slope <float>] [--maxfilter <uint_fast16_t>]
              [--initfilter <uint_fast16_t>] [--overlap <uint_fast16_t>]
              [-b <float>] [-c <unsigned int>] [-r <unsigned int>] [--]
              [--version] [-h]


Where:

   -o <String>,  --output <String>
     (required)  The output file.

   -i <String>,  --input <String>
     (required)  The input SPD file.

   --gdal <string>
     Provide the GDAL driver format (Default ENVI), Erdas Imagine is HFA,
     KEA is KEA

   --class <uint_fast16_t>
```

```
  Only use points of particular class

--image
  If set an image of the output surface will be generated rather than
  classifying the points (useful for debugging and parameter selection)

--medianfilter <uint_fast16_t>
  Size of the median filter (half size i.e., 3x3 is 1) (Default 2)

--nomedian
  Do not run a median filter on generated surface (before classifying
  ground point or export)

--grd <float>
  Threshold for deviation from identified ground surface for classifying
  the ground returns (Default 0.3)

--maxelev <float>
  Maximum elevation difference threshold (Default 5)

--initelev <float>
  Initial elevation difference threshold (Default 0.3)

--slope <float>
  Slope parameter related to terrain (Default 0.3)

--maxfilter <uint_fast16_t>
  Maximum size of the filter (Default 7)

--initfilter <uint_fast16_t>
  Initial size of the filter (note this is half the filter size so a 3x3
  will be 1 and 5x5 will be 2) (Default 1)

--overlap <uint_fast16_t>
  Size (in bins) of the overlap between processing blocks (Default 10)

-b <float>,  --binsize <float>
  Bin size for processing and output image (Default 0) - Note 0 will use
  the native SPD file bin size.

-c <unsigned int>,  --blockcols <unsigned int>
```

```
    Number of columns within a block (Default 0) - Note values greater
    than 1 result in a non-sequencial SPD file.

  -r <unsigned int>,  --blockrows <unsigned int>
    Number of rows within a block (Default 100)

  --,  --ignore_rest
    Ignores the rest of the labeled arguments following this flag.

  --version
    Displays version information and exits.

  -h,  --help
    Displays usage information and exits.


  Classifies the ground returns using the progressive morphology
  algorithm: spdpmfgrd
```

5.5.2 Multi-Curvature Classifier – spdmccgrd

The multi-scale curvature algorithm (Evans and Hudak, 2007) was created at the US Forest Service and does a good job at classifying ground returns under a forest canopy while retaining the terrain but it does not differentiate the buildings.

Examples

Default Command

Under most circumstances the default parameters for the algorithm will be fit for purpose and it is recommend that you try these first with the following command.

```
spdmccgrd -i liukdr_20090601_1m.spd -o liukdr_20090601_1m_grd.spd
```

Curvature Parameter

The curvature threshold is the parameter which decides whether a point is classified as a ground or not. If the curvature threshold is higher then more ground returns will be accepted. There are three parameters which can be edited to control the curvature parameter –initcurvetol, –mincurvetol and –stepcurvetol. The initial survature tolerance (–initcurvetol) has a default value of 1, while the minimum curvature has a default of 0.1 and the step between scales is 0.5. To easily change the behaviour of the algorithm is it best to change the initial curvature parameter first, as shown below.

```
spdmccgrd --initcurvetol 2 -i liukdr_20090601_1m.spd -o liukdr_20090601_1m_grd.spd
```

Filter points depending on class

The –class option allows the filter to be applied to returns of a particular class (i.e., if you have ground returns classified but it needs tidying up etc). It's useful as it can take a thick slice with PMF algorithm and then use the MCC algorithm to tidy that result up to get a good overall ground classification. Class 3 is ground.

```
spdmccgrd -i --class 3 liukdr_20090601_1m.spd -o liukdr_20090601_1m_grd.spd
```

Help File

```
spdmccgrd SPDLib 3.0.0, Copyright (C) 2012 Sorted Pulse Library (SPD)
This program comes with ABSOLUTELY NO WARRANTY. This is free software,
and you are welcome to redistribute it under certain conditions; See
website (http://www.spdlib.org). Bugs are to be reported on the trac
or directly to spdlib-develop@lists.sourceforge.net

USAGE:

   spdmccgrd  -o <String> -i <String> [--thresofchangemultireturn] [--class
              <uint_fast16_t>] [--median] [--thresofchange <float>]
              [--filtersize <uint_fast16_t>] [--interpnumpts
              <uint_fast16_t>] [--interpmaxradius <float>] [--stepcurvetol
              <float>] [--mincurvetol <float>] [--initcurvetol <float>]
```

```
            [--scalegaps <float>] [--numofscalesbelow <uint_fast16_t>]
            [--numofscalesabove <uint_fast16_t>] [--initscale <float>]
            [--overlap <uint_fast16_t>] [-b <float>] [-c <unsigned int>]
            [-r <unsigned int>] [--] [--version] [-h]


Where:

   -o <String>,  --output <String>
     (required)  The output SPD file.

   -i <String>,  --input <String>
     (required)  The input SPD file.

   --thresofchangemultireturn
     Use only multiple return pulses to calculate the amount of change
     between iterations.

   --class <uint_fast16_t>
     Only use points of particular class

   --median
     Use a median filter to smooth the generated raster instead of a (mean)
     averaging filter.

   --thresofchange <float>
     The threshold for the  (Default = 0.1)

   --filtersize <uint_fast16_t>
     The size of the smoothing filter (half size i.e., 3x3 is 1; Default =
     1).

   --interpnumpts <uint_fast16_t>
     The number of points used for the TPS interpolation (Default = 16)

   --interpmaxradius <float>
     Maximum search radius for the TPS interpolation (Default = 20)

   --stepcurvetol <float>
     Iteration step curveture tolerance parameter (Default = 0.5)
```

```
--mincurvetol <float>
  Minimum curveture tolerance parameter (Default = 0.1)

--initcurvetol <float>
  Initial curveture tolerance parameter (Default = 1)

--scalegaps <float>
  Gap between increments in scale (Default = 0.5)

--numofscalesbelow <uint_fast16_t>
  The number of scales below the init scale to be used (Default = 1)

--numofscalesabove <uint_fast16_t>
  The number of scales above the init scale to be used (Default = 1)

--initscale <float>
  Initial processing scale, this is usually the native resolution of the
  data.

--overlap <uint_fast16_t>
  Size (in bins) of the overlap between processing blocks (Default 10)

-b <float>,  --binsize <float>
  Bin size for processing and output image (Default 0) - Note 0 will use
  the native SPD file bin size.

-c <unsigned int>,  --blockcols <unsigned int>
  Number of columns within a block (Default 0) - Note values greater
  than 1 result in a non-sequencial SPD file.

-r <unsigned int>,  --blockrows <unsigned int>
  Number of rows within a block (Default 100)

--,  --ignore_rest
  Ignores the rest of the labeled arguments following this flag.

--version
  Displays version information and exits.

-h,  --help
  Displays usage information and exits.
```

```
Classifies the ground returns using the multiscale curvature algorithm:
spdmccgrd
```

5.5.3 Parameter Free Filter – spdpffgrd

Help File

```
spdpffgrd SPDLib 3.0.0, Copyright (C) 2012 Sorted Pulse Library (SPD)
This program comes with ABSOLUTELY NO WARRANTY. This is free software,
and you are welcome to redistribute it under certain conditions; See
website (http://www.spdlib.org). Bugs are to be reported on the trac
or directly to spdlib-develop@lists.sourceforge.net

USAGE:

   spdpffgrd  -o <String> -i <String> [--gdal <string>] [--class
              <uint_fast16_t>] [--morphmin] [--image] [-m <uint_fast16_t>]
              [-f <uint_fast32_t>] [--tophatscales <bool>] [-t
              <uint_fast32_t>] [-s <uint_fast32_t>] [-k <uint_fast32_t>]
              [--grd <float>] [--overlap <uint_fast16_t>] [-b <float>] [-c
              <unsigned int>] [-r <unsigned int>] [--] [--version] [-h]


Where:

   -o <String>,  --output <String>
     (required)  The output file.

   -i <String>,  --input <String>
     (required)  The input SPD file.

   --gdal <string>
     Provide the GDAL driver format (Default ENVI), Erdas Imagine is HFA,
     KEA is KEA

   --class <uint_fast16_t>
     Only use points of particular class
```

```
--morphmin
  Apply morphological opening and closing to remove multiple path
  returns (note this can remove really ground returns).

--image
  If set an image of the output surface will be generated rather than
  classifying the points (useful for debugging and parameter selection)

-m <uint_fast16_t>,  --mpd <uint_fast16_t>
  Minimum point density in block to use for surface estimation - default
  40

-f <uint_fast32_t>,  --tophatfactor <uint_fast32_t>
  How quickly the tophat window reduces through the resolution, higher
  numbers reduce size quicker - default 2

--tophatscales <bool>
  Whether the tophat window size decreases through the resolutions -
  default true

-t <uint_fast32_t>,  --tophatstart <uint_fast32_t>
  Starting window size (actually second, first is always 1) for tophat
  transforms, must be >= 2, setting this too big can cause segfault! -
  default 4

-s <uint_fast32_t>,  --stddev <uint_fast32_t>
  Number of standard deviations used in classification threshold -
  default 3

-k <uint_fast32_t>,  --kvalue <uint_fast32_t>
  Number of stddevs used for control point filtering - default 3

--grd <float>
  Threshold for deviation from identified ground surface for classifying
  the ground returns (Default 0.3)

--overlap <uint_fast16_t>
  Size (in bins) of the overlap between processing blocks (Default 10)

-b <float>,  --binsize <float>
```

```
    Bin size for processing and output image (Default 0) - Note 0 will use
    the native SPD file bin size.

  -c <unsigned int>,  --blockcols <unsigned int>
    Number of columns within a block (Default 0) - Note values greater
    than 1 result in a non-sequencial SPD file.

  -r <unsigned int>,  --blockrows <unsigned int>
    Number of rows within a block (Default 100)

  --,  --ignore_rest
    Ignores the rest of the labeled arguments following this flag.

  --version
    Displays version information and exits.

  -h,  --help
    Displays usage information and exits.


  Classifies the ground returns using a parameter-free filtering
  algorithm: spdpffgrd
```

5.5.4 Polynomial Ground Filter – spdpolygrd

Help File

```
spdpolygrd SPDLib 3.0.0, Copyright (C) 2012 Sorted Pulse Library (SPD)
This program comes with ABSOLUTELY NO WARRANTY. This is free software,
and you are welcome to redistribute it under certain conditions; See
website (http://www.spdlib.org). Bugs are to be reported on the trac
or directly to spdlib-develop@lists.sourceforge.net

USAGE:

  spdpolygrd  [--iters <int>] [--degree <int>] [--grdthres <float>] [-b
              <float>] [-c <unsigned int>] [-r <unsigned int>] [--]
              [--version] [-h] <string> ...
```

```
Where:

   --iters <int>
     Number of iterations for polynomial surface to converge on ground
     (Default = 2).

   --degree <int>
     Order of polynomial surface (Default = 1).

   --grdthres <float>
     Threshold for how far above the interpolated ground surface a return
     can be and be reclassified as ground (Default = 0.25).

   -b <float>,  --binsize <float>
     Bin size for processing and output image (Default 0) - Note 0 will use
     the native SPD file bin size.

   -c <unsigned int>,  --blockcols <unsigned int>
     Number of columns within a block (Default 0) - Note values greater
     than 1 result in a non-sequencial SPD file.

   -r <unsigned int>,  --blockrows <unsigned int>
     Number of rows within a block (Default 100)

   --,  --ignore_rest
     Ignores the rest of the labeled arguments following this flag.

   --version
     Displays version information and exits.

   -h,  --help
     Displays usage information and exits.

   <string>  (accepted multiple times)
     File names for the input files


   Classify ground returns using a surface fitting algorithm: spdpolygrd
```

5.5.5 Combining Filters

Another option which can improve the ground return classification is to combine more than one filtering algorithm to take advantage of their particular strengths and weaknesses. A particularly useful combination is to first run the PMF algorithm where a 'thick' slice is taken (e.g., 1 or 2 metres above the raster surface) and then the MCC is applied to find the ground returns (using the –class 3 option).

```
spdpmfgrd --grd 1.25 -i liukdr_20090601_1m.spd -o liukdr_20090601_1m_grdpmf.spd
spdmccgrd -i --class 3 liukdr_20090601_1m_grdpmf.spd -o liukdr_20090601_1m_grdpmfmcc.spd
```

5.6 Calculate Point Cloud Metrics – spdmetrics

5.6.1 Help File

```
spdmetrics SPDLib 3.0.0, Copyright (C) 2012 Sorted Pulse Library (SPD)
This program comes with ABSOLUTELY NO WARRANTY. This is free software,
and you are welcome to redistribute it under certain conditions; See
website (http://www.spdlib.org). Bugs are to be reported on the trac
or directly to spdlib-develop@lists.sourceforge.net

USAGE:

   spdmetrics  {--image|--vector|--ascii} [-v <String>] -m <String> -o
               <String> -i <String> [-f <string>] [-b <float>] [-c
               <unsigned int>] [-r <unsigned int>] [--] [--version] [-h]


Where:

   --image
     (OR required)  Run metrics with image output
         -- OR --
   --vector
     (OR required)  Run metrics with vector output
         -- OR --
   --ascii
```

```
  (OR required)  Run metrics with ASCII output


-v <String>,  --vectorfile <String>
  The input vector file.

-m <String>,  --metricsxml <String>
  (required)  The output SPD file.

-o <String>,  --output <String>
  (required)  The output file.

-i <String>,  --input <String>
  (required)  The input SPD file.

-f <string>,  --format <string>
  Image format (GDAL driver string), Default is ENVI.

-b <float>,  --binsize <float>
  Bin size for processing and output image (Default 0) - Note 0 will use
  the native SPD file bin size.

-c <unsigned int>,  --blockcols <unsigned int>
  Number of columns within a block (Default 0) - Note values greater
  than 1 result in a non-sequencial SPD file.

-r <unsigned int>,  --blockrows <unsigned int>
  Number of rows within a block (Default 100)

--,  --ignore_rest
  Ignores the rest of the labeled arguments following this flag.

--version
  Displays version information and exits.

-h,  --help
  Displays usage information and exits.


Calculate metrics : spdmetrics
```

5.7 Generate Vertical Profiles – spdprofile

5.7.1 Help File

```
spdprofile SPDLib 3.0.0, Copyright (C) 2012 Sorted Pulse Library (SPD)
This program comes with ABSOLUTELY NO WARRANTY. This is free software,
and you are welcome to redistribute it under certain conditions; See
website (http://www.spdlib.org). Bugs are to be reported on the trac
or directly to spdlib-develop@lists.sourceforge.net

USAGE:

   spdprofile  -o <String> -i <String> [-f <std::string>] [-b <float>] [-c
               <unsigned int>] [-r <unsigned int>] [-n <unsigned int>] [-t
               <unsigned int>] [-m <float>] [-w <unsigned int>] [--order
               <unsigned int>] [--smooth] [--] [--version] [-h]


Where:

   -o <String>,  --output <String>
     (required)  The output file.

   -i <String>,  --input <String>
     (required)  The input SPD file.

   -f <std::string>,  --format <std::string>
     Image format (GDAL driver string), Default is ENVI.

   -b <float>,  --binsize <float>
     Bin size for processing and output image (Default 0) - Note 0 will use
     the native SPD file bin size.

   -c <unsigned int>,  --blockcols <unsigned int>
     Number of columns within a block (Default 0) - Note values greater
     than 1 result in a non-sequencial SPD file.

   -r <unsigned int>,  --blockrows <unsigned int>
     Number of rows within a block (Default 100)
```

```
-n <unsigned int>,  --numbins <unsigned int>
  The number of bins within the profile (Default: 20).

-t <unsigned int>,  --topheight <unsigned int>
  The highest bin of the profile (Default: 40).

-m <float>,  --minheight <float>
  The the height below which points are ignored (Default: 0).

-w <unsigned int>,  --window <unsigned int>
  The window size ((w*2)+1) used for the smoothing filter (Default: 3).

--order <unsigned int>
  The order of the polynomial used to smooth the profile (Default: 3).

--smooth
  Apply a Savitzky Golay smoothing to the profiles.

--,  --ignore_rest
  Ignores the rest of the labeled arguments following this flag.

--version
  Displays version information and exits.

-h,  --help
  Displays usage information and exits.


Generate vertical profiles: spdprofile
```

5.8 Remove Noise – spdrmnoise

5.8.1 Help File

```
spdrmnoise SPDLib 3.0.0, Copyright (C) 2012 Sorted Pulse Library (SPD)
This program comes with ABSOLUTELY NO WARRANTY. This is free software,
and you are welcome to redistribute it under certain conditions; See
website (http://www.spdlib.org). Bugs are to be reported on the trac
```

```
or directly to spdlib-develop@lists.sourceforge.net

USAGE:

   spdrmnoise  -o <String> -i <String> [--rellow <float>] [--relup <float>]
               [--abslow <float>] [--absup <float>] [-c <unsigned int>] [-r
               <unsigned int>] [--] [--version] [-h]


Where:

   -o <String>,  --output <String>
     (required)  The output SPD file.

   -i <String>,  --input <String>
     (required)  The input SPD file.

   --rellow <float>
     Relative (to median) lower threshold for returns which are to be
     removed.

   --relup <float>
     Relative (to median) upper threshold for returns which are to be
     removed.

   --abslow <float>
     Absolute lower threshold for returns which are to be removed.

   --absup <float>
     Absolute upper threshold for returns which are to be removed.

   -c <unsigned int>,  --blockcols <unsigned int>
     Number of columns within a block (Default 0) - Note values greater
     than 1 result in a non-sequencial SPD file.

   -r <unsigned int>,  --blockrows <unsigned int>
     Number of rows within a block (Default 100)

   --,  --ignore_rest
     Ignores the rest of the labeled arguments following this flag.
```

```
--version
  Displays version information and exits.

-h,  --help
  Displays usage information and exits.


Remove vertical noise from LiDAR datasets: spdrmnoise
```

Chapter 6

Other Useful Scripts

6.1 Generate Batch Processing Script – spdbatchgen.py

6.1.1 Help File

```
usage: spdbatchgen.py [-h] [-i XMLOUTLINEFILE] [-o OUTFILE] [-p PATH]
                      [-e EXTENSION] [-d DIR] [-r RECURSE] [-t OUTPUTTYPE]

optional arguments:
  -h, --help            show this help message and exit
  -i XMLOUTLINEFILE, --input XMLOUTLINEFILE
                        Input shell template
  -o OUTFILE, --output OUTFILE
                        Output shell
  -p PATH, --path PATH  Path used to replace '$PATH' in shell template.
  -e EXTENSION, --ext EXTENSION
                        File extension to search for.
  -d DIR, --dir DIR     Directory to search for files with the extension
                        specified by '-e'.
  -r RECURSE, --recurse RECURSE
                        Recurse into sub directories within input directory
                        (yes | no).
  -t OUTPUTTYPE, --output_type OUTPUTTYPE
```

```
                        Output type (single - single shell file | multiple -
                        separate shell file for each file found).
```

6.2 Populate a template with multiple files – spdcmdgen.py

6.2.1 Help File

```
Usage: spdcmdgen.py [options]

Options:
  -h, --help            show this help message and exit
  -i INPUTFILE, --input=INPUTFILE
                        Input shell script template
  -o OUTPUTFILE, --output=OUTPUTFILE
                        Output shell script file
  -b BASENAMES, --base=BASENAMES
                        Output file base file names ('$FILENAME1',
                        '$FILENAME2'...'$FILENAMEn')
  -p OUTFILEPATH, --path=OUTFILEPATH
                        Output file path ('$PATH')
  -f FILES, --file=FILES
                        Input files to be replaced using $FILEPATHx for files
                        in order (e.g., $FILEPATH1, $FILEPATH2 ...
                        $FILEPATHn).
```

6.3 Build a merge command – spdbuildmergecmd.py

6.3.1 Help File

```
Usage: spdbuildmergecmd.py [options]

Options:
  -h, --help            show this help message and exit
  -d INPUTDIR, --dir=INPUTDIR
```

```
                        Input Directory
  -e EXTENSION, --ext=EXTENSION
                        Input file extension
  -c OUTPUTCLASS, --class=OUTPUTCLASS
                        Output class of points
  -o OUTPUTSPD, --out=OUTPUTSPD
                        Output SPD file from spdmerge
  --spdin=SPDINPUT      SPD input file type
  --ignorechecks        Turn on SPDMerge option to ignore input file checks.
```

6.4 Build extract tiles commands – spdbuildtileextractcmd.py

6.4.1 Help File

```
spdbuildtileextractcmd.py builds extract tiled commands
for the SPDLib library.

This script was distributed with version 3.0.0 of the SPDLib library.
For maintenance email spdlib-develop@lists.sourceforge.net
usage: spdbuildtileextractcmd.py [-h] [-f FILELIST] [-t TILES] [-b OUTPUTBASE]
                                 [-o OUTPUT] [-d]

optional arguments:
  -h, --help            show this help message and exit
  -f FILELIST, --filelist FILELIST
                        A text file with a list of input files, one per line.
  -t TILES, --tiles TILES
                        Tiles XML file.
  -b OUTPUTBASE, --outputbase OUTPUTBASE
                        The base path for the tiles.
  -o OUTPUT, --output OUTPUT
                        Output file listing the commands for extracting the
                        tiles.
  -d, --deltile         Set to include option to delete tiles with no data.
```

Part III

GUI Tools

Chapter 7

SPD Points Viewer

The SPDPointsViewer provides a simple to use and quick visualisation of the LiDAR data, both discrete returns and full waveform. It is particularly useful for understanding the data you are working with and/or checking that the result from an algorithm are correct (e.g., classification of ground returns).

To use the viewer run the `SPDPointsViewer` command from the Terminal or if you have it installed with a shortcut (e.g., under Windows or Mac OSX) then run that shortcut. The viewer will open as shown in Figure 7.1

To load data within the viewer you need to select `File > Open` and the select the SPD file of interest.

You will now be presented with an overview of the SPD file which is the 2D spatial index of the file. To load data to view it in 3D you need to draw a box on to the over view image and the data within that box will be loaded, Figure 7.2.

Once the 3D data is loaded (Figure 7.3) it will automatically move to the 3D viewer tab (note to move back you need to manually change the tab you are viewing). You can now drag the data around within the viewer using the cursor or the sliders on the lefthand side.

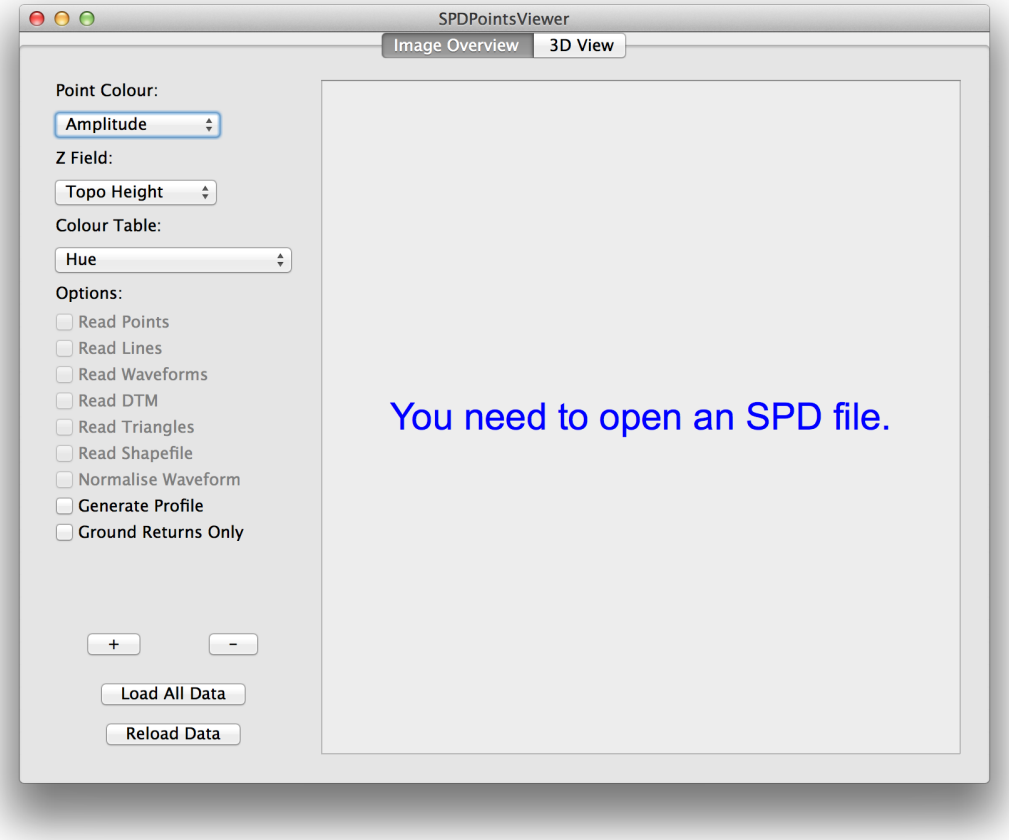


Figure 7.1: The SPDPointsViewer application on startup.

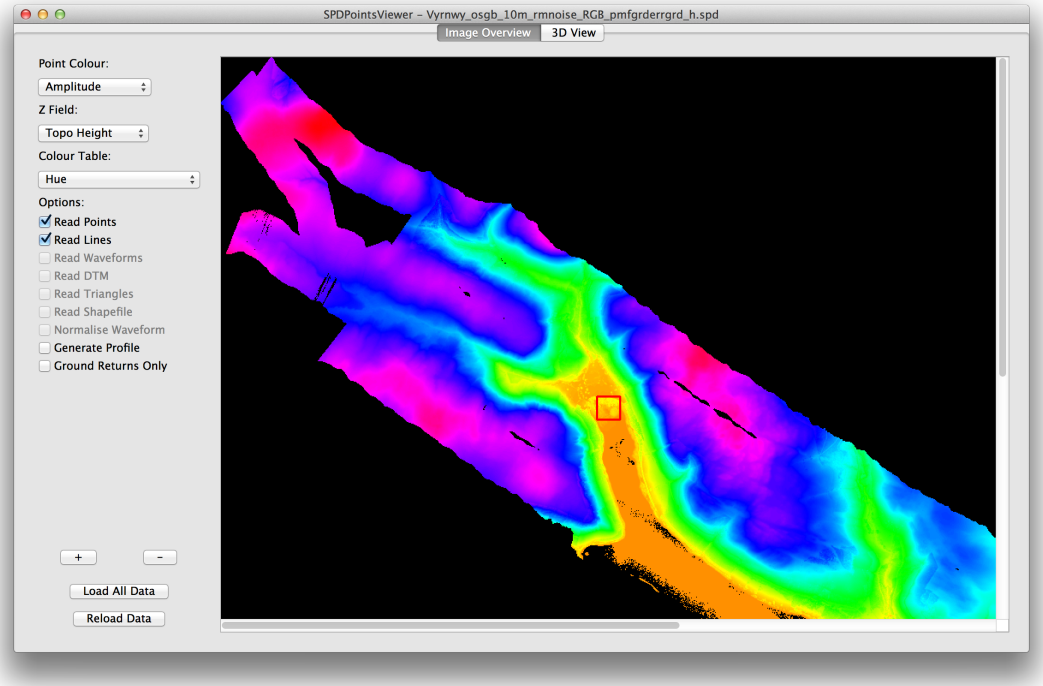


Figure 7.2: The SPDPointsViewer application with the overview.

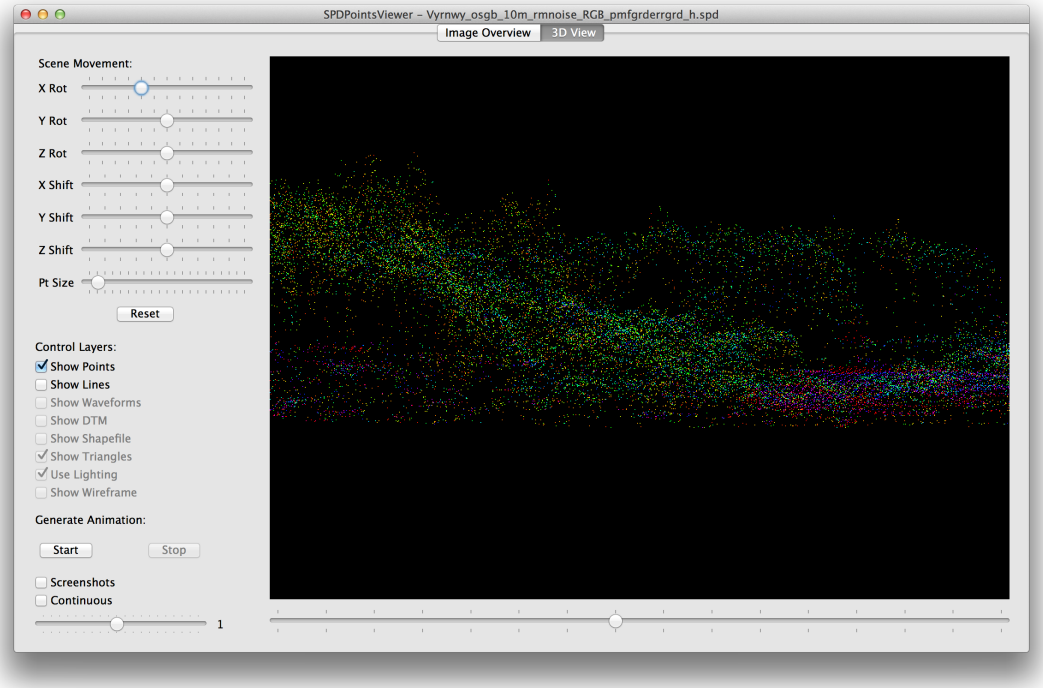


Figure 7.3: The SPDPointsViewer application with the 3D data loaded.

Part IV

Tutorials

Chapter 8

Tutorials Background

8.1 Which tutorial to look at?

First of all you need to consider your data needs, refer to Figure 8.1 as to which tutorials are explicitly useful for you. However, I would recommend you start with Tutorial 1 (Chapter 9) which steps through the main useful individual commands on a set of datasets.

8.2 Workflow

Figure 8.2 provides a suggested workflow which might be a useful guide to follow or to use parts of depending on your application. By following through tutorial 1 the commands used within this workflow will be demonstrated.

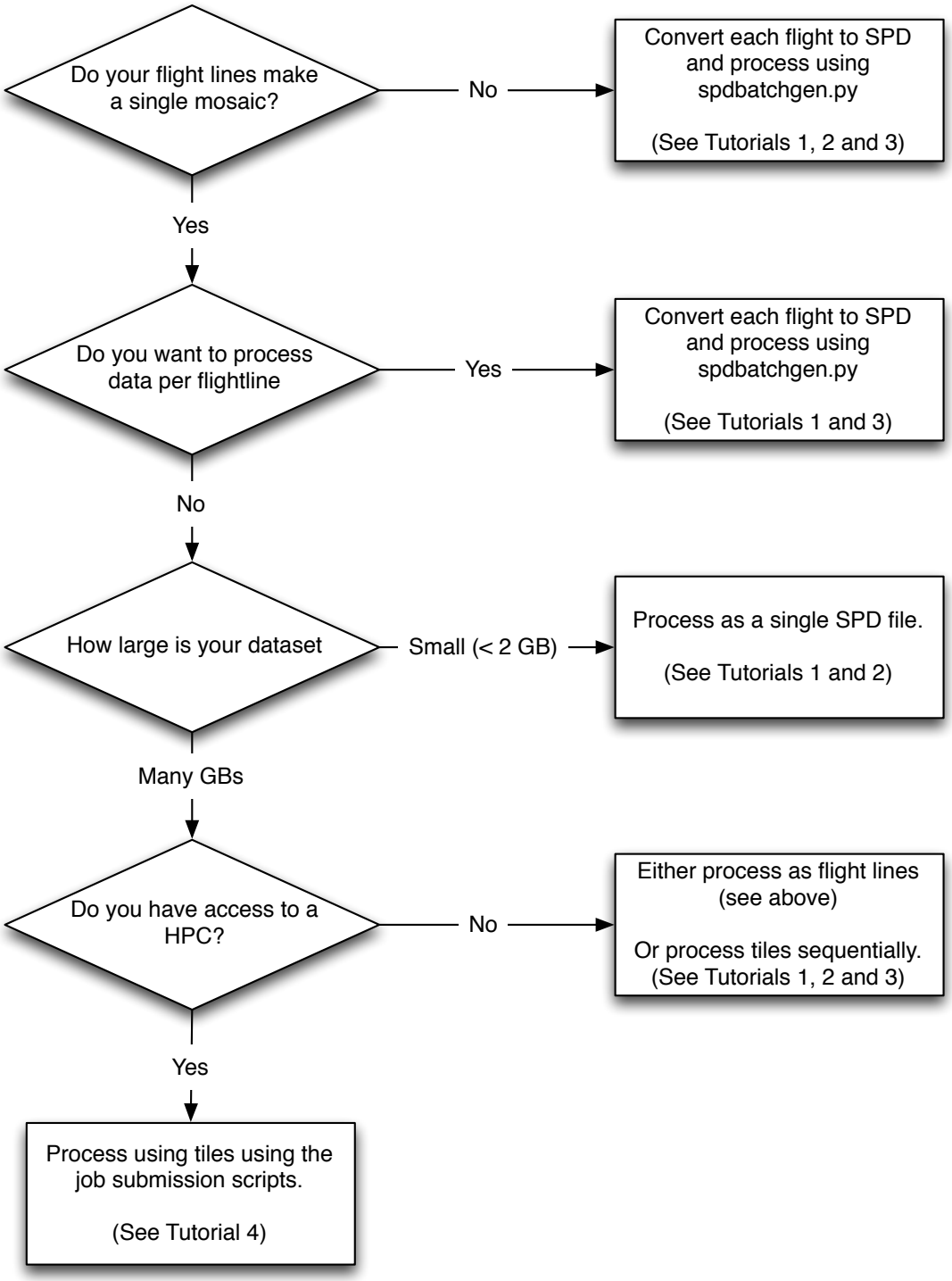


Figure 8.1: Which tutorials are useful?

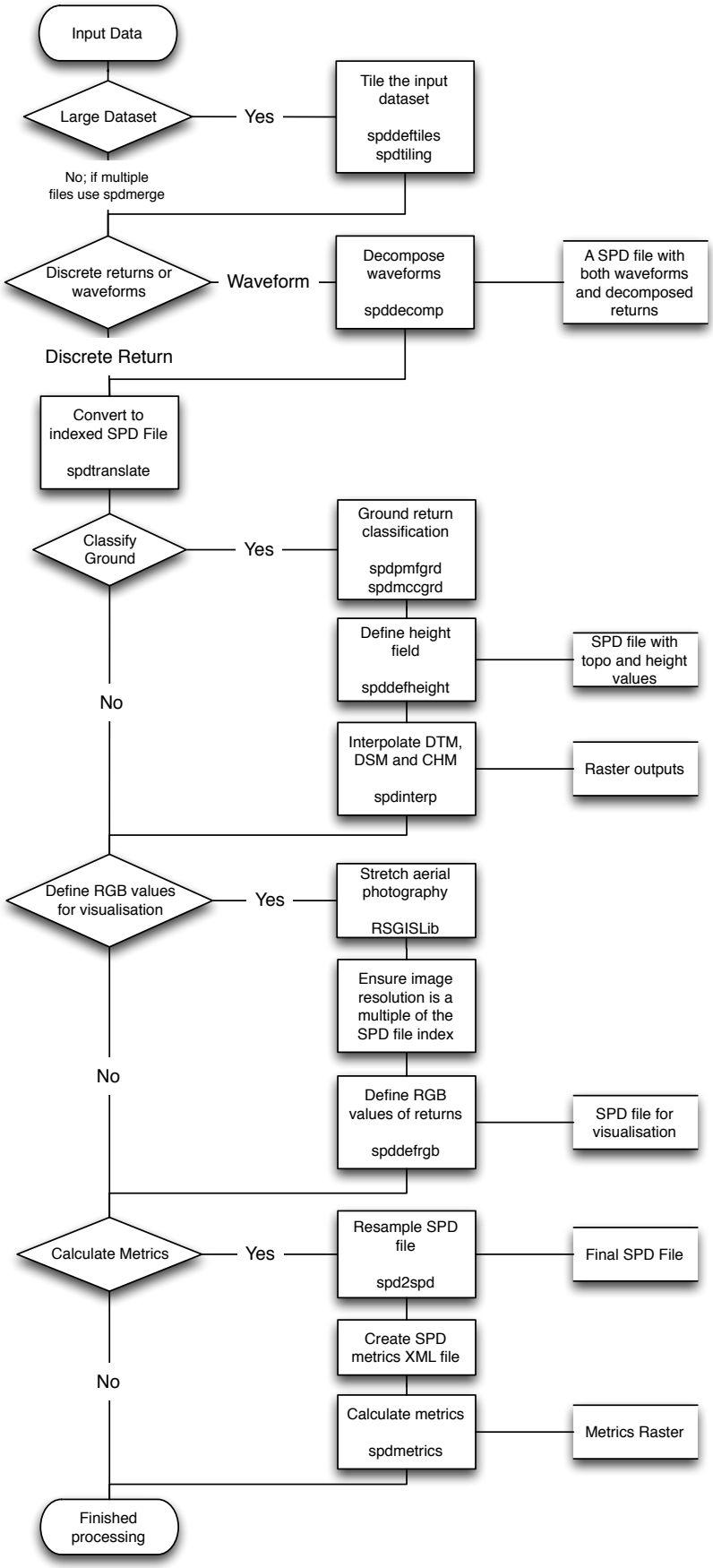


Figure 8.2: Possible processing chain.

Chapter 9

Tutorial 1: Single Commands

The following set of examples go through the basic functionality of set of functionality within SPDLib, which you will need to take a LiDAR dataset and produce results. After working through these examples you should be able to:

1. Convert LAS files to SPD files.
2. Merge input files to create a larger merged SPD file.
3. Interpolate elevation products.
4. Classify ground returns.
5. Calculate vegetation metrics and calculate forest biomass.
6. Decompose full waveform LiDAR data using a Gaussian decomposition.
7. Colour returns using an aerial photograph for visualisation.

9.1 Convert to SPD

The first step in being able to use the SPDLib software is to convert your data to the SPD file format (Bunting et al., 2013b). All processing within the SPDLib software requires the data to either be within an SPD file. There are two versions of the SPD format, the first is referred to as 'Unsorted Pulse Data' (UPD) and

the second is the 'Sorted Pulse Data' (SPD) format. The difference between the formats is that UPD does not contain a spatial index but otherwise all the same information is represented. Like a LAS file there is no ordering of the pulses forces on the data in a UPD but most commonly it would be expected to be ordered in time. An SPD file contains a spatial index and this is the format that is required for most commands as the spatial index speeds up the processing and allows for spatial selections to be made.

The simplest command for converting a LAS file to an SPD file is shown below, run this command:

```
spdtranslate --if LAS --of SPD -b 5 -x LAST_RETURN \
-i Autzen_Stadium.las -o Autzen_Stadium_5m.spd
```

Once the command has completed open the file in the SPDPointsViewer and it will look similar to that shown in Figure 9.1.

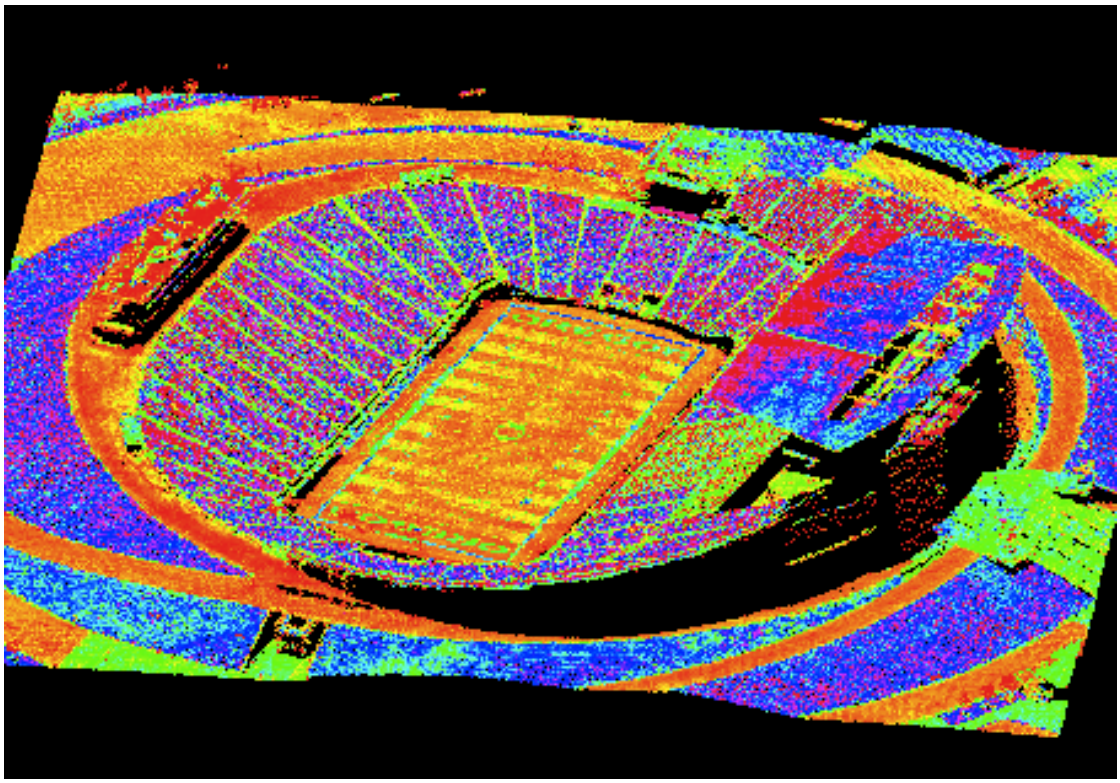


Figure 9.1: Tutorial 1 screenshot.

9.2 Interpolate DTM and DSMs

When you have a your LiDAR data as an SPD file the most command datasets to produce are Digital Terrestrial Model (DTM) and Digital Surface Model (DSM). To produce those products you need need to interpolate a raster surface from the classified ground returns and top surface points. A key parameter is the resolution of the raster which is generated, within SPDLib the resolution of the raster needs to be a whole number multiple of the SPD index, for example, if the SPD file has a bin size of 10 m then the the output raster file resolution can be 1, 2 or 5 m but not 3 m.

Run the following commands first convert the LAS file to SPD before generating raster DTM and DSM products. Hillshade models of the DTM and DSM's are then created for visualisation.

```
spdtranslate --if LAS --of SPD -b 10 -x LAST_RETURN \
-i LiDAR_GrdClassed.las -o LiDAR_GrdClassed_10m.spd

spdinterp --dtm --topo -r 50 --overlap 10 --in NATURAL_NEIGHBOR \
-f KEA -b 1 -i LiDAR_GrdClassed_10m.spd -o LiDAR_GrdClassed_1m_dtm.kea

spdinterp --dsm --topo -r 50 --overlap 10 --in NATURAL_NEIGHBOR \
-f KEA -b 1 -i LiDAR_GrdClassed_10m.spd -o LiDAR_GrdClassed_1m_dsm.kea

gdaldem hillshade -of KEA LiDAR_GrdClassed_1m_dtm.kea \
LiDAR_GrdClassed_1m_dtm_hillshade.kea

gdaldem hillshade -of KEA LiDAR_GrdClassed_1m_dsm.kea \
LiDAR_GrdClassed_1m_dsm_hillshade.kea
```

Once the commands have completed you can open the output images within TuiView (or your preferred raster image viewer) and within SPDPointsViewer alongside the point cloud (File ¿ Open DTM). Figure 9.2 shows hillshade of the outputted DTM, Figure 9.3

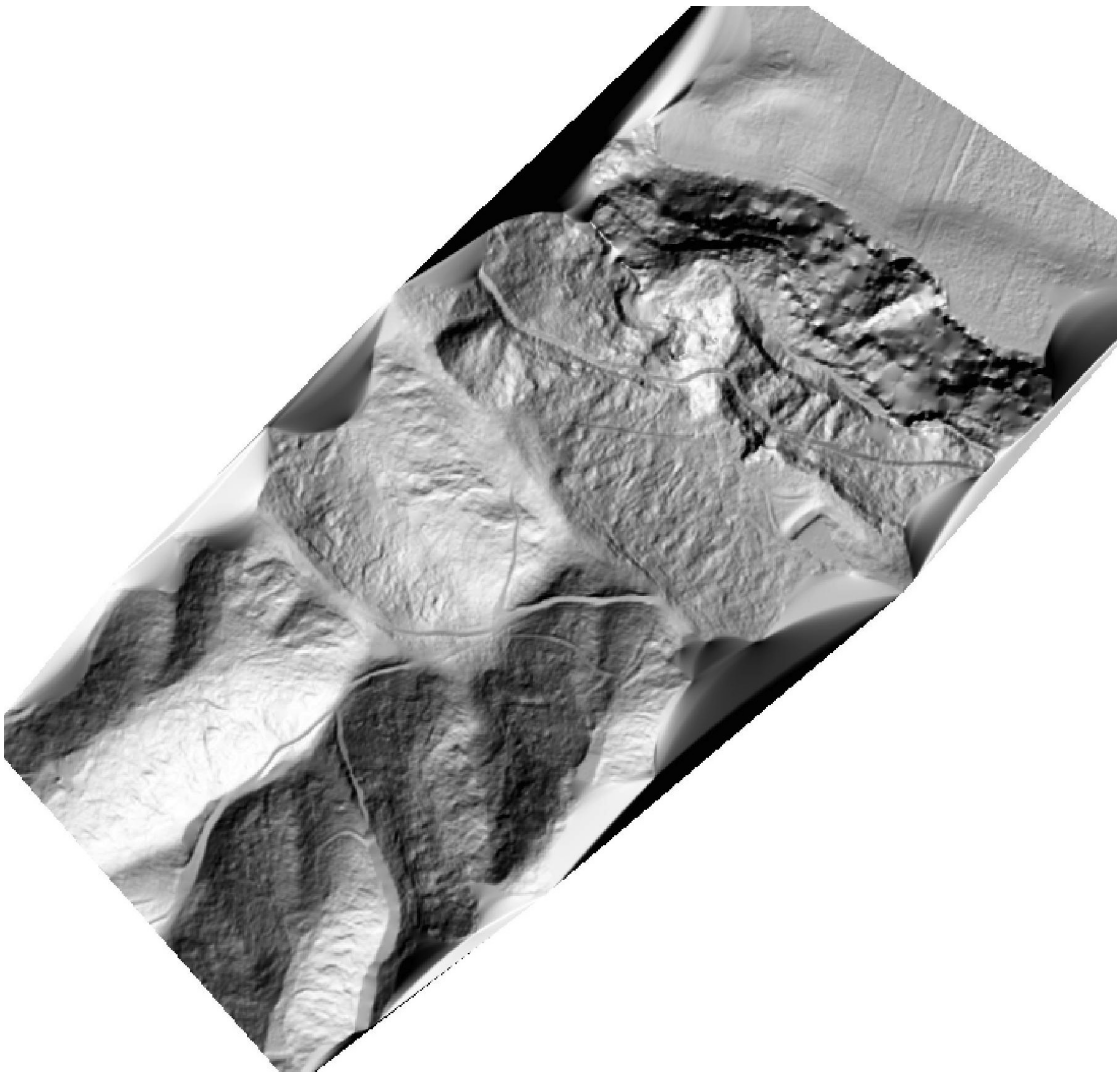


Figure 9.2: Tutorial 2 DTM screenshot: Hillshade DTM

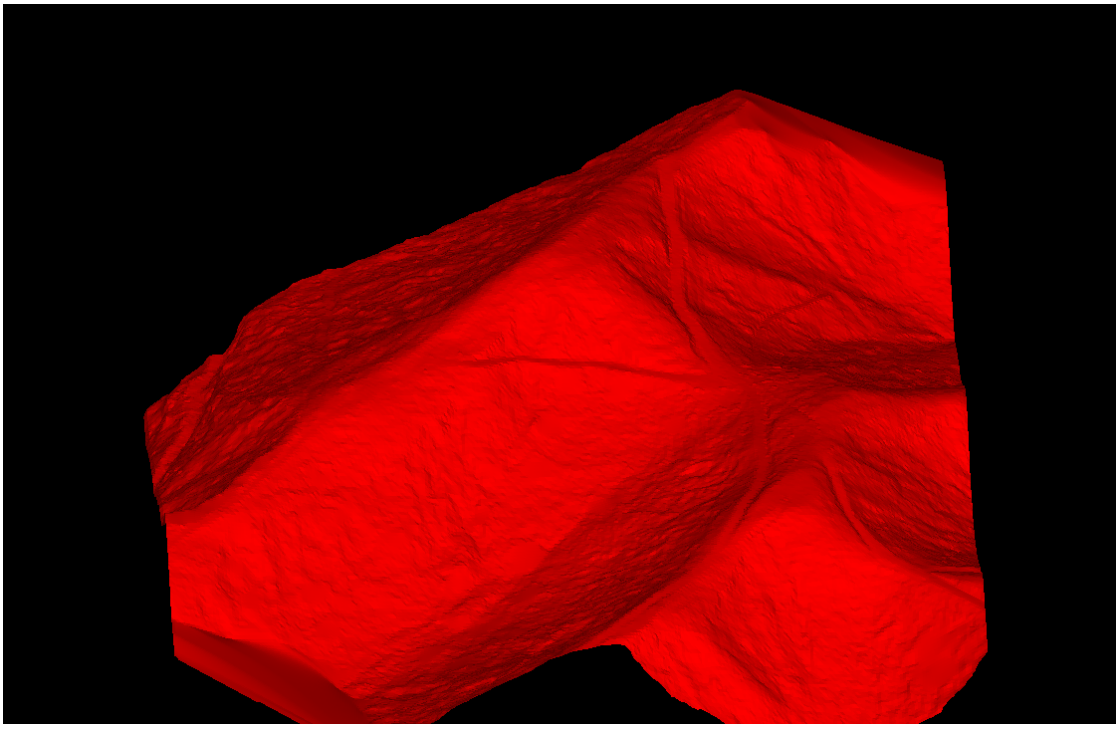


Figure 9.3: Tutorial 2 DTM screenshot: SPDPointsViewer

9.3 Classifying Ground Returns

To be able to generate a DTM you must first classify which returns are from the ground and not from features above the ground. SPDLib provides two commands for classifying the ground returns, spdpmfgrd and spdmccgrd. Read the sections associated with the command line tools for the references to those algorithms. It is recommended you experiment with both commands and their parameters. Classifying the ground returns is a key and probably the most important step when processing LiDAR data. Run the following commands to get you started.

```
spdtranslate --if LAS --of SPD -b 10 -x LAST_RETURN \
-i LiDAR.las -o LiDAR_10m.spd

spdpmfgrd -r 50 --overlap 10 --initelev 0.1 --maxfilter 14 -b 0.5 \
-i LiDAR_10m.spd -o LiDAR_10m_pmfgrd.spd

spdinterp --dtm --topo -r 50 --overlap 10 --in NATURAL_NEIGHBOR \
-f KEA -b 1 -i LiDAR_10m_pmfgrd.spd -o LiDAR_10m_pmfgrd_dtm.kea

gdaldem hillshade -of KEA LiDAR_10m_pmfgrd_dtm.kea \
LiDAR_10m_pmfgrd_dtm_hillshade.kea

spdmccgrd -r 50 --overlap 10 -i LiDAR_10m.spd -o LiDAR_10m_mccgrd.spd

spdinterp --dtm --topo -r 50 --overlap 10 --in NATURAL_NEIGHBOR \
-f KEA -b 1 -i LiDAR_10m_mccgrd.spd -o LiDAR_10m_mccgrd_dtm.kea

gdaldem hillshade -of KEA LiDAR_10m_mccgrd_dtm.kea \
LiDAR_10m_mccgrd_dtm_hillshade.kea
```

Once you have run these commands you can open the DTM's within TuiView to see the differences within the hillshade models and also open the classified SPD files within the SPDPointsViewer and select the point colouring to be by classification. In this view the points coloured red which are slightly larger than the others are the ones classified as ground returns (Figure 9.4). Note, SPDPointsViewer also has the option to only load the ground returns.

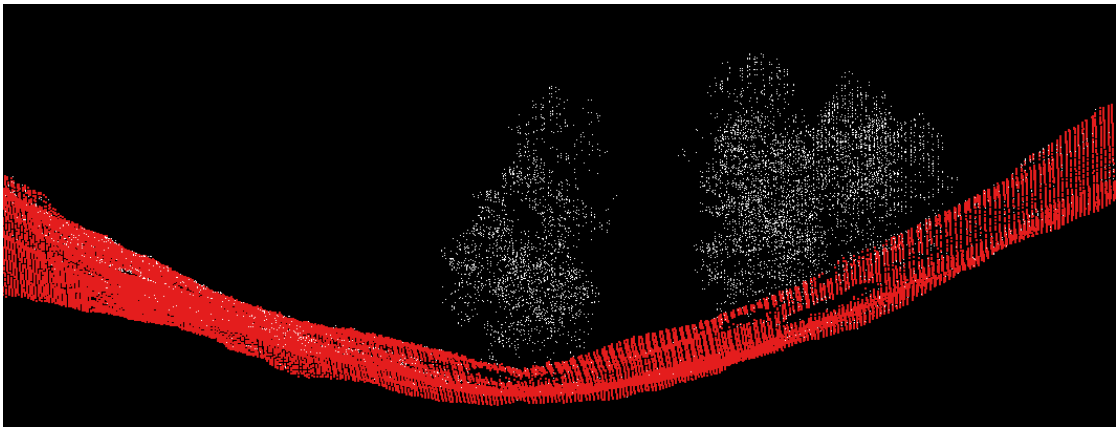


Figure 9.4: Tutorial 3 classified ground returns in SPDPointsViewer

9.4 Merging files

LiDAR datasets are often provided as a number of small tiles or flight lines, in some cases it is most convenient to merge those files into a single file before processing them. If the files are very large then it is best to either process them independently or to using the tiling command within SPDLib. Run the following commands to get an indexed SPD file, then try generating a DTM and DSM as the ground returns has already been classified.

```
spdmerge -f LAS -x FIRST_RETURN -o N144.spd N1440375.las N1440380.las N1445375.las N1445380.las

spdtranslate --if SPD --of SPD -b 10  -i N144.spd -o N144_10m.spd
```

Once this is complete open the dataset within the SPDPointsViewer. However, also try classifying the ground returns, but first remove the current classification from the points using the command below.

```
spdclearclass -i N144_10m.spd -o N144_10m_noclass.spd
```

9.5 Larger than Memory Datasets

Commonly datasets are too large to fit the into the memory (RAM) of your computer, therefore the processing needs to be aware of this and only load a small amount into memory at any one time. In SPDLib the `spdtranslate` can be provided with a temporary file path within which it will generate a series of temporary files as tiles that will then be built into a single SPD file. This requires two runs through of the input file but does allow much larger datasets to be processed. Before you run the following command don't forget to first create a `tmp` directory within the directory you are running these commands from (i.e., where you data is stored).

```
spdtranslate --if LAS --of SPD -b 10 -x LAST_RETURN --temppath tmp/ \
-r 50 -i Lincoln.las -o Lincoln_10m.spd

spdinterp -r 100 -c 100 --dsm --topo -f KEA -b 1 \
-i Lincoln_10m.spd -o Lincoln_1m_dsm.kea

gdaldem hillshade -of KEA Lincoln_1m_dsm.kea Lincoln_1m_dsm_hillshade.kea
```

If you have a region of interest for which you want to subset your dataset to just that region then this can also be done, as shown below:

```
spdsubset --shpfile StudyArea.shp -i Lincoln_10m.spd \
-o Lincoln_StudyArea_10m.spd
```

9.6 Vegetation Metrics

Within vegetation studies LiDAR is a very populate tools as it provides a direct and high resolution measurement of height and cover, the only instrument to do this. However, there are also only metrics which can be calculated and are also very useful, such as height percentiles (i.e., this height at which X % of points are beneath). SPDLib provides the `spdmetrics` commands which allows many metrics to be calculated and these are specified using an XML file, as shown below:

```
<?xml version="1.0" encoding="UTF-8" ?>
<!--
```

```
        Description:
                XML File for execution within SPDLib
                This file contains a template for the
                metrics XML interface.

        Created by Pete Bunting on Thu Mar 17 16:33:36 2011.
        Copyright (c) 2011 Pete Bunting. All rights reserved.
-->

<spdlib:metrics xmlns:spdlib="http://www.spdlib.org/xml/">
    <spdlib:metric metric="percentileheight" field="Dom_H" percentile="95"
                   return="First" class="NotGrd" lowthreshold="0.5" />
    <spdlib:metric metric="maxheight" field="Max_H" return="First"
                   class="NotGrd" lowthreshold="0.5" />
    <spdlib:metric metric="hscoi" field="hscoi" return="All"
                   class="NotGrd" vres="1"/>
    <spdlib:metric metric="canopycoverpercent" field="ccover" resolution="1"
                   radius="2" return="First" class="NotGrd" lowthreshold="0.5" />
</spdlib:metrics>
```

To generate the metrics for an input file the following commands need to be run:

```
spdtranslate --if LAS --of SPD -b 10 -x LAST_RETURN \
-i Vyrnwy_OSGB.las -o Vyrnwy_OSGB_10m.spd

spddefheight --interp -r 100 -c 100 --overlap 10 --in NATURAL_NEIGHBOR \
-i Vyrnwy_OSGB_10m.spd -o Vyrnwy_OSGB_10m_h.spd

spdmetrics --image -f KEA -r 100 -c 100 -m metrics.xml \
-i Vyrnwy_OSGB_10m_h.spd -o Vyrnwy_OSGB_1m_metrics.kea

spdinterp -r 100 -c 100 --dsm --height -f KEA -b 1 \
-i Vyrnwy_OSGB_10m_h.spd -o Vyrnwy_OSGB_1m_chm.kea

gdaldem hillshade -of KEA Vyrnwy_OSGB_1m_chm.kea Vyrnwy_OSGB_1m_chm_hs.kea
```

Note, that the `spddefheight` command is used to populate the height above the ground surface for each of the points within the file. Therefore, metrics can be calculated with reference to their height above ground rather than height as defined by the projection. Once you have calculated these metrics open them within

TuiView and look at them and consider what they are showing.

When you open the LiDAR datasets within SPDPointsViewer, please be aware that this file has the points coloured with the RGB values from an aerial photograph and therefore when selecting how the points are coloured within the viewer that the RGB option is therefore available to use (Figure 9.5)

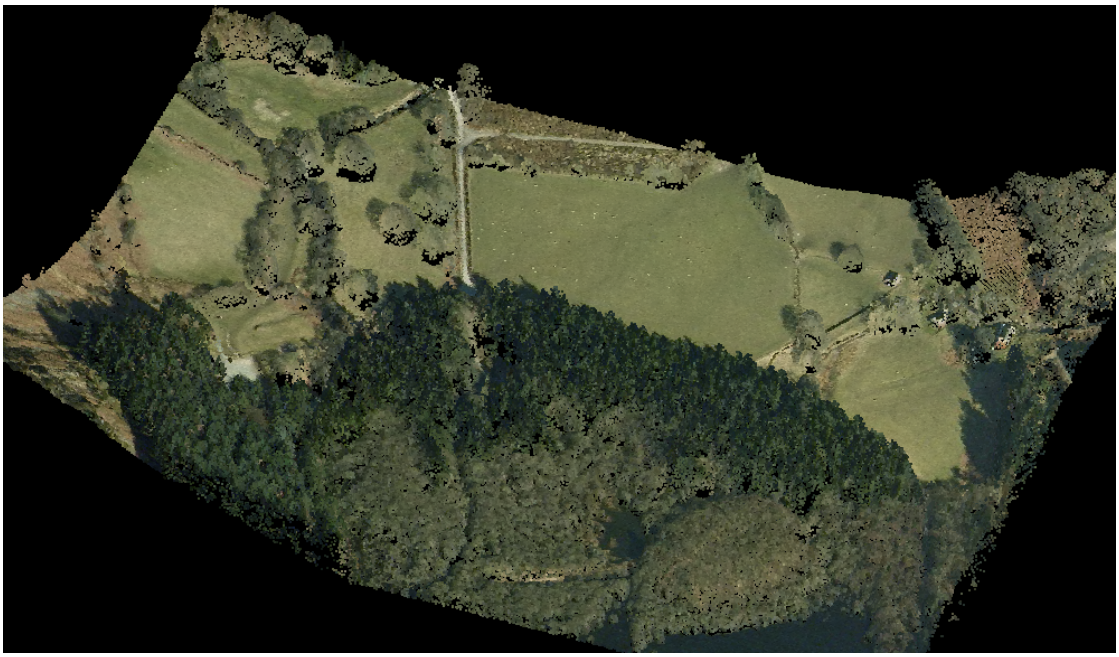


Figure 9.5: Tutorial 6 returns coloured using RGB values from aerial photo in SPDPointsViewer

Chapter 10

Tutorial 2: Combining Commands

Chapter 11

Tutorial 3: Using Processing templates

Chapter 12

Tutorial 4: Tiling Workflow

Where the primary focus is generating basic outputs primarily a DTM and maybe vegetation height over very large regions it is desirable to be able to tile datasets and run through the individual tiles simultaneously. This is particularly advantageous where a high performance computing (HPC) environment is available with a batch queue.

The following example is a workflow using SPDLib and some simple python and shell scripts on the HPC system at Auckland University, NZ, which uses the IBM loadleveler batch queue.

12.1 Step 1: Convert LAS files to SPD (Unindexed UPD)

The first step is to convert the LAS 1.2 files provided to SPD files, in this case they will initially be converted to unsorted data file (UPD; e.g., without a spatial index).

During this conversion it is recommended that you explicitly define the projection of the file by defining the –input_proj option and passing the appropriate WKT file as input. The spdproj command can be used to find the WKT string for an existing image file or SPD file.

To do this a template (convert2SPDTemplate.sh) is first created with the following content:

```
spdtranslate --if LAS --of UPD -x FIRST_RETURN  --input_proj \
./NZTM2000.wkt -i $FILEPATH -o $PATH/$FILENAME.spd
```

Using the spdbatchgen.py script the template can be used to provide a command for each input file. The output will be a single file with the command for each inputted LAS file listed.

```
spdbatchgen.py -i convert2SPDTemplate.sh -o convertLAS2SPD.sh \
-p ./Wairarapa/SPD_noIdx -d ./Wairarapa/LAS -e .las
```

The output file can be run on its own but as the HPC can run multiple jobs simultaneously we want to submit all the spdtranslate commands into the batch queue of the cluster.

To help with this a python script (Submit2Loadleveler.py) has been written which iterates through each line of an inputted text file and generates a script from each line in the input file and submits them to the cluster.

```
#! /usr/bin/env python

############################################################################
# Copyright (c) 2013 Dr. Peter Bunting, Aberystwyth University
#
# Permission is hereby granted, free of charge, to any person obtaining a copy
# of this software and associated documentation files (the "Software"), to deal
# in the Software without restriction, including without limitation the rights
# to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
# copies of the Software, and to permit persons to whom the Software is
# furnished to do so, subject to the following conditions:
# The above copyright notice and this permission notice shall be included in
# all copies or substantial portions of the Software.
#
# THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
# IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
# FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
# AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
# LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
# OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN
# THE SOFTWARE.
```

```
22 #
23 #
24 # Purpose:  A class to submit jobs to loadleveler
25 # Author: Pete Bunting
26 # Email: petebunting@mac.com
27 # Date: 19/04/2013
28 # Version: 1.0
29 #
30 # History:
31 # Version 1.0 - Created.
32 #
33 ##############################################################################
34
35 import os.path
36 import sys
37 from time import strftime
38 import argparse
39
40 class Submit2Loadleveler (object):
41
42     def startWithHash(self, line):
43         foundHash = False
44         for i in range(len(line)):
45             if line[i] == '#':
46                 foundHash = True
47         return foundHash
48
49     def createOutputFiles(self, inputFile, outputBase, memory, time, name):
50         inputFileList = open(inputFile, 'r')
51         outFileCount = 1
52         for eachLine in inputFileList:
53             if (eachLine.strip() != "") and (not self.startWithHash(eachLine)):
54                 outFile = open(outputBase + str("_") + str(outFileCount) + str(".ll"), 'w')
55
56                 outFile.write("#@ class = default\n")
57                 outFile.write("#@ group = nesi\n")
58                 outFile.write("#@ notification = never\n")
59                 outFile.write("#@ account_no = landcare\n")
60                 outFile.write("#@ wall_clock_limit = " + time + "\n");
61                 outFile.write("#@ resources = ConsumableMemory(" + memory + "mb) ConsumableVirtualMe
62                 outFile.write("#@ job_type = serial\n")
```

```
63                  outFile.write("#@ job_name = " + name + "_" + str(outFileCount) + str("\n"))
64                  outFile.write("#@ output = $(job_name).$(jobid).$(stepid).out\n")
65                  outFile.write("#@ error = $(job_name).$(jobid).$(stepid).err\n")
66                  outFile.write("#@ environment = COPY_ALL\n")
67                  outFile.write("#@ queue\n\n")
68
69                  outFile.write("ulimit -v " + str(int(memory)*1024)  + " -m " + str(int(memory)
70
71                  outFile.write(eachLine)
72
73                  outFile.write("\n\n")
74
75                  outFile.flush()
76                  outFile.close()
77
78                  command = str("llsubmit ") + outputBase + str("_") + str(outFileCount) + str("
79                  print(command)
80                  os.system(command)
81                  outFileCount+=1
82
83
84      def run(self):
85          parser = argparse.ArgumentParser()
86          parser.add_argument("-i", "--input", dest="inputFile", type=str, help="Input list of c
87          parser.add_argument("-o", "--output", dest="outputFileBase", type=str, help="Output ba
88          parser.add_argument("-m", "--memory", dest="memoryMbs", type=str, help="The amount of
89          parser.add_argument("-t", "--time", dest="timeMins", type=str, help="The time limit fo
90          parser.add_argument("-n", "--name", dest="processName", type=str, help="The name of th
91
92          args = parser.parse_args()
93
94          if args.inputFile is None:
95              print("No list of commands has been inputted.")
96              parser.print_help()
97              sys.exit()
98
99          if args.outputFileBase is None:
100             print("No output base name and path has been provided.")
101             parser.print_help()
102             sys.exit()
103
```

```
104         if args.memoryMbs is None:
105             print("The memory amount must be set.")
106             parser.print_help()
107             sys.exit()
108 
109         if args.timeMins is None:
110             print("The time (in minutes) must be set.")
111             parser.print_help()
112             sys.exit()
113 
114         if args.processName is None:
115             print("A process name must be provided.")
116             parser.print_help()
117             sys.exit()
118 
119         self.createOutputFiles(args.inputFile, args.outputFileBase, args.memoryMbs, args.timeMins, a
120 
121 if __name__ == '__main__':
122     obj = Submit2Loadleveler()
123     obj.run()
```

Using this new script the following command can be run to submit the jobs to the queue.

```
python Submit2Loadleveler.py -i convertLAS2SPD.sh -o Convert2SPDFromLAS \
-m 1000 -t 50:00 -n Convert2LAS
```

Following execution on the HPC console and error files are generated, error files have a size of zero no errors have been produced. Therefore, using a shell script (rmZeroErrOutFiles.sh) delete the error and console files where the error files have a size of zero. This makes identifying potential errors much easier.

```
1 # Created by Pete Bunting (petebunting@mac.com)
2 #
3 # A simple script to delete the error and output files
4 # of if the error file has a file of zero (i.e., there
5 #were no errors).
6 #
7 # These files a generally produced from a HPC systems.
8 #
9 
```

```
# Inputs:
# £1 is the base path

FILES=$1*.err
for f in $FILES
do
  echo "Processing $f file..."
  FILESIZE=$(stat -c%s "$f")
  echo $FILESIZE
  if [ "$FILESIZE" = 0 ] ; then
    # code
    fileBase=`basename ${f} .err`
    rm "${fileBase}.out"
    rm $f
  fi
done
```

Execute the script as follows:

```
sh ./rmZeroErrOutFiles.sh ./Convert2LAS_
```

12.2 Step 2: Define Individual Tiles

The next step is to generate a set of overlapping tiles, on which the processing will be carried out and the results merged. The overlap allows the final results to be seamlessly mosaiced after processing.

SPDLib provides two commands which need to be used to tile the data spddeftiles and spdtiling.

The first step is to create a text file which lists all the input (SPD) files which will be used as an input during these processing steps.

Again writing a python script (ListDIR2text.py) to perform this task is useful and is shown below.

```
#! /usr/bin/env python

########################################
# A script to list files with a specific
```

```
5  # extention within directory
6  #
7  # Author: Pete Bunting
8  # Email: petebunting@mac.com
9  # Date: 24/04/2013
10 # Version: 1.0
11 ######################################
12
13 import os.path
14 import sys
15 import argparse
16
17 class ListFile2Text (object):
18
19     def checkFileExtension(self, filename, extension):
20         foundExtension = False;
21         filenamesplit = os.path.splitext(filename)
22         fileExtension = filenamesplit[1].strip()
23         if(fileExtension == extension):
24             foundExtension = True
25         return foundExtension
26
27     def findFiles(self, filelist, directory, extension):
28         if os.path.exists(directory):
29             if os.path.isdir(directory):
30                 fileList = os.listdir(directory)
31                 for filename in fileList:
32                     if(os.path.isdir(os.path.join(directory,filename))):
33                         self.findFiles(filelist, os.path.join(directory,filename), extension)
34                     elif(os.path.isfile(os.path.join(directory,filename))):
35                         if(self.checkFileExtension(filename, extension)):
36                             filelist.append(os.path.join(directory,filename))
37                     else:
38                         print(filename + ' is NOT a file or directory!')
39             else:
40                 print(directory + ' is not a directory!')
41         else:
42             print(directory + ' does not exist!')
43
44     def run(self, dir, ext, output):
45         filelist = list()
```

```
46          self.findFiles(filelist, dir, ext)
47
48          outFile = open(output, 'w')
49          for file in filelist:
50              outFile.write(file)
51              outFile.write('\n')
52          outFile.close()
53
54
55  if __name__ == '__main__':
56      # Create the command line options parser.
57      parser = argparse.ArgumentParser()
58      parser.add_argument("-d", "--dir", type=str, help="The directory within which the files ar
59      parser.add_argument("-e", "--ext", type=str, help="File extention of the files of interest
60      parser.add_argument("-o", "--output", type=str, help="Output text file.")
61      # Call the parser to parse the arguments.
62      args = parser.parse_args()
63
64      if args.dir == None:
65          # Print an error message if not and exit.
66          print("Error: No dir option provided.")
67          sys.exit()
68      if args.ext == None:
69          # Print an error message if not and exit.
70          print("Error: No ext option provided.")
71          sys.exit()
72      if args.output == None:
73          # Print an error message if not and exit.
74          print("Error: No output option provided.")
75          sys.exit()
76
77
78
79      obj = ListFile2Text()
80      obj.run(args.dir, args.ext, args.output)
```

Using the scrip the following command can be executed to generate the list of files.

```
python ListDIR2text.py -d ./Wairarapa/SPD_noIdx -e .spd -o OriginalSPDFiles.lst
```

To generate a tiling the region over which the tiles are to be generated first needs

defining. The spddeftiles command has a useful tool for helping to define this by calculating the extent of all the input files, as show below.

```
spddeftiles --extent --input ./OriginalSPDFiles.lst
```

When running this command any SPD files outputted from step 1 which contain errors (i.e., cannot be opened) will also be found and provides a useful level of checking. Therefore, if any errors are produced when opening the individual files go back to step 1 and fix the cause of the error (e.g., didn't allow enough time or memory?).

Using the extent provided by the previous command use spddeftiles to define the individual tiles within a XML file. The command is shown below:

```
spddeftiles --tiles --output ./WairarapaTiles.xml --xmin 1812600 \
--xmax 1828600 --ymin 5466000 --ymax 5485600 --overlap 100 \
--ysize 1000 --xsize 1000
```

The result is an XML file with the following basic structure:

```
<tiles columns="int" overlap="double" rows="int" xmax="double"
       xmin="double" xtilesize="double" ymax="double" ymin="double"
       ytilesize="double">

    <tile col="int" corexmax="double" corexmin="double" coreymax="double"
          coreymin="double" file="string" row="int" xmax="double"
          xmin="double" ymax="double" ymin="double"/>

    <tile col="int" corexmax="double" corexmin="double" coreymax="double"
          coreymin="double" file="string" row="int" xmax="double"
          xmin="double" ymax="double" ymin="double"/>

    <tile col="int" corexmax="double" corexmin="double" coreymax="double"
          coreymin="double" file="string" row="int" xmax="double"
          xmin="double" ymax="double" ymin="double"/>
</tiles>
```

12.3 Step 4: Extract Individual Tiles

To extract the individual tiles from the whole dataset two options are available on the spdtiling command. The first (–all) is a single threaded operation which in one command extracts all the tiles but this can take quite a long time and do not use the HPC to its full potential. Therefore, the –extract option is used which just extracts a single tile but by submitting a copy of this command to the HPC batch queue then all the tiles can be generated simultaneously and therefore much more quickly.

The extract command is the following, where XX is the row and column to be extracted.

```
spdtiling --extract --deltiles --col XX --row XX \
--input ./OriginalSPDFiles.lst --tiles ./WairarapaTiles.xml \
 --output ./Wairarapa/SPDTiles/WairarapaTile
```

A python script (spdbuildtileextractcmd.py) to generate these commands has been provided as part of SPDLib and can be used as follows:

```
spdbuildtileextractcmd.py --filelist ./OriginalSPDFiles.lst \
--tiles ./WairarapaTiles.xml --outputbase ./Wairarapa/SPDTiles/WairarapaTile \
-o ./ExtractTilesCmds.txt --deltile
```

To submit these commands to the batch queue reuse the Submit2Loadleveler.py script as shown below:

```
python Submit2Loadleveler.py -i ./ExtractTilesCmds.txt \
-o ExtractSPDTile -m 2000 -t 50:00 -n ExtractTile
```

Once they have all completed check that the error files are all zero (use rmZeroErrOutFiles.sh) but also that they all completed which can be done by checking the last line of the console output which should be ‘spdtiling - end’. The tail command can be used for this, as shown below:

```
tail -n 1 ./ExtractTile_*.out
```

12.4 Step 4: Generate a Clump Image

To visual the location of the tiles and for later error checking it is also useful to create a clumps image of the tiling.

```
spdtileimg --clump --tiles ./WairarapaTiles.xml \
--output ./WairarapaTileClumps.kea --format KEA -r 10 \
--wkt NZTM2000.wkt
```

Once generated use the gdalcalcstats command to add a colour table and pyramids.

```
gdalcalcstats WairarapaTileClumps.kea -ignore 0
```

12.5 Step 5: Process Each Tile

To process each tile the spdbatchgen.py script is again used where the script will be used to generate an output script for each input file (i.e., tile). Therefore, the template needs to also include the header information for the HPC system (in this case loadleveler) and this is shown at the top of the template below.

The template is undertaking the following processing:

1. Index tile onto a 10 m grid creating an SPD file.
2. Classify the ground returns using the PMF algorithm - take a thick slice.
3. Classify the ground returns using the MCC algorithm.
4. Define the height field within the SPD file using the ground returns.
5. Interpolate a DTM from the ground returns.
6. Interpolate a CHM from using the height field.
7. Finally, tidy up tiles with are no longer required.

```
#@ class = default
#@ group = nesi
#@ notification = never
#@ account_no = [ENTER ACCOUNT NUMBER HERE]
```

```
#@ wall_clock_limit = 180:00
#@ resources = ConsumableMemory(8000mb) ConsumableVirtualMemory(8000mb)
#@ job_type = serial
#@ job_name = £FILENAME_LiDARProcess
#@ output = £(job_name).£(jobid).£(stepid).out
#@ error = £(job_name).£(jobid).£(stepid).err
#@ environment = COPY_ALL
#@ queue

ulimit -v 8192000 -m 8192000

mkdir $PATH/tmp/$FILENAME

spdtranslate --if SPD --of SPD --temppath $PATH/tmp/$FILENAME/$FILENAMETmpTile \
 --numofrows 50 -b 10 -x FIRST_RETURN -i  $FILEPATH \
-o $PATH/SPD/$FILENAME_10m.spd

spdpmfgrd -c 50 -r 50 -b 1 --grd 1 -i $PATH/SPD/$FILENAME_10m.spd \
-o $PATH/SPD/$FILENAME_10m_pmfgrd.spd

spdmccgrd -c 50 -r 50 -b 1 --class 3 --initcurvetol 1 \
-i $PATH/SPD/$FILENAME_10m_pmfgrd.spd -o $PATH/SPD/$FILENAME_10m_pmfmccgrd.spd

spddefheight --interp -c 50 -r 50 --in NATURAL_NEIGHBOR \
-i $PATH/SPD/$FILENAME_10m_pmfmccgrd.spd -o $PATH/SPD/$FILENAME_10m_pmfmccgrd_h.spd

spdinterp --dtm --topo --in NATURAL_NEIGHBOR -f KEA -b 2 -c 50 -r 50 --overlap 10 \
-i $PATH/SPD/$FILENAME_10m_pmfmccgrd_h.spd -o $PATH/DTM/$FILENAME_1m_pmfmccgrd_dtm.kea

spdinterp --dsm --height --in NATURAL_NEIGHBOR -f KEA -b 2 -c 50 -r 50 --overlap 10 \
-i $PATH/SPD/$FILENAME_10m_pmfmccgrd_h.spd -o $PATH/CHM/$FILENAME_1m_pmfmccgrd_chm.kea

rm $PATH/SPD/$FILENAME_10m_pmfgrd.spd
rm $PATH/SPD/$FILENAME_10m_pmfmccgrd.spd
rm $PATH/SPD/$FILENAME_10m.spd
rm -Rf $PATH/tmp/$FILENAME
```

To generate the scripts for the individual tiles using the template the spdbatchgen.py script is used as follows:

```
spdbatchgen.py -i WellingtonJobTemplate.sh -o WellingtonProcessing \
```

```
-d ./Wairarapa/SPDTiles -e .spd -p ./Wairarapa --output_type multiple \
--recurse no
```

The script also builds a shell script (*._runall.sh) which can be used on a single threaded machine to run all the jobs sequentially but as we are using on the HPC this file should be removed.

```
rm WellingtonProcessing_runall.sh
```

To submit the jobs to the HPC batch queue it is again convenient define a shell script for these, as shown below.

```
FILES=WellingtonProcessingWairarapaTile*.sh
for f in $FILES
do
  echo "Submitting $f to the queue..."
  llsubmit $f
done
```

These three steps can also be put together into a single shell script which simplifies the processing further so it is recommend that a script (submitTilesForProcessing.sh) is created, as shown below.

```
spdbatchgen.py -i WellingtonJobTemplate.sh -o WellingtonProcessing \
-d ./Wairarapa/SPDTiles -e .spd -p ./Wairarapa --output_type multiple \
--recurse no

rm WellingtonProcessing_runall.sh

FILES=WellingtonProcessingWairarapaTile*.sh
for f in $FILES
do
  echo "Submitting $f to the queue..."
  llsubmit $f
done
```

All the tiles can therefore be executed with the following command.

```
sh submitTilesForProcessing.sh
```

12.6 Step 6: Check Outputs are Consitant

The next step is to make sure all the tiles ran through OK, which first of all can be done with checking the error output files but also that the number and size of the output files are as expected.

Use the du command to check the directory sizes make sense (i.e., the SPD folder is a similar size to SPDTiles) and count the tiles to make sure all are present. The commands are shown below:

```
du -h
ls -l SPDTiles | wc -l
ls -l SPD | wc -l
ls -l CHM | wc -l
ls -l DTM | wc -l
```

12.7 Step 7: Mosaic Image Outputs

To mosaic the results create lists of the output image tiles (DTM and CHM folders) using the ListDIR2text.py script.

```
python ListDIR2text.py -d ./Wairarapa/CHM -e .kea -o CHMTiles.lst
python ListDIR2text.py -d ./Wairarapa/DTM -e .kea -o DTMTiles.lst
```

The spdtileimg command can then be used to mosaic the result back together.

CHM:

```
spdtileimg --mosaic --format KEA --input ./CHMTiles.lst \
--tiles ./WairarapaTiles.xml --output /./Wairarapa/Wairarapa_CHM.kea

gdalcalcstats Wairarapa_CHM.kea -ignore 0
```

DTM:

```
spdtileimg --mosaic --format KEA --input ./DTMTiles.lst \
--tiles ./WairarapaTiles.xml --output ./Wairarapa/Wairarapa_DTM.kea

gdalcalcstats Wairarapa_DTM.kea -ignore 0
```

You may also want to create a hillshade image from the DTM and this can be done using the gdaldem tool:

```
gdaldem hillshade -of KEA Wairarapa_DTM.kea Wairarapa_DTM_HS.kea

gdalcalcstats Wairarapa_DTM_HS.kea
```

Part V

Development

Bibliography

Bater, C. W., Coops, N. C., 2009. Evaluating error associated with lidar-derived DEM interpolation. Computers and Geosciences 35 (2), pp. 289–300.

Bunting, P., Armston, J., Clewley, D., Lucas, R. M., 2013a. Sorted pulse data (SPD) library. Part II: A processing framework for LiDAR data from pulsed laser systems in terrestrial environments. Computers and Geosciences 56, 207–215.

Bunting, P., Armston, J., Lucas, R. M., Clewley, D., 2013b. Sorted pulse data (SPD) library. Part I: A generic file format for LiDAR data from pulsed laser systems in terrestrial environments. Computers and Geosciences 56, 197–206.

Bunting, P., Gillingham, S., 2013. The KEA image file format. Computers and Geosciences 57, 54–58.

Evans, J. S., Hudak, A. T., 2007. A multiscale curvature algorithm for classifying discrete return lidar in forested environments. IEEE Transactions on Geoscience and Remote Sensing 45 (4), pp. 1029–1038.

The CGAL Project, 2012. CGAL User and Reference Manual, 4.1 Edition. CGAL Editorial Board, http://www.cgal.org/Manual/4.1/doc_html/cgal_manual/packages.html.

Zhang, K., Chen, S., Whitman, D., Shyu, M., Yan, J., Zhang, C., 2003. A progressive morphological filter for removing nonground measurements from airborne LIDAR data. IEEE Transactions on Geoscience and Remote Sensing 41 (4), pp. 872–882.