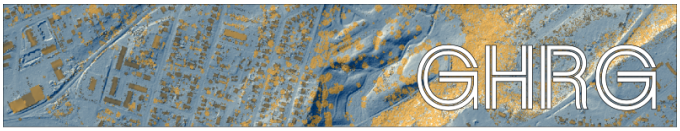
WhiteboxTools User Manual

John B. Lindsay, University of Guelph

Contents

WhiteboxTools Version 0.4
Dr. John B. Lindsay © Feb. 21, 2018
Geomorphometry and Hydrogeomatics Research Group
University of Guelph
Guelph, Canada



Geomorphometry & Hydrogeomatics Research Group

1. Introduction

WhiteboxTools is an advanced geospatial data analysis engine developed by Prof. John Lindsay (webpage; jblindsay) at the University of Guelph's Geomorphometry and Hydrogeomatics Research Group (GHRG). The project began in January 2017 and quickly evolved in terms of its analytical capabilities. *WhiteboxTools* can be used to perform common geographical information systems (GIS) analysis operations, such as cost-distance analysis, distance buffering, and raster reclassification. Remote sensing and image processing tasks include image enhancement (e.g. panchromatic sharpening, contrast adjustments), image mosaicing, numerous filtering operations, simple classification (k-means clustering), and common image transformations. *WhiteboxTools* also contains advanced tooling for spatial hydrological analysis (e.g. flow-accumulation, watershed delineation, stream network analysis, sink removal), terrain analysis (e.g. common terrain indices such as slope, curvatures, wetness index, hillshading; hypsometric analysis; multi-scale topographic position analysis), and LiDAR data processing. LiDAR point clouds can be interrogated (LidarInfo, LidarHistogram), segmented, tiled and joined, analyized for outliers, interpolated to rasters (DEMs, intensity images), and ground-points can be classified or filtered. *WhiteboxTools* is not a cartographic or spatial data visualization package; instead it is meant to serve as an analytical backend for other data visualization software, mainly GIS.

> In this manual, ***WhiteboxTools*** refers to the standalone geospatial analysis library, a collection of tools contained within a compiled binary executable command-line program and the associated Python scripts that are distributed alongside the binary file (e.g. *whitebox_tools.py* and *wb_runner.py*). ***Whitebox Geospatial Analysis Tools*** and ***Whitebox GAT*** refer to the GIS software, which includes a user-interface (front-end), point-and-click tool interfaces, and cartographic data visualization capabilities.

Although *WhiteboxTools* is intended to serve as a source of plugin tools for the *Whitebox Geospatial Analysis Tools (GAT)* open-source GIS project, the tools contained in the library are stand-alone and can run outside of the larger *Whitebox GAT* project. See *Interacting With* WhiteboxTools* From the Command Prompt* for further details. There have been a large number of requests to call *Whitebox GAT* tools and functionality from outside of the *Whitebox GAT* user-interface (e.g. from Python automation scripts). *WhiteboxTools* is intended to meet these usage requirements. The current version of *Whitebox GAT* contains many equivelent tools to those found in the *WhiteboxTools* library, although they are developed using the Java programming language. A future version of *Whitebox GAT* will replace these previous tools with the new *WhiteboxTools* backend. This transition will occur over the next several releases. Eventually most of the approximately 450 tools contained within *Whitebox GAT* will be ported to *WhiteboxTools*. In addition to separating the processing capabilities and the user-interface (and thereby reducing the reliance on Java), this migration should significantly improve processing efficiency. This is because Rust, the programming language used to develop *WhiteboxTools*, is generally faster than the equivalent Java code and because many of the *WhiteboxTools* functions are designed to process data in parallel wherever possible. In contrast, the older Java codebase included largely single-threaded applications.

In addition to *Whitebox GAT*, the *WhiteboxTools* project is related to other GHRG software projects including, the *GoSpatial* project, which has similar goals but is designed using the Go programming language instead of Rust. *WhiteboxTools* has however superseded the *GoSpatial* project, having subsumed all of its

functionality. *GoSpatial* users should now transition to *WhiteboxTools*.

2. Installation

WhiteboxTools is a stand-alone executable command-line program with no actual installation. Pre-compiled binaries can be downloaded from the *Geomorphometry and Hydrogeomatics Research Group* software web site for various supported operating systems. It is likely that *WhiteboxTools* will work on a wider variety of operating systems and architectures. If you do not find your operating system/architecture in the list of available *WhiteboxTool* binaries, then compilation from source code will be necessary. WhiteboxTools can be compiled from the source code with the following steps:

1. Install the Rust compiler; Rustup is recommended for this purpose. Further instruction can be found at this link.

2. Download the *Whitebox GAT* source code. Note: *WhiteboxTools* is currently housed as a sub-repository of the main *Whitebox GAT* repo. To download the code, click the green Clone or download button on the GitHub repository site.

3. Decompress the zipped download file.

4. Open a terminal (command prompt) window and change the working directory to the whitebox_tools sub-folder, which is contained within the decompressed downloaded Whitebox GAT folder:

```
>> cd /path/to/folder/whitebox_tools/
```

5. Finally, use the rust package manager Cargo, which will be installed alongside Rust, to compile the executable:

```
>> cargo build --release
```

Depending on your system, the compilation may take several minutes. When completed, the compiled binary executable file will be contained within the *whitebox_tools/target/release/ folder*. Type *./whitebox_tools --help* at the command prompt (after changing the directory to the containing folder) for information on how to run the executable from the terminal.

> The '>>' is shorthand used in this document to denote the command prompt and is not intended to be typed.

Be sure to follow the instructions for installing Rust carefully. In particular, if you are installing on Microsoft Windows, you must have a linker installed prior to installing the Rust compiler (*rustc*). The Rust webpage recommends either the **MS Visual C++ 2015 Build Tools** or the GNU equivalent and offers details for each installation approach. You should also consider using **RustUp** to install the Rust compiler.

3. Interacting With *WhiteboxTools* From the Command Prompt

WhiteboxTools is a command-line program and can be run either by calling it from a terminal application with appropriate commands and arguments, or, more conveniently, by calling it from a script. The following commands are recognized by the *WhiteboxTools* library:

Command	Description
--cd, --wd	Changes the working directory; used in conjunction with --run flag.
-h, --help	Prints help information.
-l, --license	Prints the whitebox-tools license.
--listtools	Lists all available tools, with tool descriptions. Keywords may also be used, --listtools slope.
-r, --run	Runs a tool; used in conjunction with --cd flag; -r="LidarInfo".
--toolbox	Prints the toolbox associated with a tool; --toolbox=Slope.
--toolhelp	Prints the help associated with a tool; --toolhelp="LidarInfo".
--toolparameters	Prints the parameters (in json form) for a specific tool; e.g. --toolparameters="FeaturePreservingDenoise".
-v	Verbose mode. Without this flag, tool outputs will not be printed.
--viewcode	Opens the source code of a tool in a web browser; --viewcode="LidarInfo".
--version	Prints the version information.

Generally, the Unix convention is that single-letter arguments (options) use a single hyphen (e.g. -h) while word-arguments (longer, more descriptive argument names) use double hyphens (e.g. --help). The same rule is used for passing arguments to tools as well. Use the *--toolhelp* argument to print information about a specific tool (e.g. --toolhelp=Clump).

> Tool names can be specified either using the snake_case or CamelCase convention (e.g. *lidar_info* or *LidarInfo*).

The following is an example of calling the *WhiteboxTools* binary executable file directly from the command prompt:

```
>>./whitebox_tools --wd='/Users/johnlindsay/Documents/data/' ^
--run=DevFromMeanElev --input='DEM clipped.dep' ^
--output='DEV raster.dep' -v
```

Notice the quotation marks (single or double) used around directories and filenames, and string tool arguments in general. Use the '-v' flag (run in verbose mode) to force the tool print output to the command prompt. Please note that the whitebox_tools executable file must have permission to be executed; on some systems, this may require setting special permissions. Also, the above example uses the forward slash character (/), the directory path separator used on unix based systems. On Windows, users should use the back slash character (\) instead. Also, it is sometimes necessary to break (^) commands across mul-

tiple lines, as above, in order to better fit with the documents format. Actual command prompts should be contained to a single line.

4. Interacting With *WhiteboxTools* Using Python Scripting

By combining the *WhiteboxTools* library with the a high-level scripting language, such as Python, users are capable of creating powerful stand-alone geospatial applications and workflow automation scripts. In fact, *WhiteboxTools* functionality can be called from many different programming languages. However, given the prevalent use of the Python language in the geospatial field, the library is distributed with several resources specifically aimed at Python scripting. This section focuses on how Python programming can be used to interact with the *WhiteboxTools* library.

> Note that all of the following material assumes the user system is configured with Python 3. The code snippets below are not guaranteed to work with older versions of the language.

Interacting with *WhiteboxTools* from Python scripts is easy. To begin, each script must start by importing the *WhiteboxTools* class, contained with the *whitebox_tools.py* script; a new `WhiteboxTools` object can then be created:

```
from whitebox_tools import WhiteboxTools

wbt = WhiteboxTools()
```

The `WhiteboxTools` class expects to find the *WhiteboxTools* executable file (*whitebox_tools.exe* on Windows and *whitebox_tools* on other platforms) within the same directory as the *whitebox_tools.py* script. If the binary file is located in a separate directory, you will need to set the executable directory as follows:

```
wbt.set_whitebox_dir('/local/path/to/whitebox/binary/')
# Or alternatively...
wbt.exe_path = '/local/path/to/whitebox/binary/'
```

Individual tools can be called using the convenience methods provided in the `WhiteboxTools` class:

```
# This line performs a 5 x 5 mean filter on 'inFile.tif':
wbt.mean_filter('/file/path/inFile.tif', '/file/path/outFile.tif', 5, 5)
```

Each tool has a cooresponding convenience method. Tools can also be called using the `run_tool()` method, specifying the tool name and a list of tool arguments. Each of the tool-specific convenience

methods collect their arguments into a properly formated list and then ultimately call the `run_tools()` method. Notice that while internally *whitebox_tools.exe* uses CamelCase (e.g. MeanFilter) to denote tool names, the Python interface of *whitebox_tools.py* uses snake_case (e.g. mean_filter), according to Python style conventions. The only exceptions are tools with names that clash with Python keywords (e.g. `And()`, `Not()`, and `Or()`).

The return value can be used to check for errors during operation:

```
if wbt.ruggedness_index('/path/DEM.flt', '/path/ruggedness.flt') != 0:
    # Non-zero returns indicate an error.
    print('ERROR running ruggedness_index')
```

If, like me, your data files tend to be burried deeply in layers of sub-directories, specifying complete file names as input parameters can be tedius. In this case, the best option is setting the working directory before calling tools:

```
from whitebox_tools import WhiteboxTools

wbt = WhiteboxTools()
wbt.work_dir = "/path/to/data/" # Sets the Whitebox working directory

# Because the working directory has been set, file arguments can be
# specified simply using file names, without paths.
wbt.d_inf_flow_accumulation("DEM.dep", "output.dep", log=True)
```

An advanced text editor, such as VS Code or Atom, can provide hints and autocompletion for available tool convenience methods and their parameters, including default values.

Sometimes, however, it can be useful to print a complete list of available tools:

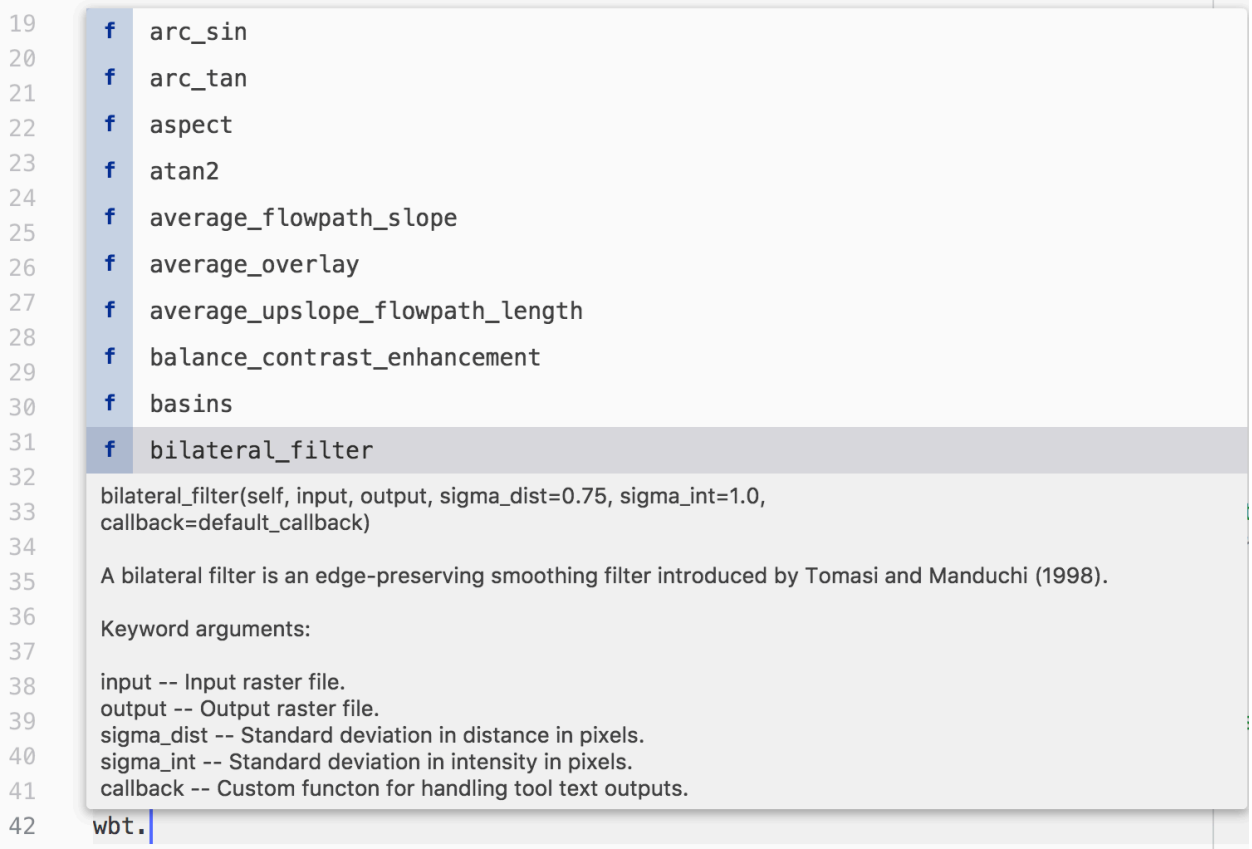
```
print(wbt.list_tools()) # List all tools in WhiteboxTools
```

The `list_tools()` method also takes an optional keywords list to search for tools:

```
# Lists tools with 'lidar' or 'LAS' in tool name or description.
print(wbt.list_tools(['lidar', 'LAS']))
```

To retrieve more detailed information for a specific tool, use the `tool_help()` method:

```
print(wbt.tool_help("elev_percentile"))
```



Autocompletion in Atom text editor makes calling *WhiteboxTools* functions easier.

`tool_help()` prints tool details including a description, tool parameters (and their flags), and example usage at the command line prompt. The above statement prints this report:

```
ElevPercentile
Description:
Calculates the elevation percentile raster from a DEM.
Toolbox: Geomorphometric Analysis
Parameters:

Flag               Description
-----------------  -----------
-i, --input, --dem Input raster DEM file.
-o, --output       Output raster file.
--filterx          Size of the filter kernel in the x-direction.
--filtery          Size of the filter kernel in the y-direction.
--sig_digits       Number of significant digits.

Example usage:
>>./whitebox_tools -r=ElevPercentile -v --wd="/path/to/data/" --dem=DEM.dep
>>-o=output.dep --filterx=25
```

Tools will frequently print text to the standard output during their execution, including warnings, progress updates and other notifications. Sometimes, when users run many tools in complex workflows and in batch mode, these output messages can be undesirable. Most tools will have their outputs suppressed by setting the *verbose* mode to *False* as follows:

```
wbt.set_verbose_mode(False)
# Or, alternatively...
wbt.verbose = False
```

Alternatively, it may be helpful to capture the text output of a tool for custom processing. This is achieved by specifying a custom *callback* function to the tool's run method:

```
# This callback function suppresses printing progress updates,
# which always use the '%' character. The callback function
# approach is flexible and allows for any level of complex
# interaction with tool outputs.
def my_callback(value):
    if not "%" in value:
        print(value)

wbt.slope('DEM.tif', 'slope_raster.tif', callback=my_callback)
```

Callback functions can also serve as a means of cancelling operations:

```
def my_callback(value):
    if user_selected_cancel_btn: # Assumes a 'Cancel' button on a GUI
        print('Cancelling operation...')
        wbt.cancel_op = True
    else:
        print(value)

wbt.breach_depressions('DEM.flt', 'DEM_breached.flt', callback=my_callback)
```

The *whitebox_tools.py* script provides several other functions for interacting with the *WhiteboxTools* library, including:

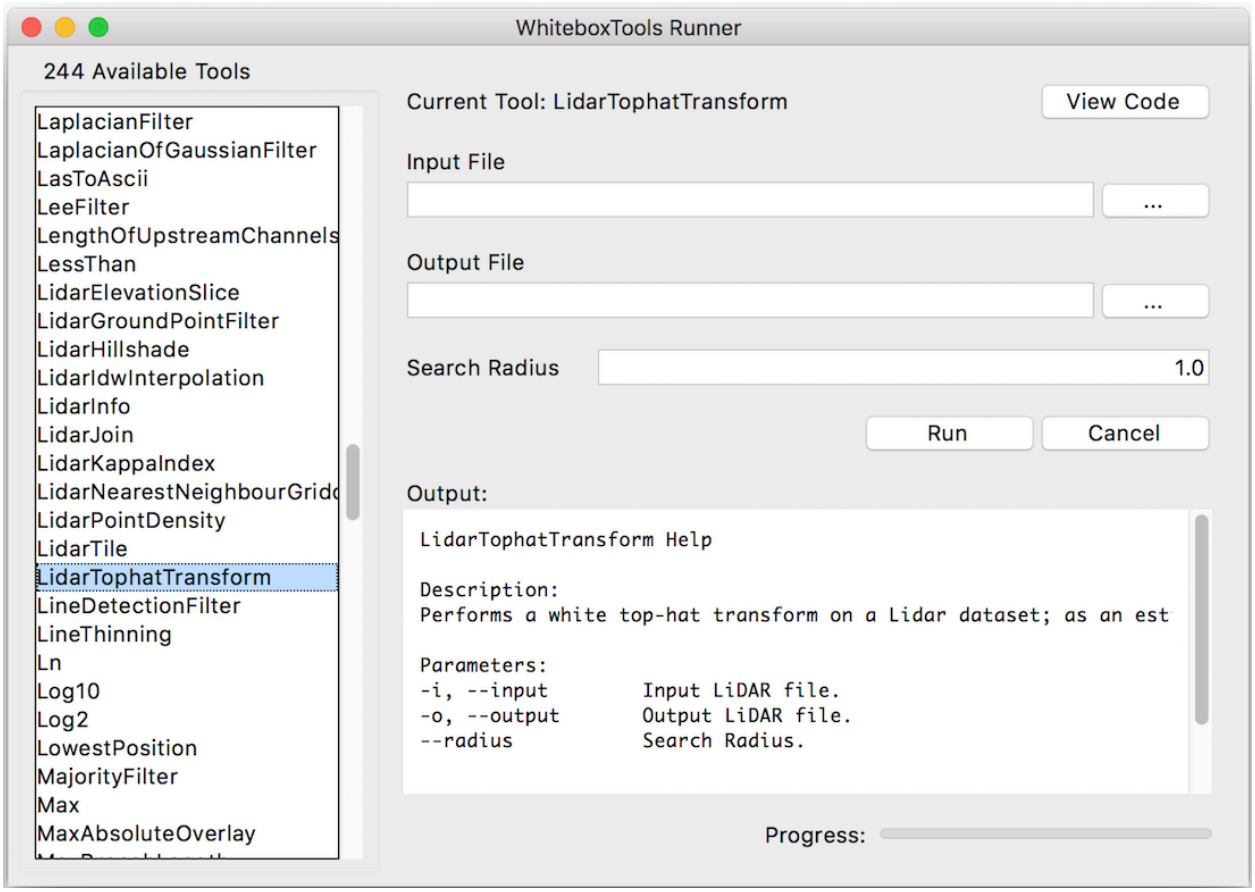
```
# Print the WhiteboxTools help...a listing of available commands
print(wbt.help())

# Print the WhiteboxTools license
print(wbt.license())

# Print the WhiteboxTools version
print("Version information: {}".format(wbt.version()))

# Get the toolbox associated with a tool
tb = wbt.toolbox('lidar_info')

# Retrieve a JSON object of a tool's parameters.
tp = tool_parameters('raster_histogram')

# Opens a browser and navigates to a tool's source code in the
# WhiteboxTools GitHub repository
view_code('watershed')
```

For a working example of how to call functions and run tools from Python, see the *whitebox_example.py* Python script, which is distributed with the *WhiteboxTools* library.

5. WhiteboxTools Runner

There is a Python script contained within the *WhiteboxTools* directory called '*wb_runner.py*'. This script is intended to provide a very basic user-interface, *WhiteboxTools Runner*, for running the tools contained

within the *WhiteboxTools* library. The user-interface uses Python's TkInter GUI library and is cross-platform. The user interface is currently experimental and is under heavy testing. Please report any issues that you experience in using it.



The *WhiteboxTools Runner* user-interface

The *WhiteboxTools Runner* does not rely on the *Whitebox GAT* user interface at all and can therefore be used indepedent of the larger project. The script must be run from a directory that also contains the '*whitebox_tools.py*' Python script and the '*whitebox_tools*' executable file. There are plans to link tool help documentation in *WhiteboxTools Runner* and to incorporate toolbox information, rather than one large listing of available tools.

6. Available Tools

Eventually most of *Whitebox GAT's* approximately 400 tools will be ported to *WhiteboxTools*, although this is an immense task. Support for vector data (Shapefile/GeoJSON) reading/writing and a topological analysis library (like the Java Topology Suite) will need to be added in order to port the tools involving vector spatial data. Opportunities to parallelize algorithms will be sought during porting. All new plugin tools will be added to *Whitebox GAT* using this library of functions.

The library currently contains the following 274 tools, which are each grouped based on their main function into one of the following categories: *Data Tools*, *Geomorphometric Analysis* (i.e. digital terrain analysis), *GIS Analysis*, *Hydrological Analysis*, *Image Analysis*, *LiDAR Analysis*, *Mathematical and Statistical Analysis*, and *Stream Network Analysis*. To retrieve detailed information about a tool's input arguments and example usage, either use the *--toolhelp* command from the terminal, or the *tool_help('tool_name')* function from the *whitebox_tools.py* script. The following is a complete listing of available tools, with brief descriptions, tool parameter, and example usage.

6.1 Data Tools

6.1.1 ConvertNodataToZero

Description: Converts nodata values in a raster to zero

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=ConvertNodataToZero -v ^
--wd="/path/to/data/" --input=in.dep -o=NewRaster.dep
```

6.1.2 ConvertRasterFormat

Description: Converts raster data from one format to another

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=ConvertRasterFormat -v ^
--wd="/path/to/data/" --input=DEM.dep -o=output.dep
```

6.1.3 NewRasterFromBase

Description: Creates a new raster using a base image

Parameters:

Flag	Description
-i, --base	Input base raster file
-o, --output	Output raster file
--value	Constant value to fill raster with; either 'nodata' or numeric value
--data_type	Output raster data type; options include 'double' (64-bit), 'float' (32-bit), and 'integer' (signed 16-bit) (default is 'float')

Example Usage:

```
>>./whitebox_tools -r=NewRasterFromBase -v ^
--wd="/path/to/data/" --base=base.dep -o=NewRaster.dep ^
--value=0.0 --data_type=integer
>>./whitebox_tools ^
-r=NewRasterFromBase -v --wd="/path/to/data/" --base=base.dep ^
-o=NewRaster.dep --value=nodata
```

6.1.4 SetNodataValue

Description: Assign a specified value in an input image to the NoData value

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file
--back_value	Background value to set to nodata

Example Usage:

```
>>./whitebox_tools -r=SetNodataValue -v --wd="/path/to/data/" ^
-i=in.dep -o=newRaster.dep --back_value=1.0
```

6.2 GIS Analysis

6.2.1 AggregateRaster

Description: Aggregates a raster to a lower resolution

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file
--agg_factor	Aggregation factor, in pixels
--type	Statistic used to fill output pixels

Example Usage:

```
>>./whitebox_tools -r=AggregateRaster -v ^
--wd="/path/to/data/" -i=input.dep -o=output.dep ^
--output_text
```

6.2.2 Centroid

Description: Calculates the centroid, or average location, of raster polygon objects

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file
--text_output	Optional text output

Example Usage:

```
>>./whitebox_tools -r=Centroid -v --wd="/path/to/data/" ^
-i=polygons.dep -o=output.dep
>>./whitebox_tools -r=Centroid ^
-v --wd="/path/to/data/" -i=polygons.dep -o=output.dep ^
--text_output
```

6.2.3 Clump

Description: Groups cells that form physically discrete areas, assigning them unique identifiers

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file
--diag	Flag indicating whether diagonal connections should be considered

Flag	Description
--zero_back	Flag indicating whether zero values should be treated as a background

Example Usage:

```
>>./whitebox_tools -r=Clump -v --wd="/path/to/data/" ^
-i=input.dep -o=output.dep --diag
```

6.2.4 CreatePlane

Description: Creates a raster image based on the equation for a simple plane

Parameters:

Flag	Description
--base	Input base raster file
-o, --output	Output raster file
--gradient	Slope gradient in degrees (-85.0 to 85.0)
--aspect	Aspect (direction) in degrees clockwise from north (0.0-360.0)
--constant	Constant value

Example Usage:

```
>>./whitebox_tools -r=CreatePlane -v --wd="/path/to/data/" ^
--base=base.dep -o=NewRaster.dep --gradient=15.0 ^
--aspect=315.0
```

6.2.5 RadiusOfGyration

Description: Calculates the distance of cells from their polygon's centroid

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file
--text_output	Optional text output

Example Usage:

```
>>./whitebox_tools -r=RadiusOfGyration -v ^
--wd="/path/to/data/" -i=polygons.dep -o=output.dep ^
```

```
--text_output
```

6.2.6 RasterCellAssignment

Description: Assign row or column number to cells

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file
-a, --assign	Which variable would you like to assign to grid cells? Options include 'column', 'row', 'x', and 'y'

Example Usage:

```
>>./whitebox_tools -r=RasterCellAssignment -v ^
--wd="/path/to/data/" -i='input.dep' -o=output.dep ^
--assign='column'
```

6.3 GIS Analysis => Distance Tools

6.3.1 BufferRaster

Description: Maps a distance-based buffer around each non-background (non-zero/non-nodata) grid cell in an input image

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file
--size	Buffer size
--gridcells	Optional flag to indicate that the 'size' threshold should be measured in grid cells instead of the default map units

Example Usage:

```
>>./whitebox_tools -r=BufferRaster -v --wd="/path/to/data/" ^
-i=DEM.dep -o=output.dep
```

6.3.2 CostAllocation

Description: Identifies the source cell to which each grid cell is connected by a least-cost pathway in a cost-distance analysis

Parameters:

Flag	Description
--source	Input source raster file
--backlink	Input backlink raster file generated by the cost-distance tool
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=CostAllocation -v --wd="/path/to/data/" ^
--source='source.dep' --backlink='backlink.dep' ^
-o='output.dep'
```

6.3.3 CostDistance

Description: Performs cost-distance accumulation on a cost surface and a group of source cells

Parameters:

Flag	Description
--source	Input source raster file
--cost	Input cost (friction) raster file
--out_accum	Output cost accumulation raster file
--out_backlink	Output backlink raster file

Example Usage:

```
>>./whitebox_tools -r=CostDistance -v --wd="/path/to/data/" ^
--source=src.dep --cost=cost.dep --out_accum=accum.dep ^
--out_backlink=backlink.dep
```

6.3.4 CostPathway

Description: Performs cost-distance pathway analysis using a series of destination grid cells

Parameters:

Flag	Description
--destination	Input destination raster file

Flag	Description
--backlink	Input backlink raster file generated by the cost-distance tool
-o, --output	Output cost pathway raster file
--zero_background	Flag indicating whether zero values should be treated as a background

Example Usage:

```
>>./whitebox_tools -r=CostPathway -v --wd="/path/to/data/" ^
--destination=dst.dep --backlink=backlink.dep ^
--output=cost_path.dep
```

6.3.5 EuclideanAllocation

Description: Assigns grid cells in the output raster the value of the nearest target cell in the input image, measured by the Shih and Wu (2004) Euclidean distance transform

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=EuclideanAllocation -v ^
--wd="/path/to/data/" -i=DEM.dep -o=output.dep
```

6.3.6 EuclideanDistance

Description: Calculates the Shih and Wu (2004) Euclidean distance transform

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=EuclideanDistance -v ^
--wd="/path/to/data/" -i=DEM.dep -o=output.dep
```

6.4 GIS Analysis => Overlay Tools

6.4.1 AverageOverlay

Description: Calculates the average for each grid cell from a group of raster images

Parameters:

Flag	Description
-i, --inputs	Input raster files
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=AverageOverlay -v --wd='/path/to/data/' ^
-i='image1.dep;image2.dep;image3.dep' -o=output.dep
```

6.4.2 HighestPosition

Description: Identifies the stack position of the maximum value within a raster stack on a cell-by-cell basis

Parameters:

Flag	Description
-i, --inputs	Input raster files
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=HighestPosition -v ^
--wd='/path/to/data/' -i='image1.dep;image2.dep;image3.dep' ^
-o=output.dep
```

6.4.3 LowestPosition

Description: Identifies the stack position of the minimum value within a raster stack on a cell-by-cell basis

Parameters:

Flag	Description
-i, --inputs	Input raster files
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=LowestPosition -v --wd='/path/to/data/' ^
-i='image1.dep;image2.dep;image3.dep' -o=output.dep
```

6.4.4 MaxAbsoluteOverlay

Description: Evaluates the maximum absolute value for each grid cell from a stack of input rasters

Parameters:

Flag	Description
-i, --inputs	Input raster files
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=MaxAbsoluteOverlay -v ^
--wd='/path/to/data/' -i='image1.dep;image2.dep;image3.dep' ^
-o=output.dep
```

6.4.5 MaxOverlay

Description: Evaluates the maximum value for each grid cell from a stack of input rasters

Parameters:

Flag	Description
-i, --inputs	Input raster files
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=MaxOverlay -v --wd='/path/to/data/' ^
-i='image1.dep;image2.dep;image3.dep' -o=output.dep
```

6.4.6 MinAbsoluteOverlay

Description: Evaluates the minimum absolute value for each grid cell from a stack of input rasters

Parameters:

Flag	Description
-i, --inputs	Input raster files

Flag	Description
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=MinAbsoluteOverlay -v ^
--wd='/path/to/data/' -i='image1.dep;image2.dep;image3.dep' ^
-o=output.dep
```

6.4.7 MinOverlay

Description: Evaluates the minimum value for each grid cell from a stack of input rasters

Parameters:

Flag	Description
-i, --inputs	Input raster files
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=MinOverlay -v --wd='/path/to/data/' ^
-i='image1.dep;image2.dep;image3.dep' -o=output.dep
```

6.4.8 PercentEqualTo

Description: Calculates the percentage of a raster stack that have cell values equal to an input on a cell-by-cell basis

Parameters:

Flag	Description
-i, --inputs	Input raster files
--comparison	Input comparison raster file
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=PercentEqualTo -v --wd='/path/to/data/' ^
-i='image1.dep;image2.dep;image3.dep' --comparison='comp.dep' ^
-o='output.dep'
```

6.4.9 PercentGreaterThan

Description: Calculates the percentage of a raster stack that have cell values greather than an input on a cell-by-cell basis

Parameters:

Flag	Description
-i, --inputs	Input raster files
--comparison	Input comparison raster file
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=PercentGreaterThan -v ^
--wd='/path/to/data/' -i='image1.dep;image2.dep;image3.dep' ^
--comparison='comp.dep' -o='output.dep'
```

6.4.10 PercentLessThan

Description: Calculates the percentage of a raster stack that have cell values less than an input on a cell-by-cell basis

Parameters:

Flag	Description
-i, --inputs	Input raster files
--comparison	Input comparison raster file
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=PercentLessThan -v ^
--wd='/path/to/data/' -i='image1.dep;image2.dep;image3.dep' ^
--comparison='comp.dep' -o='output.dep'
```

6.4.11 PickFromList

Description: Outputs the value from a raster stack specified by a position raster

Parameters:

Flag	Description
-i, --inputs	Input raster files

Flag	Description
--pos_input	Input position raster file
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=PickFromList -v --wd='/path/to/data/' ^
--pos_input=position.dep -i='image1.dep;image2.dep;image3.dep' ^
-o=output.dep
```

6.4.12 WeightedSum

Description: Performs a weighted-sum overlay on multiple input raster images

Parameters:

Flag	Description
-i, --inputs	Input raster files
-o, --output	Output raster file
-w, --weights	Weight values, contained in quotes and separated by commas or semicolons

Example Usage:

```
>>./whitebox_tools -r=WeightedSum -v --wd='/path/to/data/' ^
-i='image1.dep;image2.dep;image3.dep' -o=output.dep ^
--weights='0.3;0.2;0.5'
```

6.5 GIS Analysis => Patch Shape Tools

6.5.1 EdgeProportion

Description: Calculate the proportion of cells in a raster polygon that are edge cells

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file
--output_text	flag indicating whether a text report should also be output

Example Usage:

```
>>./whitebox_tools -r=EdgeProportion -v --wd="/path/to/data/" ^
-i=input.dep -o=output.dep --output_text
```

6.5.2 FindPatchOrClassEdgeCells

Description: Finds all cells located on the edge of patch or class features

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=FindPatchOrClassEdgeCells -v ^
--wd="/path/to/data/" -i=input.dep -o=output.dep
```

6.6 GIS Analysis => Reclass Tools

6.6.1 Reclass

Description: Reclassifies the values in a raster image

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file
--reclass_vals	Reclassification triplet values (new value; from value; to less than), e.g. '0.0;0.0;1.0;1.0;1.0;2.0'
--assign_mode	Optional Boolean flag indicating whether to operate in assign mode, reclass_vals values are interpreted as new value; old value pairs

Example Usage:

```
>>./whitebox_tools -r=Reclass -v --wd="/path/to/data/" ^
-i='input.dep' -o=output.dep ^
--reclass_vals='0.0;0.0;1.0;1.0;1.0;2.0'
>>./whitebox_tools ^
-r=Reclass -v --wd="/path/to/data/" -i='input.dep' ^
-o=output.dep --reclass_vals='10;1;20;2;30;3;40;4' ^
--assign_mode
```

6.6.2 ReclassEqualInterval

Description: Reclassifies the values in a raster image based on equal-ranges

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file
--interval	Class interval size
--start_val	Optional starting value (default is input minimum value)
--end_val	Optional ending value (default is input maximum value)

Example Usage:

```
>>./whitebox_tools -r=ReclassEqualInterval -v ^
--wd="/path/to/data/" -i='input.dep' -o=output.dep ^
--interval=10.0 --start_val=0.0
```

6.6.3 ReclassFromFile

Description: Reclassifies the values in a raster image using reclass ranges in a text file

Parameters:

Flag	Description
-i, --input	Input raster file
--reclass_file	Input text file containing reclass ranges
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=ReclassFromFile -v ^
--wd="/path/to/data/" -i='input.dep' ^
--reclass_file='reclass.txt' -o=output.dep
```

6.7 Geomorphometric Analysis

6.7.1 Aspect

Description: Calculates an aspect raster from an input DEM

Parameters:

Flag	Description
-i, --dem	Input raster DEM file
-o, --output	Output raster file
--zfactor	Optional multiplier for when the vertical and horizontal units are not the same

Example Usage:

```
>>./whitebox_tools -r=Aspect -v --wd="/path/to/data/" ^
--dem=DEM.dep -o=output.dep
```

6.7.2 DevFromMeanElev

Description: Calculates deviation from mean elevation

Parameters:

Flag	Description
-i, --input, --dem	Input raster DEM file
-o, --output	Output raster file
--filterx	Size of the filter kernel in the x-direction
--filtery	Size of the filter kernel in the y-direction

Example Usage:

```
>>./whitebox_tools -r=DevFromMeanElev -v ^
--wd="/path/to/data/" --dem=DEM.dep -o=output.dep ^
--filter=25
```

6.7.3 DiffFromMeanElev

Description: Calculates difference from mean elevation (equivalent to a high-pass filter)

Parameters:

Flag	Description
-i, --input, --dem	Input raster DEM file
-o, --output	Output raster file
--filterx	Size of the filter kernel in the x-direction
--filtery	Size of the filter kernel in the y-direction

Example Usage:

```
>>./whitebox_tools -r=DiffFromMeanElev -v ^
--wd="/path/to/data/" --dem=DEM.dep -o=output.dep ^
--filter=25
```

6.7.4 DirectionalRelief

Description: Calculates relief for cells in an input DEM for a specified direction

Parameters:

Flag	Description
-i, --dem	Input raster DEM file
-o, --output	Output raster file
--azimuth	Wind azimuth in degrees
--max_dist	Optional maximum search distance (unspecified if none; in xy units)

Example Usage:

```
>>./whitebox_tools -r=DirectionalRelief -v ^
--wd="/path/to/data/" -i='input.dep' -o=output.dep ^
--azimuth=315.0
```

6.7.5 DownslopeIndex

Description: Calculates the Hjerdt et al. (2004) downslope index

Parameters:

Flag	Description
-i, --dem	Input raster DEM file
-o, --output	Output raster file
--drop	Vertical drop value (default is 2.0)
--out_type	Output type, options include 'tangent', 'degrees', 'radians', 'distance' (default is 'tangent')

Example Usage:

```
>>./whitebox_tools -r=DownslopeIndex -v --wd="/path/to/data/" ^
--dem=pointer.dep -o=dsi.dep --drop=5.0 --out_type=distance
```

6.7.6 ElevAbovePit

Description: Calculate the elevation of each grid cell above the nearest downstream pit cell or grid edge cell

Parameters:

Flag	Description
-i, --dem	Input raster DEM file
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=ElevAbovePit -v --wd="/path/to/data/" ^
--dem=DEM.dep -o=output.dep
```

6.7.7 ElevPercentile

Description: Calculates the elevation percentile raster from a DEM

Parameters:

Flag	Description
-i, --input, --dem	Input raster DEM file
-o, --output	Output raster file
--filterx	Size of the filter kernel in the x-direction
--filtery	Size of the filter kernel in the y-direction
--sig_digits	Number of significant digits

Example Usage:

```
>>./whitebox_tools -r=ElevPercentile -v --wd="/path/to/data/" ^
--dem=DEM.dep -o=output.dep --filter=25
```

6.7.8 ElevRelativeToMinMax

Description: Calculates the elevation of a location relative to the minimum and maximum elevations in a DEM

Parameters:

Flag	Description
-i, --dem	Input raster DEM file

Flag	Description
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=ElevRelativeToMinMax -v ^
--wd="/path/to/data/" --dem=DEM.dep -o=output.dep
```

6.7.9 ElevRelativeToWatershedMinMax

Description: Calculates the elevation of a location relative to the minimum and maximum elevations in a watershed

Parameters:

Flag	Description
-i, --dem	Input raster DEM file
--watersheds	Input raster watersheds file
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=ElevRelativeToWatershedMinMax -v ^
--wd="/path/to/data/" --dem=DEM.dep --watersheds=watershed.dep ^
-o=output.dep
```

6.7.10 FeaturePreservingDenoise

Description: Reduces short-scale variation in an input DEM using a modified Sun et al. (2007) algorithm

Parameters:

Flag	Description
-i, --dem	Input raster DEM file
-o, --output	Output raster file
--filter	Size of the filter kernel
--norm_diff	Maximum difference in normal vectors, in degrees
--num_iter	Number of iterations
--zfactor	Optional multiplier for when the vertical and horizontal units are not the same

Example Usage:

```
>>./whitebox_tools -r=FeaturePreservingDenoise -v ^
--wd="/path/to/data/" --dem=DEM.dep -o=output.dep
```

6.7.11 FetchAnalysis

Description: Performs an analysis of fetch or upwind distance to an obstacle

Parameters:

Flag	Description
-i, --dem	Input raster DEM file
-o, --output	Output raster file
--azimuth	Wind azimuth in degrees in degrees
--hgt_inc	Height increment value

Example Usage:

```
>>./whitebox_tools -r=FetchAnalysis -v --wd="/path/to/data/" ^
-i='input.dep' -o=output.dep --azimuth=315.0
```

6.7.12 FillMissingData

Description: Fills nodata holes in a DEM

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file
--filter	Filter size (cells)

Example Usage:

```
>>./whitebox_tools -r=FillMissingData -v ^
--wd="/path/to/data/" -i=DEM.dep -o=output.dep --filter=25
```

6.7.13 FindRidges

Description: Identifies potential ridge and peak grid cells

Parameters:

Flag	Description
-i, --dem	Input raster DEM file
-o, --output	Output raster file
--line_thin	Optional flag indicating whether post-processing line-thinning should be performed

Example Usage:

```
>>./whitebox_tools -r=FindRidges -v --wd="/path/to/data/" ^
--dem=pointer.dep -o=out.dep --line_thin
```

6.7.14 Hillshade

Description: Calculates a hillshade raster from an input DEM

Parameters:

Flag	Description
-i, --dem	Input raster DEM file
-o, --output	Output raster file
--azimuth	Illumination source azimuth in degrees
--altitude	Illumination source altitude in degrees
--zfactor	Optional multiplier for when the vertical and horizontal units are not the same

Example Usage:

```
>>./whitebox_tools -r=Hillshade -v --wd="/path/to/data/" ^
-i=DEM.dep -o=output.dep --azimuth=315.0 --altitude=30.0
```

6.7.15 HorizonAngle

Description: Calculates horizon angle (maximum upwind slope) for each grid cell in an input DEM

Parameters:

Flag	Description
-i, --dem	Input raster DEM file
-o, --output	Output raster file
--azimuth	Wind azimuth in degrees
--max_dist	Optional maximum search distance (unspecified if none; in xy units)

Example Usage:

```
>>./whitebox_tools -r=HorizonAngle -v --wd="/path/to/data/" ^
-i='input.dep' -o=output.dep --azimuth=315.0
```

6.7.16 HypsometricAnalysis

Description: Calculates a hypsometric curve for one or more DEMs

Parameters:

Flag	Description
-i, --inputs	Input DEM files
--watershed	Input watershed files (optional)
-o, --output	Output HTML file (default name will be based on input file if unspecified)

Example Usage:

```
>>./whitebox_tools -r=HypsometricAnalysis -v ^
--wd="/path/to/data/" -i="DEM1.tif;DEM2.tif" ^
--watershed="ws1.tif;ws2.tif" -o=outfile.html
```

6.7.17 MaxAnisotropyDev

Description: Calculates the maximum anisotropy (directionality) in elevation deviation over a range of spatial scales

Parameters:

Flag	Description
-i, --dem	Input raster DEM file
--out_mag	Output raster DEVmax magnitude file
--out_scale	Output raster DEVmax scale file
--min_scale	Minimum search neighbourhood radius in grid cells
--max_scale	Maximum search neighbourhood radius in grid cells
--step	Step size as any positive non-zero integer

Example Usage:

```
>>./whitebox_tools -r=MaxAnisotropyDev -v ^
--wd="/path/to/data/" --dem=DEM.dep -out_mag=DEVmax_mag.dep ^
--out_scale=DEVmax_scale.dep --min_scale=1 --max_scale=1000 ^
--step=5
```

6.7.18 MaxBranchLength

Description: Lindsay and Seibert's (2013) branch length index is used to map drainage divides or ridge lines

Parameters:

Flag	Description
-i, --dem	Input raster DEM file
-o, --output	Output raster file
--log	Optional flag to request the output be log-transformed

Example Usage:

```
>>./whitebox_tools -r=MaxBranchLength -v ^
--wd="/path/to/data/" --dem=DEM.dep -o=output.dep
```

6.7.19 MaxDownslopeElevChange

Description: Calculates the maximum downslope change in elevation between a grid cell and its eight downslope neighbors

Parameters:

Flag	Description
-i, --dem	Input raster DEM file
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=MaxDownslopeElevChange -v ^
--wd="/path/to/data/" --dem=DEM.dep -o=out.dep
```

6.7.20 MaxElevationDeviation

Description: Calculates the maximum elevation deviation over a range of spatial scales

Parameters:

Flag	Description
-i, --dem	Input raster DEM file
--out_mag	Output raster DEVmax magnitude file
--out_scale	Output raster DEVmax scale file
--min_scale	Minimum search neighbourhood radius in grid cells
--max_scale	Maximum search neighbourhood radius in grid cells

Flag	Description
--step	Step size as any positive non-zero integer

Example Usage:

```
>>./whitebox_tools -r=MaxElevationDeviation -v ^
--wd="/path/to/data/" --dem=DEM.dep -out_mag=DEVmax_mag.dep ^
--out_scale=DEVmax_scale.dep --min_scale=1 --max_scale=1000 ^
--step=5
```

6.7.21 MinDownslopeElevChange

Description: Calculates the minimum downslope change in elevation between a grid cell and its eight downslope neighbors

Parameters:

Flag	Description
-i, --dem	Input raster DEM file
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=MinDownslopeElevChange -v ^
--wd="/path/to/data/" --dem=DEM.dep -o=out.dep
```

6.7.22 MultiscaleTopographicPositionImage

Description: Creates a multiscale topographic position image from three DEVmax rasters of differing spatial scale ranges

Parameters:

Flag	Description
--local	Input local-scale topographic position (DEVmax) raster file
--meso	Input meso-scale topographic position (DEVmax) raster file
--broad	Input broad-scale topographic position (DEVmax) raster file
-o, --output	Output raster file
--lightness	Image lightness value (default is 1.2)

Example Usage:

```
>>./whitebox_tools -r=MultiscaleTopographicPositionImage -v ^
--wd="/path/to/data/" --local=DEV_local.dep --meso=DEV_meso.dep ^
--broad=DEV_broad.dep -o=output.dep --lightness=1.5
```

6.7.23 NumDownslopeNeighbours

Description: Calculates the number of downslope neighbours to each grid cell in a DEM

Parameters:

Flag	Description
-i, --dem	Input raster DEM file
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=NumDownslopeNeighbours -v ^
--wd="/path/to/data/" -i=DEM.dep -o=output.dep
```

6.7.24 NumUpslopeNeighbours

Description: Calculates the number of upslope neighbours to each grid cell in a DEM

Parameters:

Flag	Description
-i, --dem	Input raster DEM file
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=NumUpslopeNeighbours -v ^
--wd="/path/to/data/" -i=DEM.dep -o=output.dep
```

6.7.25 PennockLandformClass

Description: Classifies hillslope zones based on slope, profile curvature, and plan curvature

Parameters:

Flag	Description
-i, --dem	Input raster DEM file
-o, --output	Output raster file

Flag	Description
--slope	Slope threshold value, in degrees (default is 3.0)
--prof	Profile curvature threshold value (default is 0.1)
--plan	Plan curvature threshold value (default is 0.0)
--zfactor	Optional multiplier for when the vertical and horizontal units are not the same

Example Usage:

```
>>./whitebox_tools -r=PennockLandformClass -v ^
--wd="/path/to/data/" --dem=DEM.dep -o=output.dep --slope=3.0 ^
--prof=0.1 --plan=0.0
```

6.7.26 PercentElevRange

Description: Calculates percent of elevation range from a DEM

Parameters:

Flag	Description
-i, --input, --dem	Input raster DEM file
-o, --output	Output raster file
--filterx	Size of the filter kernel in the x-direction
--filtery	Size of the filter kernel in the y-direction

Example Usage:

```
>>./whitebox_tools -r=PercentElevRange -v ^
--wd="/path/to/data/" -i=DEM.dep -o=output.dep --filter=25
```

6.7.27 PlanCurvature

Description: Calculates a plan (contour) curvature raster from an input DEM

Parameters:

Flag	Description
-i, --dem	Input raster DEM file
-o, --output	Output raster file
--zfactor	Optional multiplier for when the vertical and horizontal units are not the same

Example Usage:

```
>>./whitebox_tools -r=PlanCurvature -v --wd="/path/to/data/" ^
--dem=DEM.dep -o=output.dep
```

6.7.28 Profile

Description: Plots profiles from digital elevation models

Parameters:

Flag	Description
--lines	Input vector points file
--surface	Input raster surface file
-o, --output	Output HTML file

Example Usage:

```
>>./whitebox_tools -r=Profile -v --wd="/path/to/data/" ^
--lines=profile.shp --surface=dem.dep -o=profile.html
```

6.7.29 ProfileCurvature

Description: Calculates a profile curvature raster from an input DEM

Parameters:

Flag	Description
-i, --dem	Input raster DEM file
-o, --output	Output raster file
--zfactor	Optional multiplier for when the vertical and horizontal units are not the same

Example Usage:

```
>>./whitebox_tools -r=ProfileCurvature -v ^
--wd="/path/to/data/" --dem=DEM.dep -o=output.dep
```

6.7.30 RelativeAspect

Description: Calculates relative aspect (relative to a user-specified direction) from an input DEM

Parameters:

Flag	Description
-i, --dem	Input raster DEM file

Flag	Description
-o, --output	Output raster file
--azimuth	Illumination source azimuth
--zfactor	Optional multiplier for when the vertical and horizontal units are not the same

Example Usage:

```
>>./whitebox_tools -r=RelativeAspect -v --wd="/path/to/data/" ^
--dem=DEM.dep -o=output.dep --azimuth=180.0
```

6.7.31 RelativeStreamPowerIndex

Description: Calculates the relative stream power index

Parameters:

Flag	Description
--sca	Input raster specific contributing area (SCA) file
--slope	Input raster slope file
-o, --output	Output raster file
--exponent	SCA exponent value

Example Usage:

```
>>./whitebox_tools -r=RelativeStreamPowerIndex -v ^
--wd="/path/to/data/" --sca='flow_accum.dep' ^
--slope='slope.dep' -o=output.dep --exponent=1.1
```

6.7.32 RelativeTopographicPosition

Description: Calculates the relative topographic position index from a DEM

Parameters:

Flag	Description
-i, --dem	Input raster DEM file
-o, --output	Output raster file
--filterx	Size of the filter kernel in the x-direction
--filtery	Size of the filter kernel in the y-direction

Example Usage:

```
>>./whitebox_tools -r=RelativeTopographicPosition -v ^
--wd="/path/to/data/" --dem=DEM.dep -o=output.dep ^
--filter=25
```

6.7.33 RemoveOffTerrainObjects

Description: Removes off-terrain objects from a raster digital elevation model (DEM)

Parameters:

Flag	Description
-i, --input, --dem	Input raster DEM file
-o, --output	Output raster file
--filter	Filter size (cells)
--slope	Slope threshold value

Example Usage:

```
>>./whitebox_tools -r=RemoveOffTerrainObjects -v ^
--wd="/path/to/data/" --dem=DEM.dep -o=bare_earth_DEM.dep ^
--filter=25 --slope=10.0
```

6.7.34 RuggednessIndex

Description: Calculates the Riley et al.'s (1999) terrain ruggedness index from an input DEM

Parameters:

Flag	Description
-i, --dem	Input raster DEM file
-o, --output	Output raster file
--zfactor	Optional multiplier for when the vertical and horizontal units are not the same

Example Usage:

```
>>./whitebox_tools -r=RuggednessIndex -v ^
--wd="/path/to/data/" --dem=DEM.dep -o=output.dep
```

6.7.35 SedimentTransportIndex

Description: Calculates the sediment transport index

Parameters:

Flag	Description
--sca	Input raster specific contributing area (SCA) file
--slope	Input raster slope file
-o, --output	Output raster file
--sca_exponent	SCA exponent value
--slope_exponent	Slope exponent value

Example Usage:

```
>>./whitebox_tools -r=SedimentTransportIndex -v ^
--wd="/path/to/data/" --sca='flow_accum.dep' ^
--slope='slope.dep' -o=output.dep --sca_exponent=0.5 ^
--slope_exponent=1.0
```

6.7.36 Slope

Description: Calculates a slope raster from an input DEM

Parameters:

Flag	Description
-i, --dem	Input raster DEM file
-o, --output	Output raster file
--zfactor	Optional multiplier for when the vertical and horizontal units are not the same

Example Usage:

```
>>./whitebox_tools -r=Slope -v --wd="/path/to/data/" ^
--dem=DEM.dep -o=output.dep
```

6.7.37 SlopeVsElevationPlot

Description: Creates a slope vs. elevation plot for one or more DEMs

Parameters:

Flag	Description
-i, --inputs	Input DEM files
--watershed	Input watershed files (optional)
-o, --output	Output HTML file (default name will be based on input file if unspecified)

Example Usage:

```
>>./whitebox_tools -r=SlopeVsElevationPlot -v ^
--wd="/path/to/data/" -i="DEM1.tif;DEM2.tif" ^
--watershed="ws1.tif;ws2.tif" -o=outfile.html
```

6.7.38 TangentialCurvature

Description: Calculates a tangential curvature raster from an input DEM

Parameters:

Flag	Description
-i, --dem	Input raster DEM file
-o, --output	Output raster file
--zfactor	Optional multiplier for when the vertical and horizontal units are not the same

Example Usage:

```
>>./whitebox_tools -r=TangentialCurvature -v ^
--wd="/path/to/data/" --dem=DEM.dep -o=output.dep
```

6.7.39 TotalCurvature

Description: Calculates a total curvature raster from an input DEM

Parameters:

Flag	Description
-i, --dem	Input raster DEM file
-o, --output	Output raster file
--zfactor	Optional multiplier for when the vertical and horizontal units are not the same

Example Usage:

```
>>./whitebox_tools -r=TotalCurvature -v --wd="/path/to/data/" ^
--dem=DEM.dep -o=output.dep
```

6.7.40 Viewshed

Description: Identifies the viewshed for a point or set of points

Parameters:

Flag	Description
--dem	Input raster DEM file
--stations	Input viewing station raster file
-o, --output	Output raster file
--height	Viewing station height, in z units

Example Usage:

```
>>./whitebox_tools -r=Viewshed -v --wd="/path/to/data/" ^
--dem='dem.dep' --stations='stations.dep' -o=output.dep ^
--height=10.0
```

6.7.41 WetnessIndex

Description: Calculates the topographic wetness index, Ln(A / tan(slope))

Parameters:

Flag	Description
--sca	Input raster specific contributing area (SCA) file
--slope	Input raster slope file
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=WetnessIndex -v --wd="/path/to/data/" ^
--sca='flow_accum.dep' --slope='slope.dep' -o=output.dep
```

6.8 Hydrological Analysis

6.8.1 AverageFlowpathSlope

Description: Measures the average slope gradient from each grid cell to all upslope divide cells

Parameters:

Flag	Description
-i, --dem	Input raster DEM file
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=AverageFlowpathSlope -v ^
--wd="/path/to/data/" -i=DEM.dep -o=output.dep
```

6.8.2 AverageUpslopeFlowpathLength

Description: Measures the average length of all upslope flowpaths draining each grid cell

Parameters:

Flag	Description
-i, --dem	Input raster DEM file
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=AverageUpslopeFlowpathLength -v ^
--wd="/path/to/data/" -i=DEM.dep -o=output.dep
```

6.8.3 Basins

Description: Identifies drainage basins that drain to the DEM edge

Parameters:

Flag	Description
--d8_pntr	Input raster D8 pointer file
-o, --output	Output raster file
--esri_pntr	D8 pointer uses the ESRI style scheme

Example Usage:

```
>>./whitebox_tools -r=Basins -v --wd="/path/to/data/" ^
--d8_pntr='d8pntr.dep' -o='output.dep'
```

6.8.4 BreachDepressions

Description: Breaches all of the depressions in a DEM using Lindsay's (2016) algorithm. This should be preferred over depression filling in most cases

Parameters:

Flag	Description
-i, --dem	Input raster DEM file

Flag	Description
-o, --output	Output raster file
--max_depth	Optional maximum breach depth (default is Inf)
--max_length	Optional maximum breach channel length (in grid cells; default is Inf)

Example Usage:

```
>>./whitebox_tools -r=BreachDepressions -v ^
--wd="/path/to/data/" --dem=DEM.dep -o=output.dep
```

6.8.5 BreachSingleCellPits

Description: Removes single-cell pits from an input DEM by breaching

Parameters:

Flag	Description
-i, --dem	Input raster DEM file
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=BreachSingleCellPits -v ^
--wd="/path/to/data/" --dem=DEM.dep -o=output.dep
```

6.8.6 D8FlowAccumulation

Description: Calculates a D8 flow accumulation raster from an input DEM

Parameters:

Flag	Description
-i, --dem	Input raster DEM file
-o, --output	Output raster file
--out_type	Output type; one of 'cells', 'specific contributing area' (default), and 'catchment area'
--log	Optional flag to request the output be log-transformed
--clip	Optional flag to request clipping the display max by 1%

Example Usage:

```
>>./whitebox_tools -r=D8FlowAccumulation -v ^
```

```
--wd="/path/to/data/" --dem=DEM.dep -o=output.dep ^
--out_type='cells'
>>./whitebox_tools -r=D8FlowAccumulation -v ^
--wd="/path/to/data/" --dem=DEM.dep -o=output.dep ^
--out_type='specific catchment area' --log --clip
```

6.8.7 D8MassFlux

Description: Performs a D8 mass flux calculation

Parameters:

Flag	Description
--dem	Input raster DEM file
--loading	Input loading raster file
--efficiency	Input efficiency raster file
--absorption	Input absorption raster file
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=D8MassFlux -v --wd="/path/to/data/" ^
--dem=DEM.dep --loading=load.dep --efficiency=eff.dep ^
--absorption=abs.dep -o=output.dep
```

6.8.8 D8Pointer

Description: Calculates a D8 flow pointer raster from an input DEM

Parameters:

Flag	Description
-i, --dem	Input raster DEM file
-o, --output	Output raster file
--esri_pntr	D8 pointer uses the ESRI style scheme

Example Usage:

```
>>./whitebox_tools -r=D8Pointer -v --wd="/path/to/data/" ^
--dem=DEM.dep -o=output.dep
```

6.8.9 DInfFlowAccumulation

Description: Calculates a D-infinity flow accumulation raster from an input DEM

Parameters:

Flag	Description
-i, --dem	Input raster DEM file
-o, --output	Output raster file
--out_type	Output type; one of 'cells', 'sca' (default), and 'ca'
--threshold	Optional convergence threshold parameter, in grid cells; default is inifinity
--log	Optional flag to request the output be log-transformed
--clip	Optional flag to request clipping the display max by 1%

Example Usage:

```
>>./whitebox_tools -r=DInfFlowAccumulation -v ^
--wd="/path/to/data/" --dem=DEM.dep -o=output.dep ^
--out_type=sca
>>./whitebox_tools -r=DInfFlowAccumulation -v ^
--wd="/path/to/data/" --dem=DEM.dep -o=output.dep ^
--out_type=sca --threshold=10000 --log --clip
```

6.8.10 DInfMassFlux

Description: Performs a D-infinity mass flux calculation

Parameters:

Flag	Description
--dem	Input raster DEM file
--loading	Input loading raster file
--efficiency	Input efficiency raster file
--absorption	Input absorption raster file
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=DInfMassFlux -v --wd="/path/to/data/" ^
--dem=DEM.dep --loading=load.dep --efficiency=eff.dep ^
--absorption=abs.dep -o=output.dep
```

6.8.11 DInfPointer

Description: Calculates a D-infinity flow pointer (flow direction) raster from an input DEM

Parameters:

Flag	Description
-i, --dem	Input raster DEM file
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=DInfPointer -v --wd="/path/to/data/" ^
--dem=DEM.dep
```

6.8.12 DepthInSink

Description: Measures the depth of sinks (depressions) in a DEM

Parameters:

Flag	Description
-i, --dem	Input raster DEM file
-o, --output	Output raster file
--zero_background	Flag indicating whether the background value of zero should be used

Example Usage:

```
>>./whitebox_tools -r=DepthInSink -v --wd="/path/to/data/" ^
--dem=DEM.dep -o=output.dep --zero_background
```

6.8.13 DownslopeDistanceToStream

Description: Measures distance to the nearest downslope stream cell

Parameters:

Flag	Description
-i, --dem	Input raster DEM file
--streams	Input raster streams file
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=DownslopeDistanceToStream -v ^
--wd="/path/to/data/" --dem='dem.dep' --streams='streams.dep' ^
-o='output.dep'
```

6.8.14 DownslopeFlowpathLength

Description: Calculates the downslope flowpath length from each cell to basin outlet

Parameters:

Flag	Description
--d8_pntr	Input D8 pointer raster file
--watersheds	Optional input watershed raster file
--weights	Optional input weights raster file
-o, --output	Output raster file
--esri_pntr	D8 pointer uses the ESRI style scheme

Example Usage:

```
>>./whitebox_tools -r=DownslopeFlowpathLength -v ^
--wd="/path/to/data/" --d8_pntr=pointer.dep ^
-o=flowpath_len.dep
>>./whitebox_tools ^
-r=DownslopeFlowpathLength -v --wd="/path/to/data/" ^
--d8_pntr=pointer.flt --watersheds=basin.flt ^
--weights=weights.flt -o=flowpath_len.flt --esri_pntr
```

6.8.15 ElevationAboveStream

Description: Calculates the elevation of cells above the nearest downslope stream cell

Parameters:

Flag	Description
-i, --dem	Input raster DEM file
--streams	Input raster streams file
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=ElevationAboveStream -v ^
--wd="/path/to/data/" --dem='dem.dep' --streams='streams.dep' ^
-o='output.dep'
```

6.8.16 FD8FlowAccumulation

Description: Calculates an FD8 flow accumulation raster from an input DEM

Parameters:

Flag	Description
-i, --dem	Input raster DEM file
-o, --output	Output raster file
--out_type	Output type; one of 'cells', 'specific contributing area' (default), and 'catchment area'
--exponent	Optional exponent parameter; default is 1.1
--threshold	Optional convergence threshold parameter, in grid cells; default is inifinity
--log	Optional flag to request the output be log-transformed
--clip	Optional flag to request clipping the display max by 1%

Example Usage:

```
>>./whitebox_tools -r=FD8FlowAccumulation -v ^
--wd="/path/to/data/" --dem=DEM.dep -o=output.dep ^
--out_type='cells'
>>./whitebox_tools -r=FD8FlowAccumulation -v ^
--wd="/path/to/data/" --dem=DEM.dep -o=output.dep ^
--out_type='catchment area' --exponent=1.5 --threshold=10000 ^
--log --clip
```

6.8.17 FD8Pointer

Description: Calculates an FD8 flow pointer raster from an input DEM

Parameters:

Flag	Description
-i, --dem	Input raster DEM file
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=FD8Pointer -v --wd="/path/to/data/" ^
--dem=DEM.dep -o=output.dep
```

6.8.18 FillDepressions

Description: Fills all of the depressions in a DEM. Depression breaching should be preferred in most cases

Parameters:

Flag	Description
-i, --dem	Input raster DEM file
-o, --output	Output raster file
--fix_flats	Optional flag indicating whether flat areas should have a small gradient applied

Example Usage:

```
>>./whitebox_tools -r=FillDepressions -v ^
--wd="/path/to/data/" --dem=DEM.dep -o=output.dep ^
--fix_flats
```

6.8.19 FillSingleCellPits

Description: Raises pit cells to the elevation of their lowest neighbour

Parameters:

Flag	Description
-i, --dem	Input raster DEM file
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=FillSingleCellPits -v ^
--wd="/path/to/data/" --dem=DEM.dep -o=NewRaster.dep
```

6.8.20 FindNoFlowCells

Description: Finds grid cells with no downslope neighbours

Parameters:

Flag	Description
-i, --dem	Input raster DEM file
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=FindNoFlowCells -v ^
--wd="/path/to/data/" --dem=DEM.dep -o=NewRaster.dep
```

6.8.21 FindParallelFlow

Description: Finds areas of parallel flow in D8 flow direction rasters

Parameters:

Flag	Description
--d8_pntr	Input D8 pointer raster file
--streams	Input raster streams file
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=FindParallelFlow -v ^
--wd="/path/to/data/" --d8_pntr=pointer.dep ^
-o=out.dep
>>./whitebox_tools -r=FindParallelFlow -v ^
--wd="/path/to/data/" --d8_pntr=pointer.dep -o=out.dep ^
--streams='streams.dep'
```

6.8.22 FloodOrder

Description: Assigns each DEM grid cell its order in the sequence of inundations that are encountered during a search starting from the edges, moving inward at increasing elevations

Parameters:

Flag	Description
-i, --dem	Input raster DEM file
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=FloodOrder -v --wd="/path/to/data/" ^
--dem=DEM.dep -o=output.dep
```

6.8.23 FlowAccumulationFullWorkflow

Description: Resolves all of the depressions in a DEM, outputting a breached DEM, an aspect-aligned non-divergent flow pointer, a flow accumulation raster

Parameters:

Flag	Description
-i, --dem	Input raster DEM file
--out_dem	Output raster DEM file
--out_pntr	Output raster flow pointer file
--out_accum	Output raster flow accumulation file
--out_type	Output type; one of 'cells', 'sca' (default), and 'ca'
--log	Optional flag to request the output be log-transformed
--clip	Optional flag to request clipping the display max by 1%
--esri_pntr	D8 pointer uses the ESRI style scheme

Example Usage:

```
>>./whitebox_tools -r=FlowAccumulationFullWorkflow -v ^
--wd="/path/to/data/" --dem='DEM.dep' ^
--out_dem='DEM_filled.dep' --out_pntr='pointer.dep' ^
--out_accum='accum.dep' --out_type=sca --log --clip
```

6.8.24 FlowLengthDiff

Description: Calculates the local maximum absolute difference in downslope flowpath length, useful in mapping drainage divides and ridges

Parameters:

Flag	Description
--d8_pntr	Input D8 pointer raster file
-o, --output	Output raster file
--esri_pntr	D8 pointer uses the ESRI style scheme

Example Usage:

```
>>./whitebox_tools -r=FlowLengthDiff -v --wd="/path/to/data/" ^
--d8_pntr=pointer.dep -o=output.dep
```

6.8.25 Hillslopes

Description: Identifies the individual hillslopes draining to each link in a stream network

Parameters:

Flag	Description
--d8_pntr	Input raster D8 pointer file
--streams	Input raster streams file
-o, --output	Output raster file
--esri_pntr	D8 pointer uses the ESRI style scheme

Example Usage:

```
>>./whitebox_tools -r=Hillslopes -v --wd="/path/to/data/" ^
--d8_pntr='d8pntr.dep' --streams='streams.dep' ^
-o='output.dep'
```

6.8.26 Isobasins

Description: Divides a landscape into nearly equal sized drainage basins (i.e. watersheds)

Parameters:

Flag	Description
-i, --dem	Input raster DEM file
-o, --output	Output raster file
--size	Target basin size, in grid cells

Example Usage:

```
>>./whitebox_tools -r=Isobasins -v --wd="/path/to/data/" ^
--dem=DEM.dep -o=output.dep --size=1000
```

6.8.27 JensonSnapPourPoints

Description: Moves outlet points used to specify points of interest in a watershedding operation to the nearest stream cell

Parameters:

Flag	Description
--pour_pts	Input raster pour points (outlet) file
--streams	Input raster streams file
-o, --output	Output raster file
--snap_dist	Maximum snap distance in map units

Example Usage:

```
>>./whitebox_tools -r=JensonSnapPourPoints -v ^
--wd="/path/to/data/" --pour_pts='pour_pts.dep' ^
--streams='streams.dep' -o='output.dep' --snap_dist=15.0
```

6.8.28 MaxUpslopeFlowpathLength

Description: Measures the maximum length of all upslope flowpaths draining each grid cell

Parameters:

Flag	Description
-i, --dem	Input raster DEM file
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=MaxUpslopeFlowpathLength -v ^
--wd="/path/to/data/" -i=DEM.dep ^
-o=output.dep
>>./whitebox_tools -r=MaxUpslopeFlowpathLength -v ^
--wd="/path/to/data/" --dem=DEM.dep -o=output.dep --log ^
--clip
```

6.8.29 NumInflowingNeighbours

Description: Computes the number of inflowing neighbours to each cell in an input DEM based on the D8 algorithm

Parameters:

Flag	Description
-i, --dem	Input raster DEM file
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=NumInflowingNeighbours -v ^
--wd="/path/to/data/" -i=DEM.dep -o=output.dep
```

6.8.30 Rho8Pointer

Description: Calculates a stochastic Rho8 flow pointer raster from an input DEM

Parameters:

Flag	Description
-i, --dem	Input raster DEM file
-o, --output	Output raster file
--esri_pntr	D8 pointer uses the ESRI style scheme

Example Usage:

```
>>./whitebox_tools -r=Rho8Pointer -v --wd="/path/to/data/" ^
--dem=DEM.dep -o=output.dep
```

6.8.31 Sink

Description: Identifies the depressions in a DEM, giving each feature a unique identifier

Parameters:

Flag	Description
-i, --dem	Input raster DEM file
-o, --output	Output raster file
--zero_background	Flag indicating whether a background value of zero should be used

Example Usage:

```
>>./whitebox_tools -r=Sink -v --wd="/path/to/data/" ^
--dem=DEM.dep -o=output.dep --zero_background
```

6.8.32 SnapPourPoints

Description: Moves outlet points used to specify points of interest in a watershedding operation to the cell with the highest flow accumulation in its neighbourhood

Parameters:

Flag	Description
--pour_pts	Input raster pour points (outlet) file
--flow_accum	Input raster D8 flow accumulation file
-o, --output	Output raster file
--snap_dist	Maximum snap distance in map units

Example Usage:

```
>>./whitebox_tools -r=SnapPourPoints -v --wd="/path/to/data/" ^
--pour_pts='pour_pts.dep' --flow_accum='d8accum.dep' ^
-o='output.dep' --snap_dist=15.0
```

6.8.33 StrahlerOrderBasins

Description: Identifies Strahler-order basins from an input stream network

Parameters:

Flag	Description
--d8_pntr	Input raster D8 pointer file
--streams	Input raster streams file
-o, --output	Output raster file
--esri_pntr	D8 pointer uses the ESRI style scheme

Example Usage:

```
>>./whitebox_tools -r=StrahlerOrderBasins -v ^
--wd="/path/to/data/" --d8_pntr='d8pntr.dep' ^
--streams='streams.dep' -o='output.dep'
```

6.8.34 Subbasins

Description: Identifies the catchments, or sub-basin, draining to each link in a stream network

Parameters:

Flag	Description
--d8_pntr	Input D8 pointer raster file
--streams	Input raster streams file
-o, --output	Output raster file
--esri_pntr	D8 pointer uses the ESRI style scheme

Example Usage:

```
>>./whitebox_tools -r=Subbasins -v --wd="/path/to/data/" ^
--d8_pntr='d8pntr.dep' --streams='streams.dep' ^
-o='output.dep'
```

6.8.35 TraceDownslopeFlowpaths

Description: Traces downslope flowpaths from one or more target sites (i.e. seed points)

Parameters:

Flag	Description
--seed_pts	Input raster seed points file
--d8_pntr	Input D8 pointer raster file
-o, --output	Output raster file
--esri_pntr	D8 pointer uses the ESRI style scheme
--zero_background	Flag indicating whether a background value of zero should be used

Example Usage:

```
>>./whitebox_tools -r=TraceDownslopeFlowpaths -v ^
--wd="/path/to/data/" --seed_pts=seeds.dep ^
--flow_dir=flow_directions.dep --output=flow_paths.dep
```

6.8.36 Watershed

Description: Identifies the watershed, or drainage basin, draining to a set of target cells

Parameters:

Flag	Description
--d8_pntr	Input D8 pointer raster file
--pour_pts	Input vector pour points (outlet) file
-o, --output	Output raster file
--esri_pntr	D8 pointer uses the ESRI style scheme

Example Usage:

```
>>./whitebox_tools -r=Watershed -v --wd="/path/to/data/" ^
--d8_pntr='d8pntr.dep' --pour_pts='pour_pts.dep' ^
-o='output.dep'
```

6.9 Image Processing Tools

6.9.1 Closing

Description: A closing is a mathematical morphology operating involving an erosion (min filter) of a dilation (max filter) set

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file
--filterx	Size of the filter kernel in the x-direction
--filtery	Size of the filter kernel in the y-direction

Example Usage:

```
>>./whitebox_tools -r=Closing -v --wd="/path/to/data/" ^
-i=image.dep -o=output.dep --filter=25
```

6.9.2 CreateColourComposite

Description: Creates a colour-composite image from three bands of multispectral imagery

Parameters:

Flag	Description
--red	Input red band image file
--green	Input green band image file
--blue	Input blue band image file
--opacity	Input opacity band image file (optional)
-o, --output	Output colour composite file
--enhance	Optional flag indicating whether a balance contrast enhancement is performed

Example Usage:

```
>>./whitebox_tools -r=CreateColourComposite -v ^
--wd="/path/to/data/" --red=band3.dep --green=band2.dep ^
--blue=band1.dep -o=output.dep
>>./whitebox_tools ^
-r=CreateColourComposite -v --wd="/path/to/data/" ^
--red=band3.dep --green=band2.dep --blue=band1.dep ^
--opacity=a.dep -o=output.dep
```

6.9.3 FlipImage

Description: Reflects an image in the vertical or horizontal axis

Parameters:

Flag	Description
-i, --input	Input raster file

Flag	Description
-o, --output	Output raster file
--direction	Direction of reflection; options include 'v' (vertical), 'h' (horizontal), and 'b' (both)

Example Usage:

```
>>./whitebox_tools -r=FlipImage -v --wd="/path/to/data/" ^
--input=in.dep -o=out.dep --direction=h
```

6.9.4 IntegralImage

Description: Transforms an input image (summed area table) into its integral image equivalent

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=IntegralImage -v --wd="/path/to/data/" ^
-i=image.dep -o=output.dep
```

6.9.5 KMeansClustering

Description: Performs a k-means clustering operation on a multi-spectral dataset

Parameters:

Flag	Description
-i, --inputs	Input raster files
-o, --output	Output raster file
--out_html	Output HTML report file
--classes	Number of classes
--max_iterations	Maximum number of iterations
--class_change	Minimum percent of cells changed between iterations before completion
--initialize	How to initialize cluster centres?
--min_class_size	Minimum class size, in pixels

Example Usage:

```
>>./whitebox_tools -r=KMeansClustering -v ^
--wd='/path/to/data/' -i='image1.tif;image2.tif;image3.tif' ^
-o=output.tif --out_html=report.html --classes=15 ^
--max_iterations=25 --class_change=1.5 --initialize='random' ^
--min_class_size=500
```

6.9.6 LineThinning

Description: Performs line thinning a on Boolean raster image; intended to be used with the RemoveSpurs tool

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=LineThinning -v --wd="/path/to/data/" ^
--input=DEM.dep -o=output.dep
```

6.9.7 ModifiedKMeansClustering

Description: Performs a modified k-means clustering operation on a multi-spectral dataset

Parameters:

Flag	Description
-i, --inputs	Input raster files
-o, --output	Output raster file
--out_html	Output HTML report file
--start_clusters	Initial number of clusters
--merger_dist	Cluster merger distance
--max_iterations	Maximum number of iterations
--class_change	Minimum percent of cells changed between iterations before completion

Example Usage:

```
>>./whitebox_tools -r=ModifiedKMeansClustering -v ^
--wd='/path/to/data/' -i='image1.tif;image2.tif;image3.tif' ^
-o=output.tif --out_html=report.html --start_clusters=100 ^
--merger_dist=30.0 --max_iterations=25 --class_change=1.5
```

6.9.8 Mosaic

Description: Mosaics two or more images together

Parameters:

Flag	Description
-i, --inputs	Input raster files
-o, --output	Output raster file
--method	Resampling method

Example Usage:

```
>>./whitebox_tools -r=Mosaic -v --wd='/path/to/data/' ^
-i='image1.dep;image2.dep;image3.dep' -o=dest.dep ^
--method='cc
```

6.9.9 NormalizedDifferenceVegetationIndex

Description: Calculates the normalized difference vegetation index (NDVI) from near-infrared and red imagery

Parameters:

Flag	Description
--nir	Input near-infrared band image
--red	Input red band image
-o, --output	Output raster file
--clip	Optional amount to clip the distribution tails by, in percent
--osavi	Optional flag indicating whether the optimized soil-adjusted veg index (OSAVI) should be used

Example Usage:

```
>>./whitebox_tools -r=NormalizedDifferenceVegetationIndex -v ^
--wd="/path/to/data/" --nir=band4.dep --red=band3.dep ^
-o=output.dep
>>./whitebox_tools ^
-r=NormalizedDifferenceVegetationIndex -v --wd="/path/to/data/" ^
--nir=band4.dep --red=band3.dep -o=output.dep --clip=1.0 ^
--osavi
```

6.9.10 Opening

Description: An opening is a mathematical morphology operating involving a dilation (max filter) of an erosion (min filter) set

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file
--filterx	Size of the filter kernel in the x-direction
--filtery	Size of the filter kernel in the y-direction

Example Usage:

```
>>./whitebox_tools -r=Opening -v --wd="/path/to/data/" ^
-i=image.dep -o=output.dep --filter=25
```

6.9.11 RemoveSpurs

Description: Removes the spurs (pruning operation) from a Boolean line image.; intended to be used on the output of the LineThinning tool

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file
--iterations	Maximum number of iterations

Example Usage:

```
>>./whitebox_tools -r=RemoveSpurs -v --wd="/path/to/data/" ^
--input=DEM.dep -o=output.dep --iterations=10
```

6.9.12 Resample

Description: Resamples one or more input images into a destination image

Parameters:

Flag	Description
-i, --inputs	Input raster files
--destination	Destination raster file

Flag	Description
--method	Resampling method

Example Usage:

```
>>./whitebox_tools -r=Resample -v --wd='/path/to/data/' ^
-i='image1.dep;image2.dep;image3.dep' --destination=dest.dep ^
--method='cc
```

6.9.13 RgbToIhs

Description: Converts red, green, and blue (RGB) images into intensity, hue, and saturation (IHS) images

Parameters:

Flag	Description
--red	Input red band image file. Optionally specified if colour-composite not specified
--green	Input green band image file. Optionally specified if colour-composite not specified
--blue	Input blue band image file. Optionally specified if colour-composite not specified
--composite	Input colour-composite image file. Only used if individual bands are not specified
--intensity	Output intensity raster file
--hue	Output hue raster file
--saturation	Output saturation raster file

Example Usage:

```
>>./whitebox_tools -r=RgbToIhs -v --wd="/path/to/data/" ^
--red=band3.dep --green=band2.dep --blue=band1.dep ^
--intensity=intensity.dep --hue=hue.dep ^
--saturation=saturation.dep
>>./whitebox_tools -r=RgbToIhs -v ^
--wd="/path/to/data/" --composite=image.dep ^
--intensity=intensity.dep --hue=hue.dep ^
--saturation=saturation.dep
```

6.9.14 SplitColourComposite

Description: This tool splits an RGB colour composite image into seperate multispectral images

Parameters:

Flag	Description
-i, --input	Input colour composite image file
-o, --output	Output raster file (suffixes of '_r', '_g', and '_b' will be appended)

Example Usage:

```
>>./whitebox_tools -r=SplitColourComposite -v ^
--wd="/path/to/data/" -i=input.dep -o=output.dep
```

6.9.15 ThickenRasterLine

Description: Thickens single-cell wide lines within a raster image

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=ThickenRasterLine -v ^
--wd="/path/to/data/" --input=DEM.dep -o=output.dep
```

6.9.16 TophatTransform

Description: Performs either a white or black top-hat transform on an input image

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file
--filterx	Size of the filter kernel in the x-direction
--filtery	Size of the filter kernel in the y-direction
--variant	Optional variant value. Options include 'white' and 'black'

Example Usage:

```
>>./whitebox_tools -r=TophatTransform -v ^
--wd="/path/to/data/" -i=image.dep -o=output.dep --filter=25
```

6.9.17 WriteFunctionMemoryInsertion

Description: Performs a write function memory insertion for single-band multi-date change detection

Parameters:

Flag	Description
--i1, --input1	Input raster file associated with the first date
--i2, --input2	Input raster file associated with the second date
--i3, --input3	Optional input raster file associated with the third date
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=WriteFunctionMemoryInsertion -v ^
--wd="/path/to/data/" -i1=input1.dep -i2=input2.dep ^
-o=output.dep
```

6.10 Image Processing Tools => Filters

6.10.1 AdaptiveFilter

Description: Performs an adaptive filter on an image

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file
--filterx	Size of the filter kernel in the x-direction
--filtery	Size of the filter kernel in the y-direction
--threshold	Difference from mean threshold, in standard deviations

Example Usage:

```
>>./whitebox_tools -r=AdaptiveFilter -v --wd="/path/to/data/" ^
-i=DEM.dep -o=output.dep --filter=25 --threshold = 2.0
```

6.10.2 BilateralFilter

Description: A bilateral filter is an edge-preserving smoothing filter introduced by Tomasi and Manduchi (1998)

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file
--sigma_dist	Standard deviation in distance in pixels
--sigma_int	Standard deviation in intensity in pixels

Example Usage:

```
>>./whitebox_tools -r=BilateralFilter -v ^
--wd="/path/to/data/" -i=image.dep -o=output.dep ^
--sigma_dist=2.5 --sigma_int=4.0
```

6.10.3 ConservativeSmoothingFilter

Description: Performs a conservative-smoothing filter on an image

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file
--filterx	Size of the filter kernel in the x-direction
--filtery	Size of the filter kernel in the y-direction

Example Usage:

```
>>./whitebox_tools -r=ConservativeSmoothingFilter -v ^
--wd="/path/to/data/" -i=image.dep -o=output.dep --filter=25
```

6.10.4 DiffOfGaussianFilter

Description: Performs a Difference of Gaussian (DoG) filter on an image

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file
--sigma1	Standard deviation distance in pixels
--sigma2	Standard deviation distance in pixels

Example Usage:

```
>>./whitebox_tools -r=DiffOfGaussianFilter -v ^
--wd="/path/to/data/" -i=image.dep -o=output.dep --sigma1=2.0 ^
--sigma2=4.0
```

6.10.5 DiversityFilter

Description: Assigns each cell in the output grid the number of different values in a moving window centred on each grid cell in the input raster

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file
--filterx	Size of the filter kernel in the x-direction
--filtery	Size of the filter kernel in the y-direction

Example Usage:

```
>>./whitebox_tools -r=DiversityFilter -v ^
--wd="/path/to/data/" -i=image.dep -o=output.dep --filter=25
```

6.10.6 EmbossFilter

Description: Performs an emboss filter on an image, similar to a hillshade operation

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file
--direction	Direction of reflection; options include 'n', 's', 'e', 'w', 'ne', 'se', 'nw', 'sw'
--clip	Optional amount to clip the distribution tails by, in percent

Example Usage:

```
>>./whitebox_tools -r=EmbossFilter -v --wd="/path/to/data/" ^
-i=image.dep -o=output.dep --direction='s' --clip=1.0
```

6.10.7 GaussianFilter

Description: Performs a Gaussian filter on an image

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file
--sigma	Standard deviation distance in pixels

Example Usage:

```
>>./whitebox_tools -r=GaussianFilter -v --wd="/path/to/data/" ^
-i=image.dep -o=output.dep --sigma=2.0
```

6.10.8 HighPassFilter

Description: Performs a high-pass filter on an input image

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file
--filterx	Size of the filter kernel in the x-direction
--filtery	Size of the filter kernel in the y-direction

Example Usage:

```
>>./whitebox_tools -r=HighPassFilter -v --wd="/path/to/data/" ^
-i=image.dep -o=output.dep --filter=25
```

6.10.9 KNearestMeanFilter

Description: A k-nearest mean filter is a type of edge-preserving smoothing filter

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file
--filterx	Size of the filter kernel in the x-direction
--filtery	Size of the filter kernel in the y-direction
-k	k-value in pixels; this is the number of nearest-valued neighbours to use

Example Usage:

```
>>./whitebox_tools -r=KNearestMeanFilter -v ^
--wd="/path/to/data/" -i=image.dep -o=output.dep --filter=9 ^
-k=5
>>./whitebox_tools -r=KNearestMeanFilter -v ^
--wd="/path/to/data/" -i=image.dep -o=output.dep --filtery=7 ^
--filtery=9 -k=5
```

6.10.10 LaplacianFilter

Description: Performs a Laplacian filter on an image

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file
--variant	Optional variant value. Options include 3x3(1), 3x3(2), 3x3(3), 3x3(4), 5x5(1), and 5x5(2) (default is 3x3(1))
--clip	Optional amount to clip the distribution tails by, in percent

Example Usage:

```
>>./whitebox_tools -r=LaplacianFilter -v ^
--wd="/path/to/data/" -i=image.dep -o=output.dep ^
--variant='3x3(1)' --clip=1.0
```

6.10.11 LaplacianOfGaussianFilter

Description: Performs a Laplacian-of-Gaussian (LoG) filter on an image

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file
--sigma	Standard deviation in pixels

Example Usage:

```
>>./whitebox_tools -r=LaplacianOfGaussianFilter -v ^
--wd="/path/to/data/" -i=image.dep -o=output.dep --sigma=2.0
```

6.10.12 LeeFilter

Description: Performs a Lee (Sigma) smoothing filter on an image

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file
--filterx	Size of the filter kernel in the x-direction
--filtery	Size of the filter kernel in the y-direction
--sigma	Sigma value should be related to the standarad deviation of the distribution of image speckle noise
-m	M-threshold value the minimum allowable number of pixels within the intensity range

Example Usage:

```
>>./whitebox_tools -r=LeeFilter -v --wd="/path/to/data/" ^
-i=image.dep -o=output.dep --filter=9 --sigma=10.0 ^
-m=5
>>./whitebox_tools -r=LeeFilter -v --wd="/path/to/data/" ^
-i=image.dep -o=output.dep --filtery=7 --filtery=9 ^
--sigma=10.0 -m=5
```

6.10.13 LineDetectionFilter

Description: Performs a line-detection filter on an image

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file
--variant	Optional variant value. Options include 'v' (vertical), 'h' (horizontal), '45', and '135' (default is 'v')
--absvals	Optional flag indicating whether outputs should be absolute values
--clip	Optional amount to clip the distribution tails by, in percent

Example Usage:

```
>>./whitebox_tools -r=LineDetectionFilter -v ^
--wd="/path/to/data/" -i=image.dep -o=output.dep --variant=h ^
--clip=1.0
```

6.10.14 MajorityFilter

Description: Assigns each cell in the output grid the most frequently occurring value (mode) in a moving window centred on each grid cell in the input raster

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file
--filterx	Size of the filter kernel in the x-direction
--filtery	Size of the filter kernel in the y-direction

Example Usage:

```
>>./whitebox_tools -r=MajorityFilter -v --wd="/path/to/data/" ^
-i=image.dep -o=output.dep --filter=25
```

6.10.15 MaximumFilter

Description: Assigns each cell in the output grid the maximum value in a moving window centred on each grid cell in the input raster

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file
--filterx	Size of the filter kernel in the x-direction
--filtery	Size of the filter kernel in the y-direction

Example Usage:

```
>>./whitebox_tools -r=MaximumFilter -v --wd="/path/to/data/" ^
-i=image.dep -o=output.dep --filter=25
```

6.10.16 MeanFilter

Description: Performs a mean filter (low-pass filter) on an input image

Parameters:

Flag	Description
-i, --input	Input raster file

Flag	Description
-o, --output	Output raster file
--filterx	Size of the filter kernel in the x-direction
--filtery	Size of the filter kernel in the y-direction

Example Usage:

```
>>./whitebox_tools -r=MeanFilter -v --wd="/path/to/data/" ^
-i=image.dep -o=output.dep --filterx=25 --filtery=25
```

6.10.17 MedianFilter

Description: Performs a median filter on an input image

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file
--filterx	Size of the filter kernel in the x-direction
--filtery	Size of the filter kernel in the y-direction
--sig_digits	Number of significant digits

Example Usage:

```
>>./whitebox_tools -r=MedianFilter -v --wd="/path/to/data/" ^
-i=input.dep -o=output.dep --filter=25
```

6.10.18 MinimumFilter

Description: Assigns each cell in the output grid the minimum value in a moving window centred on each grid cell in the input raster

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file
--filterx	Size of the filter kernel in the x-direction
--filtery	Size of the filter kernel in the y-direction

Example Usage:

```
>>./whitebox_tools -r=MinimumFilter -v --wd="/path/to/data/" ^
-i=image.dep -o=output.dep --filter=25
```

6.10.19 OlympicFilter

Description: Performs an olympic smoothing filter on an image

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file
--filterx	Size of the filter kernel in the x-direction
--filtery	Size of the filter kernel in the y-direction

Example Usage:

```
>>./whitebox_tools -r=OlympicFilter -v --wd="/path/to/data/" ^
-i=image.dep -o=output.dep --filter=25
```

6.10.20 PercentileFilter

Description: Performs a percentile filter on an input image

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file
--filterx	Size of the filter kernel in the x-direction
--filtery	Size of the filter kernel in the y-direction
--sig_digits	Number of significant digits

Example Usage:

```
>>./whitebox_tools -r=PercentileFilter -v ^
--wd="/path/to/data/" -i=input.dep -o=output.dep --filter=25
```

6.10.21 PrewittFilter

Description: Performs a Prewitt edge-detection filter on an image

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file
--clip	Optional amount to clip the distribution tails by, in percent

Example Usage:

```
>>./whitebox_tools -r=PrewittFilter -v --wd="/path/to/data/" ^
-i=image.dep -o=output.dep --clip=1.0
```

6.10.22 RangeFilter

Description: Assigns each cell in the output grid the range of values in a moving window centred on each grid cell in the input raster

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file
--filterx	Size of the filter kernel in the x-direction
--filtery	Size of the filter kernel in the y-direction

Example Usage:

```
>>./whitebox_tools -r=RangeFilter -v --wd="/path/to/data/" ^
-i=image.dep -o=output.dep --filter=25
```

6.10.23 RobertsCrossFilter

Description: Performs a Robert's cross edge-detection filter on an image

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file
--clip	Optional amount to clip the distribution tails by, in percent

Example Usage:

```
>>./whitebox_tools -r=RobertsCrossFilter -v ^
```

```
--wd="/path/to/data/" -i=image.dep -o=output.dep --clip=1.0
```

6.10.24 ScharrFilter

Description: Performs a Scharr edge-detection filter on an image

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file
--clip	Optional amount to clip the distribution tails by, in percent

Example Usage:

```
>>./whitebox_tools -r=ScharrFilter -v --wd="/path/to/data/" ^
-i=image.dep -o=output.dep --clip=1.0
```

6.10.25 SobelFilter

Description: Performs a Sobel edge-detection filter on an image

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file
--variant	Optional variant value. Options include 3x3 and 5x5 (default is 3x3)
--clip	Optional amount to clip the distribution tails by, in percent (default is 0.0)

Example Usage:

```
>>./whitebox_tools -r=SobelFilter -v --wd="/path/to/data/" ^
-i=image.dep -o=output.dep --variant=5x5 --clip=1.0
```

6.10.26 StandardDeviationFilter

Description: Assigns each cell in the output grid the standard deviation of values in a moving window centred on each grid cell in the input raster

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file
--filterx	Size of the filter kernel in the x-direction
--filtery	Size of the filter kernel in the y-direction

Example Usage:

```
>>./whitebox_tools -r=StandardDeviationFilter -v ^
--wd="/path/to/data/" -i=image.dep -o=output.dep --filter=25
```

6.10.27 TotalFilter

Description: Performs a total filter on an input image

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file
--filterx	Size of the filter kernel in the x-direction
--filtery	Size of the filter kernel in the y-direction

Example Usage:

```
>>./whitebox_tools -r=TotalFilter -v --wd="/path/to/data/" ^
-i=image.dep -o=output.dep --filter=25
```

6.11 Image Processing Tools => Image Enhancement

6.11.1 BalanceContrastEnhancement

Description: Performs a balance contrast enhancement on a colour-composite image of multispectral data

Parameters:

Flag	Description
-i, --input	Input colour composite image file
-o, --output	Output raster file
--band_mean	Band mean value

Example Usage:

```
>>./whitebox_tools -r=BalanceContrastEnhancement -v ^
--wd="/path/to/data/" --input=image.dep -o=output.dep ^
--band_mean=120
```

6.11.2 DirectDecorrelationStretch

Description: Performs a direct decorrelation stretch enhancement on a colour-composite image of multi-spectral data

Parameters:

Flag	Description
-i, --input	Input colour composite image file
-o, --output	Output raster file
-k	Achromatic factor (k) ranges between 0 (no effect) and 1 (full saturation stretch), although typical values range from 0.3 to 0.7
--clip	Optional percent to clip the upper tail by during the stretch

Example Usage:

```
>>./whitebox_tools -r=DirectDecorrelationStretch -v ^
--wd="/path/to/data/" --input=image.dep -o=output.dep -k=0.4
```

6.11.3 GammaCorrection

Description: Performs a sigmoidal contrast stretch on input images

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file
--gamma	Gamma value

Example Usage:

```
>>./whitebox_tools -r=GammaCorrection -v ^
--wd="/path/to/data/" -i=input.dep -o=output.dep --gamma=0.5
```

6.11.4 HistogramEqualization

Description: Performs a histogram equalization contrast enhancment on an image

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file
--num_tones	Number of tones in the output image

Example Usage:

```
>>./whitebox_tools -r=HistogramEqualization -v ^
--wd="/path/to/data/" -i=input.dep -o=output.dep ^
--num_tones=1024
```

6.11.5 HistogramMatching

Description: Alters the statistical distribution of a raster image matching it to a specified PDF

Parameters:

Flag	Description
-i, --input	Input raster file
--histo_file	Input reference probability distribution function (pdf) text file
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=HistogramMatching -v ^
--wd="/path/to/data/" -i=input1.dep --histo_file=histo.txt ^
-o=output.dep
```

6.11.6 HistogramMatchingTwoImages

Description: This tool alters the cumulative distribution function of a raster image to that of another image

Parameters:

Flag	Description
--i1, --input1	Input raster file to modify
--i2, --input2	Input reference raster file
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=HistogramMatchingTwoImages -v ^
--wd="/path/to/data/" --i1=input1.dep --i2=input2.dep ^
-o=output.dep
```

6.11.7 MinMaxContrastStretch

Description: Performs a min-max contrast stretch on an input greytone image

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file
--min_val	Lower tail clip value
--max_val	Upper tail clip value
--num_tones	Number of tones in the output image

Example Usage:

```
>>./whitebox_tools -r=MinMaxContrastStretch -v ^
--wd="/path/to/data/" -i=input.dep -o=output.dep ^
--min_val=45.0 --max_val=200.0 --num_tones=1024
```

6.11.8 PanchromaticSharpening

Description: Increases the spatial resolution of image data by combining multispectral bands with panchromatic data

Parameters:

Flag	Description
--red	Input red band image file. Optionally specified if colour-composite not specified
--green	Input green band image file. Optionally specified if colour-composite not specified
--blue	Input blue band image file. Optionally specified if colour-composite not specified
--composite	Input colour-composite image file. Only used if individual bands are not specified
--pan	Input panchromatic band file
-o, --output	Output colour composite file
--method	Options include 'brovey' (default) and 'ihs'

Example Usage:

```
>>./whitebox_tools -r=PanchromaticSharpening -v ^
--wd="/path/to/data/" --red=red.dep --green=green.dep ^
--blue=blue.dep --pan=pan.dep --output=pan_sharp.dep ^
--method='brovey'
>>./whitebox_tools -r=PanchromaticSharpening ^
-v --wd="/path/to/data/" --composite=image.dep --pan=pan.dep ^
--output=pan_sharp.dep --method='ihs'
```

6.11.9 PercentageContrastStretch

Description: Performs a percentage linear contrast stretch on input images

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file
--clip	Optional amount to clip the distribution tails by, in percent
--tail	Specified which tails to clip; options include 'upper', 'lower', and 'both' (default is 'both')
--num_tones	Number of tones in the output image

Example Usage:

```
>>./whitebox_tools -r=PercentageContrastStretch -v ^
--wd="/path/to/data/" -i=input.dep -o=output.dep --clip=2.0 ^
--tail='both' --num_tones=1024
```

6.11.10 SigmoidalContrastStretch

Description: Performs a sigmoidal contrast stretch on input images

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file
--cutoff	Cutoff value between 0.0 and 0.95
--gain	Gain value
--num_tones	Number of tones in the output image

Example Usage:

```
>>./whitebox_tools -r=SigmoidalContrastStretch -v ^
--wd="/path/to/data/" -i=input.dep -o=output.dep --cutoff=0.1 ^
--gain=2.0 --num_tones=1024
```

6.11.11 StandardDeviationContrastStretch

Description: Performs a standard-deviation contrast stretch on input images

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file
--clip, --stdev	Standard deviation clip value
--num_tones	Number of tones in the output image

Example Usage:

```
>>./whitebox_tools -r=StandardDeviationContrastStretch -v ^
--wd="/path/to/data/" -i=input.dep -o=output.dep --stdev=2.0 ^
--num_tones=1024
```

6.12 LiDAR Tools

6.12.1 BlockMaximum

Description: Creates a block-maximum raster from an input LAS file

Parameters:

Flag	Description
-i, --input	Input LiDAR file
-o, --output	Output file
--resolution	Output raster's grid resolution

Example Usage:

```
>>./whitebox_tools -r=BlockMaximum -v --wd="/path/to/data/" ^
-i=file.las -o=outfile.dep --resolution=2.0"
./whitebox_tools ^
-r=BlockMaximum -v --wd="/path/to/data/" -i=file.las ^
-o=outfile.dep --resolution=5.0 --palette=light_quant.plt
```

6.12.2 BlockMinimum

Description: Creates a block-minimum raster from an input LAS file

Parameters:

Flag	Description
-i, --input	Input LiDAR file
-o, --output	Output file
--resolution	Output raster's grid resolution

Example Usage:

```
>>./whitebox_tools -r=BlockMinimum -v --wd="/path/to/data/" ^
-i=file.las -o=outfile.dep --resolution=2.0"
./whitebox_tools ^
-r=BlockMinimum -v --wd="/path/to/data/" -i=file.las ^
-o=outfile.dep --resolution=5.0 --palette=light_quant.plt
```

6.12.3 FilterLidarScanAngles

Description: Removes points in a LAS file with scan angles greater than a threshold

Parameters:

Flag	Description
-i, --input	Input LiDAR file
-o, --output	Output LiDAR file
--threshold	Scan angle threshold

Example Usage:

```
>>./whitebox_tools -r=FilterLidarScanAngles -v ^
--wd="/path/to/data/" -i="input.las" -o="output.las" ^
--threshold=10.0
```

6.12.4 FindFlightlineEdgePoints

Description: Identifies points along a flightline's edge in a LAS file

Parameters:

Flag	Description
-i, --input	Input LiDAR file

Flag	Description
-o, --output	Output file

Example Usage:

```
>>./whitebox_tools -r=FindFlightlineEdgePoints -v ^
--wd="/path/to/data/" -i="input.las" -o="output.las"
```

6.12.5 FlightlineOverlap

Description: Reads a LiDAR (LAS) point file and outputs a raster containing the number of overlapping flight lines in each grid cell

Parameters:

Flag	Description
-i, --input	Input LiDAR file
-o, --output	Output file
--resolution	Output raster's grid resolution

Example Usage:

```
>>./whitebox_tools -r=FlightlineOverlap -v ^
--wd="/path/to/data/" -i=file.las -o=outfile.dep ^
--resolution=2.0"
./whitebox_tools -r=FlightlineOverlap -v ^
--wd="/path/to/data/" -i=file.las -o=outfile.dep ^
--resolution=5.0 --palette=light_quant.plt
```

6.12.6 LasToAscii

Description: Converts one or more LAS files into ASCII text files

Parameters:

Flag	Description
-i, --inputs	Input LiDAR files

Example Usage:

```
>>./whitebox_tools -r=LasToAscii -v --wd="/path/to/data/" ^
-i="file1.las, file2.las, file3.las" -o=outfile.las"
```

6.12.7 LidarColourize

Description: Adds the red-green-blue colour fields of a LiDAR (LAS) file based on an input image

Parameters:

Flag	Description
--in_lidar	Input LiDAR file
--in_image	Input colour image file
-o, --output	Output LiDAR file

Example Usage:

```
>>./whitebox_tools -r=LidarColourize -v --wd="/path/to/data/" ^
--in_lidar="input.las" --in_image="image.dep" ^
-o="output.las"
```

6.12.8 LidarElevationSlice

Description: Outputs all of the points within a LiDAR (LAS) point file that lie between a specified elevation range

Parameters:

Flag	Description
-i, --input	Input LiDAR file
-o, --output	Output LiDAR file
--minz	Minimum elevation value (optional)
--maxz	Maximum elevation value (optional)
--class	Optional boolean flag indicating whether points outside the range should be retained in output but reclassified
--inclassval	Optional parameter specifying the class value assigned to points within the slice
--outclassval	Optional parameter specifying the class value assigned to points within the slice

Example Usage:

```
>>./whitebox_tools -r=LidarElevationSlice -v ^
--wd="/path/to/data/" -i="input.las" -o="output.las" ^
--minz=100.0 --maxz=250.0
>>./whitebox_tools ^
-r=LidarElevationSlice -v -i="/path/to/data/input.las" ^
-o="/path/to/data/output.las" --minz=100.0 --maxz=250.0 ^
```

```
--class
>>./whitebox_tools -r=LidarElevationSlice -v ^
-i="/path/to/data/input.las" -o="/path/to/data/output.las" ^
--minz=100.0 --maxz=250.0 --inclassval=1 --outclassval=0
```

6.12.9 LidarGroundPointFilter

Description: Identifies ground points within LiDAR dataset using a slope-based method

Parameters:

Flag	Description
-i, --input	Input LiDAR file
-o, --output	Output LiDAR file
--radius	Search Radius
--slope_threshold	Maximum inter-point slope to be considered an off-terrain point
--height_threshold	Inter-point height difference to be considered an off-terrain point

Example Usage:

```
>>./whitebox_tools -r=LidarGroundPointFilter -v ^
--wd="/path/to/data/" -i="input.las" -o="output.las" ^
--radius=10.0
```

6.12.10 LidarHillshade

Description: Calculates a hillshade value for points within a LAS file and stores these data in the RGB field

Parameters:

Flag	Description
-i, --input	Input LiDAR file
-o, --output	Output file
--azimuth	Illumination source azimuth in degrees
--altitude	Illumination source altitude in degrees
--radius	Search Radius

Example Usage:

```
>>./whitebox_tools -r=LidarHillshade -v --wd="/path/to/data/" ^
-i="input.las" -o="output.las" --radius=10.0
>>./whitebox_tools ^
-r=LidarHillshade -v --wd="/path/to/data/" -i="input.las" ^
```

```
-o="output.las" --azimuth=180.0 --altitude=20.0 --radius=1.0
```

6.12.11 LidarHistogram

Description: Creates a histogram from LiDAR data

Parameters:

Flag	Description
-i, --input	Input LiDAR file
-o, --output	Output HTML file (default name will be based on input file if unspecified)
--parameter	Parameter; options are 'elevation' (default), 'intensity', 'scan angle', 'class
--clip	Amount to clip distribution tails (in percent)

Example Usage:

```
>>./whitebox_tools -r=LidarHistogram -v --wd="/path/to/data/" ^
-i="file1.tif, file2.tif, file3.tif" -o=outfile.htm ^
--contiguity=Bishopsl
```

6.12.12 LidarIdwInterpolation

Description: Interpolates LAS files using an inverse-distance weighted (IDW) scheme

Parameters:

Flag	Description
-i, --input	Input LiDAR file (including extension)
-o, --output	Output raster file (including extension)
--parameter	Interpolation parameter; options are 'elevation' (default), 'intensity', 'class', 'scan angle', 'user data'
--returns	Point return types to include; options are 'all' (default), 'last', 'first'
--resolution	Output raster's grid resolution
--weight	IDW weight value
--radius	Search Radius
--exclude_cls	Optional exclude classes from interpolation; Valid class values range from 0 to 18, based on LAS specifications. Example, –exclude_cls='3,4,5,6,7,18'
--minz	Optional minimum elevation for inclusion in interpolation
--maxz	Optional maximum elevation for inclusion in interpolation

Example Usage:

```
>>./whitebox_tools -r=LidarIdwInterpolation -v ^
```

```
--wd="/path/to/data/" -i=file.las -o=outfile.dep ^
--resolution=2.0 --radius=5.0"
./whitebox_tools ^
-r=LidarIdwInterpolation --wd="/path/to/data/" -i=file.las ^
-o=outfile.dep --resolution=5.0 --weight=2.0 --radius=2.0 ^
--exclude_cls='3,4,5,6,7,18' --palette=light_quant.plt
```

6.12.13 LidarInfo

Description: Prints information about a LiDAR (LAS) dataset, including header, point return frequency, and classification data and information about the variable length records (VLRs) and geokeys

Parameters:

Flag	Description
-i, --input	Input LiDAR file
-o, --output	Output HTML file for regression summary report
--vlr	Flag indicating whether or not to print the variable length records (VLRs)
--geokeys	Flag indicating whether or not to print the geokeys

Example Usage:

```
>>./whitebox_tools -r=LidarInfo -v --wd="/path/to/data/" ^
-i=file.las --vlr --geokeys"
./whitebox_tools -r=LidarInfo ^
--wd="/path/to/data/" -i=file.las
```

6.12.14 LidarJoin

Description: Joins multiple LiDAR (LAS) files into a single LAS file

Parameters:

Flag	Description
-i, --inputs	Input LiDAR files
-o, --output	Output LiDAR file

Example Usage:

```
>>./whitebox_tools -r=LidarJoin -v --wd="/path/to/data/" ^
-i="file1.las, file2.las, file3.las" -o=outfile.las"
```

6.12.15 LidarKappaIndex

Description: Performs a kappa index of agreement (KIA) analysis on the classifications of two LAS files

Parameters:

Flag	Description
--i1, --input1	Input LiDAR classification file
--i2, --input2	Input LiDAR reference file
-o, --output	Output HTML file

Example Usage:

```
>>./whitebox_tools -r=LidarKappaIndex -v ^
--wd="/path/to/data/" --i1=class.tif --i2=reference.tif ^
-o=kia.html
```

6.12.16 LidarNearestNeighbourGridding

Description: Grids LAS files using nearest-neighbour scheme

Parameters:

Flag	Description
-i, --input	Input LiDAR file (including extension)
-o, --output	Output raster file (including extension)
--parameter	Interpolation parameter; options are 'elevation' (default), 'intensity', 'class', 'scan angle', 'user data'
--returns	Point return types to include; options are 'all' (default), 'last', 'first'
--resolution	Output raster's grid resolution
--radius	Search Radius
--exclude_cls	Optional exclude classes from interpolation; Valid class values range from 0 to 18, based on LAS specifications. Example, –exclude_cls='3,4,5,6,7,18'
--minz	Optional minimum elevation for inclusion in interpolation
--maxz	Optional maximum elevation for inclusion in interpolation

Example Usage:

```
>>./whitebox_tools -r=LidarNearestNeighbourGridding -v ^
--wd="/path/to/data/" -i=file.las -o=outfile.dep ^
--resolution=2.0 --radius=5.0"
./whitebox_tools ^
-r=LidarNearestNeighbourGridding --wd="/path/to/data/" ^
-i=file.las -o=outfile.dep --resolution=5.0 --radius=2.0 ^
```

```
--exclude_cls='3,4,5,6,7,18' --palette=light_quant.plt
```

6.12.17 LidarPointDensity

Description: Calculates the spatial pattern of point density for a LiDAR data set

Parameters:

Flag	Description
-i, --input	Input LiDAR file (including extension)
-o, --output	Output raster file (including extension)
--returns	Point return types to include; options are 'all' (default), 'last', 'first'
--resolution	Output raster's grid resolution
--radius	Search Radius
--exclude_cls	Optional exclude classes from interpolation; Valid class values range from 0 to 18, based on LAS specifications. Example, –exclude_cls='3,4,5,6,7,18'
--minz	Optional minimum elevation for inclusion in interpolation
--maxz	Optional maximum elevation for inclusion in interpolation

Example Usage:

```
>>./whitebox_tools -r=LidarPointDensity -v ^
--wd="/path/to/data/" -i=file.las -o=outfile.dep ^
--resolution=2.0 --radius=5.0"
./whitebox_tools ^
-r=LidarPointDensity -v --wd="/path/to/data/" -i=file.las ^
-o=outfile.dep --resolution=5.0 --radius=2.0 ^
--exclude_cls='3,4,5,6,7,18' --palette=light_quant.plt
```

6.12.18 LidarPointStats

Description: Creates several rasters summarizing the distribution of LAS point data

Parameters:

Flag	Description
-i, --input	Input LiDAR file
--resolution	Output raster's grid resolution
--num_points	Flag indicating whether or not to output the number of points raster
--num_pulses	Flag indicating whether or not to output the number of pulses raster
--z_range	Flag indicating whether or not to output the elevation range raster
--intensity_range	Flag indicating whether or not to output the intensity range raster
--predom_class	Flag indicating whether or not to output the predominant classification raster

Example Usage:

```
>>./whitebox_tools -r=LidarPointStats -v ^
--wd="/path/to/data/" -i=file.las --resolution=1.0 ^
--num_points
```

6.12.19 LidarRemoveOutliers

Description: Removes outliers (high and low points) in a LiDAR point cloud

Parameters:

Flag	Description
-i, --input	Input LiDAR file
-o, --output	Output LiDAR file
--radius	Search Radius
--elev_diff	Max. elevation difference

Example Usage:

```
>>./whitebox_tools -r=LidarRemoveOutliers -v ^
--wd="/path/to/data/" -i="input.las" -o="output.las" ^
--radius=10.0 --elev_diff=25.0
```

6.12.20 LidarSegmentation

Description: Segments a LiDAR point cloud based on normal vectors

Parameters:

Flag	Description
-i, --input	Input LiDAR file
-o, --output	Output file
--dist, --radius	Search Radius
--norm_diff	Maximum difference in normal vectors, in degrees
--maxzdiff	Maximum difference in elevation (z units) between neighbouring points of the same segment

Example Usage:

```
>>./whitebox_tools -r=LidarSegmentation -v ^
--wd="/path/to/data/" -i="input.las" -o="output.las" ^
--radius=10.0 --norm_diff=2.5 --maxzdiff=0.75
```

6.12.21 LidarSegmentationBasedFilter

Description: Identifies ground points within LiDAR point clouds using a segmentation based approach

Parameters:

Flag	Description
-i, --input	Input LiDAR file
-o, --output	Output file
--dist, --radius	Search Radius
--norm_diff	Maximum difference in normal vectors, in degrees
--maxzdiff	Maximum difference in elevation (z units) between neighbouring points of the same segment
--classify	Classify points as ground (2) or off-ground (1)

Example Usage:

```
>>./whitebox_tools -r=LidarSegmentationBasedFilter -v ^
--wd="/path/to/data/" -i="input.las" -o="output.las" ^
--radius=10.0 --norm_diff=2.5 --maxzdiff=0.75 --classify
```

6.12.22 LidarTile

Description: Tiles a LiDAR LAS file into multiple LAS files

Parameters:

Flag	Description
-i, --input	Input LiDAR file
--width_x	Width of tiles in the X dimension; default 1000.0
--width_y	Width of tiles in the Y dimension
--origin_x	Origin point X coordinate for tile grid
--origin_y	Origin point Y coordinate for tile grid
--min_points	Minimum number of points contained in a tile for it to be saved

Example Usage:

```
>>./whitebox_tools -r=LidarTile -v -i=/path/to/data/input.las ^
--width_x=1000.0 --width_y=2500.0 -=min_points=100
```

6.12.23 LidarTophatTransform

Description: Performs a white top-hat transform on a Lidar dataset; as an estimate of height above ground, this is useful for modelling the vegetation canopy

Parameters:

Flag	Description
-i, --input	Input LiDAR file
-o, --output	Output LiDAR file
--radius	Search Radius

Example Usage:

```
>>./whitebox_tools -r=LidarTophatTransform -v ^
--wd="/path/to/data/" -i="input.las" -o="output.las" ^
--radius=10.0
```

6.12.24 NormalVectors

Description: Calculates normal vectors for points within a LAS file and stores these data (XYZ vector components) in the RGB field

Parameters:

Flag	Description
-i, --input	Input LiDAR file
-o, --output	Output LiDAR file
--radius	Search Radius

Example Usage:

```
>>./whitebox_tools -r=NormalVectors -v --wd="/path/to/data/" ^
-i="input.las" -o="output.las" --radius=10.0
```

6.13 Math and Stats Tools

6.13.1 AbsoluteValue

Description: Calculates the absolute value of every cell in a raster

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=AbsoluteValue -v --wd="/path/to/data/" ^
-i='input.dep' -o=output.dep
```

6.13.2 Add

Description: Performs an addition operation on two rasters or a raster and a constant value

Parameters:

Flag	Description
--input1	Input raster file or constant value
--input2	Input raster file or constant value
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=Add -v --wd="/path/to/data/" ^
--input1='in1.dep' --input2='in2.dep' -o=output.dep
```

6.13.3 And

Description: Performs a logical AND operator on two Boolean raster images

Parameters:

Flag	Description
--input1	Input raster file
--input2	Input raster file
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=And -v --wd="/path/to/data/" ^
--input1='in1.dep' --input2='in2.dep' -o=output.dep
```

6.13.4 Anova

Description: Performs an analysis of variance (ANOVA) test on a raster dataset

Parameters:

Flag	Description
-i, --input	Input raster file
--features	Feature definition (or class) raster
-o, --output	Output HTML file

Example Usage:

```
>>./whitebox_tools -r=Anova -v --wd="/path/to/data/" ^
-i=data.tif --features=classes.tif -o=anova.html
```

6.13.5 ArcCos

Description: Returns the inverse cosine (arccos) of each values in a raster

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=ArcCos -v --wd="/path/to/data/" ^
-i='input.dep' -o=output.dep
```

6.13.6 ArcSin

Description: Returns the inverse sine (arcsin) of each values in a raster

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=ArcSin -v --wd="/path/to/data/" ^
-i='input.dep' -o=output.dep
```

6.13.7 ArcTan

Description: Returns the inverse tangent (arctan) of each values in a raster

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=ArcTan -v --wd="/path/to/data/" ^
-i='input.dep' -o=output.dep
```

6.13.8 Atan2

Description: Returns the 2-argument inverse tangent (atan2)

Parameters:

Flag	Description
--input_y	Input y raster file or constant value (rise)
--input_x	Input x raster file or constant value (run)
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=Atan2 -v --wd="/path/to/data/" ^
--input_y='in1.dep' --input_x='in2.dep' -o=output.dep
```

6.13.9 Ceil

Description: Returns the smallest (closest to negative infinity) value that is greater than or equal to the values in a raster

Parameters:

Flag	Description
-i, --input	Input raster file

Flag	Description
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=Ceil -v --wd="/path/to/data/" ^
-i='input.dep' -o=output.dep
```

6.13.10 Cos

Description: Returns the cosine (cos) of each values in a raster

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=Cos -v --wd="/path/to/data/" ^
-i='input.dep' -o=output.dep
```

6.13.11 Cosh

Description: Returns the hyperbolic cosine (cosh) of each values in a raster

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=Cosh -v --wd="/path/to/data/" ^
-i='input.dep' -o=output.dep
```

6.13.12 CrispnessIndex

Description: Calculates the Crispness Index, which is used to quantify how crisp (or conversely how fuzzy) a probability image is

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Optional output html file (default name will be based on input file if unspecified)

Example Usage:

```
>>./whitebox_tools -r=CrispnessIndex -v --wd="/path/to/data/" ^
-i=input.dep
>>./whitebox_tools -r=CrispnessIndex -v ^
--wd="/path/to/data/" -o=crispness.html
```

6.13.13 CrossTabulation

Description: Performs a cross-tabulation on two categorical images

Parameters:

Flag	Description
--i1, --input1	Input raster file 1
--i2, --input2	Input raster file 1
-o, --output	Output HTML file (default name will be based on input file if unspecified)

Example Usage:

```
>>./whitebox_tools -r=CrossTabulation -v ^
--wd="/path/to/data/" --i1="file1.tif" --i2="file2.tif" ^
-o=outfile.html
```

6.13.14 CumulativeDistribution

Description: Converts a raster image to its cumulative distribution function

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=CumulativeDistribution -v ^
--wd="/path/to/data/" -i=DEM.dep -o=output.dep
```

6.13.15 Decrement

Description: Decreases the values of each grid cell in an input raster by 1.0

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=Decrement -v --wd="/path/to/data/" ^
-i='input.dep' -o=output.dep
```

6.13.16 Divide

Description: Performs a division operation on two rasters or a raster and a constant value

Parameters:

Flag	Description
--input1	Input raster file or constant value
--input2	Input raster file or constant value
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=Divide -v --wd="/path/to/data/" ^
--input1='in1.dep' --input2='in2.dep' -o=output.dep
```

6.13.17 EqualTo

Description: Performs a equal-to comparison operation on two rasters or a raster and a constant value

Parameters:

Flag	Description
--input1	Input raster file or constant value
--input2	Input raster file or constant value

Flag	Description
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=EqualTo -v --wd="/path/to/data/" ^
--input1='in1.dep' --input2='in2.dep' -o=output.dep
```

6.13.18 Exp

Description: Returns the exponential (base e) of values in a raster

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=Exp -v --wd="/path/to/data/" ^
-i='input.dep' -o=output.dep
```

6.13.19 Exp2

Description: Returns the exponential (base 2) of values in a raster

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=Exp2 -v --wd="/path/to/data/" ^
-i='input.dep' -o=output.dep
```

6.13.20 ExtractRasterStatistics

Description: Extracts descriptive statistics for a group of patches in a raster

Parameters:

Flag	Description
-i, --input	Input data raster file
--features	Input feature definition raster file
-o, --output	Output raster file
--stat	Statistic to extract
--out_table	Output HTML Table file

Example Usage:

```
>>./whitebox_tools -r=ExtractRasterStatistics -v ^
--wd="/path/to/data/" -i='input.dep' --features='groups.dep' ^
-o='output.dep' --stat='minimum'
>>./whitebox_tools ^
-r=ExtractRasterStatistics -v --wd="/path/to/data/" ^
-i='input.dep' --features='groups.dep' ^
--out_table='output.html'
```

6.13.21 Floor

Description: Returns the largest (closest to positive infinity) value that is less than or equal to the values in a raster

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=Floor -v --wd="/path/to/data/" ^
-i='input.dep' -o='output.dep'
```

6.13.22 GreaterThan

Description: Performs a greater-than comparison operation on two rasters or a raster and a constant value

Parameters:

Flag	Description
--input1	Input raster file or constant value

Flag	Description
--input2	Input raster file or constant value
-o, --output	Output raster file
--incl_equals	Perform a greater-than-or-equal-to operation

Example Usage:

```
>>./whitebox_tools -r=GreaterThan -v --wd="/path/to/data/" ^
--input1='in1.dep' --input2='in2.dep' -o=output.dep ^
--incl_equals
```

6.13.23 ImageAutocorrelation

Description: Performs Moran's I analysis on two or more input images

Parameters:

Flag	Description
-i, --inputs	Input raster files
--contiguity	Contiguity type
-o, --output	Output HTML file (default name will be based on input file if unspecified)

Example Usage:

```
>>./whitebox_tools -r=ImageAutocorrelation -v ^
--wd="/path/to/data/" -i="file1.tif, file2.tif, file3.tif" ^
-o=outfile.html --contiguity=Bishops
```

6.13.24 ImageCorrelation

Description: Performs image correlation on two or more input images

Parameters:

Flag	Description
-i, --inputs	Input raster files
-o, --output	Output HTML file (default name will be based on input file if unspecified)

Example Usage:

```
>>./whitebox_tools -r=ImageCorrelation -v ^
--wd="/path/to/data/" -i="file1.tif, file2.tif, file3.tif" ^
```

```
-o=outfile.html
```

6.13.25 ImageRegression

Description: Performs image regression analysis on two input images

Parameters:

Flag	Description
--i1, --input1	Input raster file (independent variable, X)
--i2, --input2	Input raster file (dependent variable, Y)
-o, --output	Output HTML file for regression summary report
--out_residuals	Output raster regression resdidual file
--standardize	Optional flag indicating whether to standardize the residuals map

Example Usage:

```
>>./whitebox_tools -r=ImageRegression -v ^
--wd="/path/to/data/" --i1='file1.tif' --i2='file2.tif' ^
-o='outfile.html' --out_residuals='residuals.tif' ^
--standardize
```

6.13.26 Increment

Description: Increases the values of each grid cell in an input raster by 1.0

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=Increment -v --wd="/path/to/data/" ^
-i='input.dep' -o=output.dep
```

6.13.27 IntegerDivision

Description: Performs an integer division operation on two rasters or a raster and a constant value

Parameters:

Flag	Description
--input1	Input raster file or constant value
--input2	Input raster file or constant value
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=IntegerDivision -v ^
--wd="/path/to/data/" --input1='in1.dep' --input2='in2.dep' ^
-o=output.dep
```

6.13.28 IsNoData

Description: Identifies NoData valued pixels in an image

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=IsNoData -v --wd="/path/to/data/" ^
-i='input.dep' -o=output.dep
```

6.13.29 KSTestForNormality

Description: Evaluates whether the values in a raster are normally distributed

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output HTML file
--num_samples	Number of samples. Leave blank to use whole image

Example Usage:

```
>>./whitebox_tools -r=KSTestForNormality -v ^
--wd="/path/to/data/" -i=input.dep -o=output.html ^
--num_samples=1000
```

```
>>./whitebox_tools -r=KSTestForNormality -v ^
--wd="/path/to/data/" -i=input.dep -o=output.html
```

6.13.30 KappaIndex

Description: Performs a kappa index of agreement (KIA) analysis on two categorical raster files

Parameters:

Flag	Description
--i1, --input1	Input classification raster file
--i2, --input2	Input reference raster file
-o, --output	Output HTML file

Example Usage:

```
>>./whitebox_tools -r=KappaIndex -v --wd="/path/to/data/" ^
--i1=class.tif --i2=reference.tif -o=kia.html
```

6.13.31 LessThan

Description: Performs a less-than comparison operation on two rasters or a raster and a constant value

Parameters:

Flag	Description
--input1	Input raster file or constant value
--input2	Input raster file or constant value
-o, --output	Output raster file
--incl_equals	Perform a less-than-or-equal-to operation

Example Usage:

```
>>./whitebox_tools -r=LessThan -v --wd="/path/to/data/" ^
--input1='in1.dep' --input2='in2.dep' -o=output.dep ^
--incl_equals
```

6.13.32 Ln

Description: Returns the natural logarithm of values in a raster

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=Ln -v --wd="/path/to/data/" ^
-i='input.dep' -o=output.dep
```

6.13.33 Log10

Description: Returns the base-10 logarithm of values in a raster

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=Log10 -v --wd="/path/to/data/" ^
-i='input.dep' -o=output.dep
```

6.13.34 Log2

Description: Returns the base-2 logarithm of values in a raster

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=Log2 -v --wd="/path/to/data/" ^
-i='input.dep' -o=output.dep
```

6.13.35 Max

Description: Performs a MAX operation on two rasters or a raster and a constant value

Parameters:

Flag	Description
--input1	Input raster file or constant value
--input2	Input raster file or constant value
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=Max -v --wd="/path/to/data/" ^
--input1='in1.dep' --input2='in2.dep' -o=output.dep
```

6.13.36 Min

Description: Performs a MIN operation on two rasters or a raster and a constant value

Parameters:

Flag	Description
--input1	Input raster file or constant value
--input2	Input raster file or constant value
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=Min -v --wd="/path/to/data/" ^
--input1='in1.dep' --input2='in2.dep' -o=output.dep
```

6.13.37 Modulo

Description: Performs a modulo operation on two rasters or a raster and a constant value

Parameters:

Flag	Description
--input1	Input raster file or constant value
--input2	Input raster file or constant value
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=Modulo -v --wd="/path/to/data/" ^
--input1='in1.dep' --input2='in2.dep' -o=output.dep
```

6.13.38 Multiply

Description: Performs a multiplication operation on two rasters or a raster and a constant value

Parameters:

Flag	Description
--input1	Input raster file or constant value
--input2	Input raster file or constant value
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=Multiply -v --wd="/path/to/data/" ^
--input1='in1.dep' --input2='in2.dep' -o=output.dep
```

6.13.39 Negate

Description: Changes the sign of values in a raster or the 0-1 values of a Boolean raster

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=Negate -v --wd="/path/to/data/" ^
-i='input.dep' -o=output.dep
```

6.13.40 Not

Description: Performs a logical NOT operator on two Boolean raster images

Parameters:

Flag	Description
--input1	Input raster file
--input2	Input raster file
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=Not -v --wd="/path/to/data/" ^
--input1='in1.dep' --input2='in2.dep' -o=output.dep
```

6.13.41 NotEqualTo

Description: Performs a not-equal-to comparison operation on two rasters or a raster and a constant value

Parameters:

Flag	Description
--input1	Input raster file or constant value
--input2	Input raster file or constant value
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=NotEqualTo -v --wd="/path/to/data/" ^
--input1='in1.dep' --input2='in2.dep' -o=output.dep
```

6.13.42 Or

Description: Performs a logical OR operator on two Boolean raster images

Parameters:

Flag	Description
--input1	Input raster file
--input2	Input raster file
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=Or -v --wd="/path/to/data/" ^
--input1='in1.dep' --input2='in2.dep' -o=output.dep
```

6.13.43 Power

Description: Raises the values in grid cells of one rasters, or a constant value, by values in another raster or constant value

Parameters:

Flag	Description
--input1	Input raster file or constant value
--input2	Input raster file or constant value
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=Power -v --wd="/path/to/data/" ^
--input1='in1.dep' --input2='in2.dep' -o=output.dep
```

6.13.44 Quantiles

Description: Transforms raster values into quantiles

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file
--num_quantiles	Number of quantiles

Example Usage:

```
>>./whitebox_tools -r=Quantiles -v --wd="/path/to/data/" ^
-i=DEM.dep -o=output.dep --num_quantiles=5
```

6.13.45 RandomField

Description: Creates an image containing random values

Parameters:

Flag	Description
-i, --base	Input raster file
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=RandomField -v --wd="/path/to/data/" ^
--base=in.dep -o=out.dep
```

6.13.46 RandomSample

Description: Creates an image containing randomly located sample grid cells with unique IDs

Parameters:

Flag	Description
-i, --base	Input raster file
-o, --output	Output raster file
--num_samples	Number of samples

Example Usage:

```
>>./whitebox_tools -r=RandomSample -v --wd="/path/to/data/" ^
--base=in.dep -o=out.dep --num_samples=1000
```

6.13.47 RasterHistogram

Description: Creates a histogram from raster values

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output HTML file (default name will be based on input file if unspecified)

Example Usage:

```
>>./whitebox_tools -r=RasterHistogram -v ^
--wd="/path/to/data/" -i="file1.tif" -o=outfile.html
```

6.13.48 RasterSummaryStats

Description: Measures a rasters average, standard deviation, num. non-nodata cells, and total

Parameters:

Flag	Description
-i, --input	Input raster file

Example Usage:

```
>>./whitebox_tools -r=RasterSummaryStats -v ^
```

```
--wd="/path/to/data/" -i=DEM.dep
```

6.13.49 Reciprocal

Description: Returns the reciprocal (i.e. 1 / z) of values in a raster

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=Reciprocal -v --wd="/path/to/data/" ^
-i='input.dep' -o=output.dep
```

6.13.50 RescaleValueRange

Description: Performs a min-max contrast stretch on an input greytone image

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file
--out_min_val	New minimum value in output image
--out_max_val	New maximum value in output image
--clip_min	Optional lower tail clip value
--clip_max	Optional upper tail clip value

Example Usage:

```
>>./whitebox_tools -r=RescaleValueRange -v ^
--wd="/path/to/data/" -i=input.dep -o=output.dep ^
--out_min_val=0.0 --out_max_val=1.0
>>./whitebox_tools ^
-r=RescaleValueRange -v --wd="/path/to/data/" -i=input.dep ^
-o=output.dep --out_min_val=0.0 --out_max_val=1.0 ^
--clip_min=45.0 --clip_max=200.0
```

6.13.51 RootMeanSquareError

Description: Calculates the RMSE and other accuracy statistics

Parameters:

Flag	Description
-i, --input	Input raster file
--base	Input base raster file used for comparison

Example Usage:

```
>>./whitebox_tools -r=RootMeanSquareError -v ^
--wd="/path/to/data/" -i=DEM.dep
```

6.13.52 Round

Description: Rounds the values in an input raster to the nearest integer value

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=Round -v --wd="/path/to/data/" ^
-i='input.dep' -o=output.dep
```

6.13.53 Sin

Description: Returns the sine (sin) of each values in a raster

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=Sin -v --wd="/path/to/data/" ^
```

```
-i='input.dep' -o=output.dep
```

6.13.54 Sinh

Description: Returns the hyperbolic sine (sinh) of each values in a raster

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=Sinh -v --wd="/path/to/data/" ^
-i='input.dep' -o=output.dep
```

6.13.55 Square

Description: Squares the values in a raster

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=Square -v --wd="/path/to/data/" ^
-i='input.dep' -o=output.dep
```

6.13.56 SquareRoot

Description: Returns the square root of the values in a raster

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=SquareRoot -v --wd="/path/to/data/" ^
-i='input.dep' -o=output.dep
```

6.13.57 Subtract

Description: Performs a differencing operation on two rasters or a raster and a constant value

Parameters:

Flag	Description
--input1	Input raster file or constant value
--input2	Input raster file or constant value
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=Subtract -v --wd="/path/to/data/" ^
--input1='in1.dep' --input2='in2.dep' -o=output.dep
```

6.13.58 Tan

Description: Returns the tangent (tan) of each values in a raster

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=Tan -v --wd="/path/to/data/" ^
-i='input.dep' -o=output.dep
```

6.13.59 Tanh

Description: Returns the hyperbolic tangent (tanh) of each values in a raster

Parameters:

Flag	Description
-i, --input	Input raster file

Flag	Description
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=Tanh -v --wd="/path/to/data/" ^
-i='input.dep' -o=output.dep
```

6.13.60 ToDegrees

Description: Converts a raster from radians to degrees

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=ToDegrees -v --wd="/path/to/data/" ^
-i='input.dep' -o=output.dep
```

6.13.61 ToRadians

Description: Converts a raster from degrees to radians

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=ToRadians -v --wd="/path/to/data/" ^
-i='input.dep' -o=output.dep
```

6.13.62 Truncate

Description: Truncates the values in a raster to the desired number of decimal places

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file
--num_decimals	Number of decimals left after truncation (default is zero)

Example Usage:

```
>>./whitebox_tools -r=Truncate -v --wd="/path/to/data/" ^
-i='input.dep' -o=output.dep --num_decimals=2
```

6.13.63 TurningBandsSimulation

Description: Creates an image containing random values based on a turning-bands simulation

Parameters:

Flag	Description
-i, --base	Input base raster file
-o, --output	Output file
--range	The field's range, in xy-units, related to the extent of spatial autocorrelation
--iterations	The number of iterations

Example Usage:

```
>>./whitebox_tools -r=TurningBandsSimulation -v ^
--wd="/path/to/data/" --base=in.dep -o=out.dep --range=850.0 ^
--iterations=2500
```

6.13.64 Xor

Description: Performs a logical XOR operator on two Boolean raster images

Parameters:

Flag	Description
--input1	Input raster file
--input2	Input raster file
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=Xor -v --wd="/path/to/data/" ^
--input1='in1.dep' --input2='in2.dep' -o=output.dep
```

6.13.65 ZScores

Description: Standardizes the values in an input raster by converting to z-scores

Parameters:

Flag	Description
-i, --input	Input raster file
-o, --output	Output raster file

Example Usage:

```
>>./whitebox_tools -r=ZScores -v --wd="/path/to/data/" ^
-i=DEM.dep -o=output.dep
```

6.14 Stream Network Analysis

6.14.1 DistanceToOutlet

Description: Calculates the distance of stream grid cells to the channel network outlet cell

Parameters:

Flag	Description
--d8_pntr	Input raster D8 pointer file
--streams	Input raster streams file
-o, --output	Output raster file
--esri_pntr	D8 pointer uses the ESRI style scheme
--zero_background	Flag indicating whether a background value of zero should be used

Example Usage:

```
>>./whitebox_tools -r=DistanceToOutlet -v ^
--wd="/path/to/data/" --d8_pntr=D8.dep --streams=streams.dep ^
-o=output.dep
>>./whitebox_tools -r=DistanceToOutlet -v ^
--wd="/path/to/data/" --d8_pntr=D8.flt --streams=streams.flt ^
-o=output.flt --esri_pntr --zero_background
```

6.14.2 ExtractStreams

Description: Extracts stream grid cells from a flow accumulation raster

Parameters:

Flag	Description
--flow_accum	Input raster D8 flow accumulation file
-o, --output	Output raster file
--threshold	Threshold in flow accumulation values for channelization
--zero_background	Flag indicating whether a background value of zero should be used

Example Usage:

```
>>./whitebox_tools -r=ExtractStreams -v --wd="/path/to/data/" ^
--flow_accum='d8accum.dep' -o='output.dep' --threshold=100.0 ^
--zero_background
```

6.14.3 ExtractValleys

Description: Identifies potential valley bottom grid cells based on local topolography alone

Parameters:

Flag	Description
-i, --dem	Input raster DEM file
-o, --output	Output raster file
--variant	Options include 'lq' (lower quartile), 'JandR' (Johnston and Rosenfeld), and 'PandD' (Peucker and Douglas); default is 'lq'
--line_thin	Optional flag indicating whether post-processing line-thinning should be performed
--filter	Optional argument (only used when variant='lq') providing the filter size, in grid cells, used for lq-filtering (default is 5)

Example Usage:

```
>>./whitebox_tools -r=ExtractValleys -v --wd="/path/to/data/" ^
--dem=pointer.dep -o=out.dep --variant='JandR' ^
--line_thin
>>./whitebox_tools -r=ExtractValleys -v ^
--wd="/path/to/data/" --dem=pointer.dep -o=out.dep ^
--variant='lq' --filter=7 --line_thin
```

6.14.4 FarthestChannelHead

Description: Calculates the distance to the furthest upstream channel head for each stream cell

Parameters:

Flag	Description
--d8_pntr	Input raster D8 pointer file
--streams	Input raster streams file
-o, --output	Output raster file
--esri_pntr	D8 pointer uses the ESRI style scheme
--zero_background	Flag indicating whether a background value of zero should be used

Example Usage:

```
>>./whitebox_tools -r=FarthestChannelHead -v ^
--wd="/path/to/data/" --d8_pntr=D8.dep --streams=streams.dep ^
-o=output.dep
>>./whitebox_tools -r=FarthestChannelHead -v ^
--wd="/path/to/data/" --d8_pntr=D8.flt --streams=streams.flt ^
-o=output.flt --esri_pntr --zero_background
```

6.14.5 FindMainStem

Description: Finds the main stem, based on stream lengths, of each stream network

Parameters:

Flag	Description
--d8_pntr	Input raster D8 pointer file
--streams	Input raster streams file
-o, --output	Output raster file
--esri_pntr	D8 pointer uses the ESRI style scheme
--zero_background	Flag indicating whether a background value of zero should be used

Example Usage:

```
>>./whitebox_tools -r=FindMainStem -v --wd="/path/to/data/" ^
--d8_pntr=D8.dep --streams=streams.dep ^
-o=output.dep
>>./whitebox_tools -r=FindMainStem -v ^
--wd="/path/to/data/" --d8_pntr=D8.flt --streams=streams.flt ^
-o=output.flt --esri_pntr --zero_background
```

6.14.6 HackStreamOrder

Description: Assigns the Hack stream order to each tributary in a stream network

Parameters:

Flag	Description
--d8_pntr	Input raster D8 pointer file
--streams	Input raster streams file
-o, --output	Output raster file
--esri_pntr	D8 pointer uses the ESRI style scheme
--zero_background	Flag indicating whether a background value of zero should be used

Example Usage:

```
>>./whitebox_tools -r=HackStreamOrder -v ^
--wd="/path/to/data/" --d8_pntr=D8.dep --streams=streams.dep ^
-o=output.dep
>>./whitebox_tools -r=HackStreamOrder -v ^
--wd="/path/to/data/" --d8_pntr=D8.flt --streams=streams.flt ^
-o=output.flt --esri_pntr --zero_background
```

6.14.7 HortonStreamOrder

Description: Assigns the Horton stream order to each tributary in a stream network

Parameters:

Flag	Description
--d8_pntr	Input raster D8 pointer file
--streams	Input raster streams file
-o, --output	Output raster file
--esri_pntr	D8 pointer uses the ESRI style scheme
--zero_background	Flag indicating whether a background value of zero should be used

Example Usage:

```
>>./whitebox_tools -r=HortonStreamOrder -v ^
--wd="/path/to/data/" --d8_pntr=D8.dep --streams=streams.dep ^
-o=output.dep
>>./whitebox_tools -r=HortonStreamOrder -v ^
--wd="/path/to/data/" --d8_pntr=D8.flt --streams=streams.flt ^
-o=output.flt --esri_pntr --zero_background
```

6.14.8 LengthOfUpstreamChannels

Description: Calculates the total length of channels upstream

Parameters:

Flag	Description
--d8_pntr	Input raster D8 pointer file
--streams	Input raster streams file
-o, --output	Output raster file
--esri_pntr	D8 pointer uses the ESRI style scheme
--zero_background	Flag indicating whether a background value of zero should be used

Example Usage:

```
>>./whitebox_tools -r=LengthOfUpstreamChannels -v ^
--wd="/path/to/data/" --d8_pntr=D8.dep --streams=streams.dep ^
-o=output.dep
>>./whitebox_tools -r=LengthOfUpstreamChannels -v ^
--wd="/path/to/data/" --d8_pntr=D8.flt --streams=streams.flt ^
-o=output.flt --esri_pntr --zero_background
```

6.14.9 LongProfile

Description: Plots the stream longitudinal profiles for one or more rivers

Parameters:

Flag	Description
--d8_pntr	Input raster D8 pointer file
--streams	Input raster streams file
--dem	Input raster DEM file
-o, --output	Output HTML file
--esri_pntr	D8 pointer uses the ESRI style scheme

Example Usage:

```
>>./whitebox_tools -r=LongProfile -v --wd="/path/to/data/" ^
--d8_pntr=D8.dep --streams=streams.dep --dem=dem.dep ^
-o=output.html --esri_pntr
```

6.14.10 LongProfileFromPoints

Description: Plots the longitudinal profiles from flow-paths initiating from a set of vector points

Parameters:

Flag	Description
--d8_pntr	Input raster D8 pointer file
--points	Input vector points file
--dem	Input raster DEM file
-o, --output	Output HTML file
--esri_pntr	D8 pointer uses the ESRI style scheme

Example Usage:

```
>>./whitebox_tools -r=LongProfileFromPoints -v ^
--wd="/path/to/data/" --d8_pntr=D8.dep --points=stream_head.shp ^
--dem=dem.dep -o=output.html --esri_pntr
```

6.14.11 RemoveShortStreams

Description: Removes short first-order streams from a stream network

Parameters:

Flag	Description
--d8_pntr	Input raster D8 pointer file
--streams	Input raster streams file
-o, --output	Output raster file
--min_length	Minimum tributary length (in map units) used for network prunning
--esri_pntr	D8 pointer uses the ESRI style scheme

Example Usage:

```
>>./whitebox_tools -r=RemoveShortStreams -v ^
--wd="/path/to/data/" --d8_pntr=D8.dep --streams=streams.dep ^
-o=output.dep
```

6.14.12 ShreveStreamMagnitude

Description: Assigns the Shreve stream magnitude to each link in a stream network

Parameters:

Flag	Description
--d8_pntr	Input raster D8 pointer file
--streams	Input raster streams file
-o, --output	Output raster file
--esri_pntr	D8 pointer uses the ESRI style scheme
--zero_background	Flag indicating whether a background value of zero should be used

Example Usage:

```
>>./whitebox_tools -r=ShreveStreamMagnitude -v ^
--wd="/path/to/data/" --d8_pntr=D8.dep --streams=streams.dep ^
-o=output.dep
>>./whitebox_tools -r=ShreveStreamMagnitude -v ^
--wd="/path/to/data/" --d8_pntr=D8.flt --streams=streams.flt ^
-o=output.flt --esri_pntr --zero_background
```

6.14.13 StrahlerStreamOrder

Description: Assigns the Strahler stream order to each link in a stream network

Parameters:

Flag	Description
--d8_pntr	Input raster D8 pointer file
--streams	Input raster streams file
-o, --output	Output raster file
--esri_pntr	D8 pointer uses the ESRI style scheme
--zero_background	Flag indicating whether a background value of zero should be used

Example Usage:

```
>>./whitebox_tools -r=StrahlerStreamOrder -v ^
--wd="/path/to/data/" --d8_pntr=D8.dep --streams=streams.dep ^
-o=output.dep
>>./whitebox_tools -r=StrahlerStreamOrder -v ^
--wd="/path/to/data/" --d8_pntr=D8.flt --streams=streams.flt ^
-o=output.flt --esri_pntr --zero_background
```

6.14.14 StreamLinkClass

Description: Identifies the exterior/interior links and nodes in a stream network

Parameters:

Flag	Description
--d8_pntr	Input raster D8 pointer file
--streams	Input raster streams file
-o, --output	Output raster file
--esri_pntr	D8 pointer uses the ESRI style scheme
--zero_background	Flag indicating whether a background value of zero should be used

Example Usage:

```
>>./whitebox_tools -r=StreamLinkClass -v ^
--wd="/path/to/data/" --d8_pntr=D8.dep --streams=streams.dep ^
-o=output.dep
>>./whitebox_tools -r=StreamLinkClass -v ^
--wd="/path/to/data/" --d8_pntr=D8.flt --streams=streams.flt ^
-o=output.flt --esri_pntr --zero_background
```

6.14.15 StreamLinkIdentifier

Description: Assigns a unique identifier to each link in a stream network

Parameters:

Flag	Description
--d8_pntr	Input raster D8 pointer file
--streams	Input raster streams file
-o, --output	Output raster file
--esri_pntr	D8 pointer uses the ESRI style scheme
--zero_background	Flag indicating whether a background value of zero should be used

Example Usage:

```
>>./whitebox_tools -r=StreamLinkIdentifier -v ^
--wd="/path/to/data/" --d8_pntr=D8.dep --streams=streams.dep ^
-o=output.dep
>>./whitebox_tools -r=StreamLinkIdentifier -v ^
--wd="/path/to/data/" --d8_pntr=D8.flt --streams=streams.flt ^
-o=output.flt --esri_pntr --zero_background
```

6.14.16 StreamLinkLength

Description: Estimates the length of each link (or tributary) in a stream network

Parameters:

Flag	Description
--d8_pntr	Input raster D8 pointer file
--linkid	Input raster streams link ID (or tributary ID) file
-o, --output	Output raster file
--esri_pntr	D8 pointer uses the ESRI style scheme
--zero_background	Flag indicating whether a background value of zero should be used

Example Usage:

```
>>./whitebox_tools -r=StreamLinkLength -v ^
--wd="/path/to/data/" --d8_pntr=D8.dep --linkid=streamsID.dep ^
--dem=dem.dep -o=output.dep
>>./whitebox_tools ^
-r=StreamLinkLength -v --wd="/path/to/data/" --d8_pntr=D8.flt ^
--linkid=streamsID.flt --dem=dem.flt -o=output.flt --esri_pntr ^
--zero_background
```

6.14.17 StreamLinkSlope

Description: Estimates the average slope of each link (or tributary) in a stream network

Parameters:

Flag	Description
--d8_pntr	Input raster D8 pointer file
--linkid	Input raster streams link ID (or tributary ID) file
-i, --dem	Input raster DEM file
-o, --output	Output raster file
--esri_pntr	D8 pointer uses the ESRI style scheme
--zero_background	Flag indicating whether a background value of zero should be used

Example Usage:

```
>>./whitebox_tools -r=StreamLinkSlope -v ^
--wd="/path/to/data/" --d8_pntr=D8.dep --linkid=streamsID.dep ^
--dem=dem.dep -o=output.dep
>>./whitebox_tools ^
-r=StreamLinkSlope -v --wd="/path/to/data/" --d8_pntr=D8.flt ^
--linkid=streamsID.flt --dem=dem.flt -o=output.flt --esri_pntr ^
--zero_background
```

6.14.18 StreamSlopeContinuous

Description: Estimates the slope of each grid cell in a stream network

Parameters:

Flag	Description
--d8_pntr	Input raster D8 pointer file
--streams	Input raster streams file
-i, --dem	Input raster DEM file
-o, --output	Output raster file
--esri_pntr	D8 pointer uses the ESRI style scheme
--zero_background	Flag indicating whether a background value of zero should be used

Example Usage:

```
>>./whitebox_tools -r=StreamSlopeContinuous -v ^
--wd="/path/to/data/" --d8_pntr=D8.dep --linkid=streamsID.dep ^
--dem=dem.dep -o=output.dep
>>./whitebox_tools ^
-r=StreamSlopeContinuous -v --wd="/path/to/data/" ^
--d8_pntr=D8.flt --streams=streamsID.flt --dem=dem.flt ^
-o=output.flt --esri_pntr --zero_background
```

6.14.19 TopologicalStreamOrder

Description: Assigns each link in a stream network its topological order

Parameters:

Flag	Description
--d8_pntr	Input raster D8 pointer file
--streams	Input raster streams file
-o, --output	Output raster file
--esri_pntr	D8 pointer uses the ESRI style scheme
--zero_background	Flag indicating whether a background value of zero should be used

Example Usage:

```
>>./whitebox_tools -r=TopologicalStreamOrder -v ^
--wd="/path/to/data/" --d8_pntr=D8.dep --streams=streams.dep ^
-o=output.dep
>>./whitebox_tools -r=TopologicalStreamOrder -v ^
--wd="/path/to/data/" --d8_pntr=D8.flt --streams=streams.flt ^
```

```
-o=output.flt --esri_pntr --zero_background
```

6.14.20 TributaryIdentifier

Description: Assigns a unique identifier to each tributary in a stream network

Parameters:

Flag	Description
--d8_pntr	Input raster D8 pointer file
--streams	Input raster streams file
-o, --output	Output raster file
--esri_pntr	D8 pointer uses the ESRI style scheme
--zero_background	Flag indicating whether a background value of zero should be used

Example Usage:

```
>>./whitebox_tools -r=TributaryIdentifier -v ^
--wd="/path/to/data/" --d8_pntr=D8.dep --streams=streams.dep ^
-o=output.dep
>>./whitebox_tools -r=TributaryIdentifier -v ^
--wd="/path/to/data/" --d8_pntr=D8.flt --streams=streams.flt ^
-o=output.flt --esri_pntr --zero_background
```

7. Supported Data Formats

The **WhiteboxTools** library can currently support reading/writing raster data in *Whitebox GAT*, GeoTIFF, ESRI (ArcGIS) ASCII and binary (.flt & .hdr), GRASS GIS, Idrisi, SAGA GIS (binary and ASCII), and Surfer 7 data formats. The library is primarily tested using Whitebox raster data sets and if you encounter issues when reading/writing data in other formats, you should report the issue. Please note that there are no plans to incorporate third-party libraries, like GDAL, in the project given the design goal of keeping a pure (or as close as possible) Rust codebase.

At present, there is limited ability in *WhiteboxTools* to read vector geospatial data. Support for Shapefile (and other common vector formats) will be enhanced within the library soon.

LiDAR data can be read/written in the common LAS data format. *WhiteboxTools* can read and write LAS files that have been compressed (zipped with a .zip extension) using the common DEFLATE algorithm. Note that only LAS file should be contained within a zipped archive file. The compressed LiDAR format LAZ and ESRI LiDAR format are not currently supported by the library. The following is an example of running a LiDAR tool using zipped input/output files:

```
>>./whitebox_tools -r=LidarTophatTransform -v --wd="/path/to/data/"
-i="input.las.zip" -o="output.las.zip" --radius=10.0
```

Note that the double extensions (.las.zip) in the above command are not necessary and are only used for convenience of keeping track of LiDAR data sets (i.e. .zip extensions work too). The extra work of decoding/encoding compressed files does add additional processing time, although the Rust compression library that is used is highly efficient and usually only adds a few seconds to tool run times. Zipping LAS files frequently results 40-60% smaller binary files, making the additional processing time worthwhile for larger LAS file data sets with massive storage requirements.

8. Contributing

If you would like to contribute to the project as a developer, follow these instructions to get started:

1. Fork the larger Whitebox project (in which whitebox-tools exists) (https://github.com/jblindsay/whitebox-geospatial-analysis-tools)
2. Create your feature branch (git checkout -b my-new-feature)
3. Commit your changes (git commit -am 'Add some feature')
4. Push to the branch (git push origin my-new-feature)
5. Create a new Pull Request

Unless explicitly stated otherwise, any contribution intentionally submitted for inclusion in the work shall be licensed as above without any additional terms or conditions.

If you would like to contribute financial support for the project, please contact John Lindsay. We also welcome contributions in the form of media exposure. If you have written an article or blog about *WhiteboxTools* please let us know about it.

9. Reporting Bugs

WhiteboxTools is distributed as is and without warranty of suitability for application. If you encounter flaws with the software (i.e. bugs) please report the issue. Providing a detailed description of the conditions under which the bug occurred will help to identify the bug. *Use the Issues tracker on GitHub to report issues with the software and to request feature enchancements.* Please do not email Dr. Lindsay directly with bugs.

10. Known Issues and Limitations

- There is limited support for reading, writing, or analyzing vector data yet. Plans include native support for the ESRI Shapefile format and possibly GeoJSON data.
- The LAZ compressed LiDAR data format is currently unsupported although zipped LAS files (.zip) are.
- There is no support for reading waveform data contained within or associated with LAS files.
- File directories cannot contain apostrophes (', e.g. /John's data/) as they will be interpreted in the arguments array as single quoted strings.

- The Python scripts included with **WhiteboxTools** require Python 3. They will not work with Python 2, which is frequently the default Python version installed on many systems.

11. License

The **WhiteboxTools** library is distributed under the MIT license, a permissive open-source (free software) license.

12. Frequently Asked Questions

12.1 Do I need Whitebox GAT to use WhiteboxTools?

No you do not. You can call the tools contained within *WhiteboxTools* completely independent from the *Whitebox GAT* user interface using a Remote Procedure Call (RPC) approach. In fact, you can interact with the tools using Python scripting or directly, using a terminal application (command prompt). See *Interacting With* WhiteboxTools* From the Command Prompt* for further details.

12.2 How do I request a tool be added?

Eventually most of the tools in *Whitebox GAT* will be ported over to *WhiteboxTools* and all new tools will be added to this library as well. Naturally, this will take time. The order by which tools are ported is partly

a function of ease of porting, existing infrastructure (i.e. raster and LiDAR tools will be ported first since their is currently no support in the library for vector I/O), and interest. If you are interested in making a tool a higher priority for porting, email John Lindsay.

12.3 Can WhiteboxTools be incorporated into other software and open-source GIS projects?

WhiteboxTools was developed with the open-source GIS Whitebox GAT in mind. That said, the tools can be accessed independently and so long as you abide by the terms of the MIT license, there is no reason why other software and GIS projects cannot use *WhiteboxTools* as well. In fact, this was one of the motivating factors for creating the library in the first place. Feel free to use *WhiteboxTools* as the geospatial analysis engine in your open-source software project.

12.4 What platforms does WhiteboxTools support?

WhiteboxTools is developed using the Rust programming language, which supports a wide variety of platforms including MS Windows, MacOS, and Linux operating systems and common chip architectures. Interestingly, Rust also supports mobile platforms, and *WhiteboxTools* should therefore be capable of targeting (although no testing has been completed in this regard to date). Nearly all development and testing of the software is currently carried out on MacOS and we cannot guarantee a bug-free performance on other platforms. In particularly, MS Windows is the most different from the other platforms and is therefore the most likely to encounter platform-specific bugs. If you encounter bugs in the software, please consider reporting an issue using the GitHub support for issue-tracking.

12.5 What are the system requirements?

The answer to this question depends strongly on the type of analysis and data that you intend to process. However, generally we find performance to be optimal with a recommended minimum of 8-16GB of memory (RAM), a modern multi-core processor (e.g. 64-bit i5 or i7), and an solid-state-drive (SSD). It is likely that *WhiteboxTools* will have satisfactory performance on lower-spec systems if smaller datasets are being processed. Because *WhiteboxTools* reads entire raster datasets into system memory (for optimal performance, and in recognition that modern systems have increasingly larger amounts of fast RAM), this tends to be the limiting factor for the upper-end of data size successfully processed by the library. 64-bit operating systems are recommended and extensive testing has not been carried out on 32-bit OSs. See **"What platforms does WhiteboxTools support?"** for further details on supported platforms.

12.6 Are pre-compiled executables of WhiteboxTools available?

Pre-compiled binaries for *WhiteboxTools* can be downloaded from the *Geomorphometry and Hydrogeomatics Research Group* software web site for various supported operating systems. If you need binaries for

other operating systems/system architectures, you will need to compile the executable from source files. See Installation for details.

12.7 Why is WhiteboxTools programmed in Rust?

I spent a long time evaluating potential programming language for future development efforts for the *Whitebox GAT* project. My most important criterion for a language was that it compile to native code, rather than target the Java virtual machine (JVM). I have been keen to move Whitebox GAT away from Java because of some of the challenges that supporting the JVM has included for many Whitebox users. The language should be fast and productive–Java is already quite fast, but if I am going to change development languages, I would like a performance boost. Furthermore, given that many, though not all, of the algorithms used for geospatial analysis scale well with concurrent (parallel) implementations, I favoured languages that offered easy and safe concurrent programming. Although many would consider C/C++ for this work, I was looking for a modern and safe language. Fortunately, we are living through a renaissance period in programming language development and there are many newer languages that fit the bill nicely. Over the past two years, I considered each of Go, Rust, D, Nim, and Crystal for Whitebox development and ultimately decided on Rust. [See *GoSpatial* and *lidario*.]

Each of the languages I examined has its own advantages of disadvantages, so why Rust? It's a combination of factors that made it a compelling option for this project. Compared with many on the list, Rust is a mature language with a vibrant user community. Like C/C++, it's a high-performance and low-level language that allows for complete control of the system. However, Rust is also one of the safest languages, meaning that I can be confident that *WhiteboxTools* will not contain common bugs, such as memory use-after-release, memory leaks and race conditions within concurrent code. Importantly, and quite uniquely, this safety is achieved in the Rust language without the use of a garbage collector (automatic memory management). Garbage collectors can be great, but they do generally come with a certain efficiency trade-off that Rust does not have. The other main advantage of Rust's approach to memory management is that it allows for a level of interaction with scripting languages (e.g. Python) that is quite difficult to do in garbage collected languages. Although **WhiteboxTools** is currently set up to use an automation approach to interacting with Python code that calls it, I like the fact that I have the option to create a *WhiteboxTools* shared library.

Not everything with Rust is perfect however. It is still a very young language and there are many pieces still missing from its ecosystem. Furthermore, it is not the easiest language to learn, particularly for people who are inexperienced with programming. This may limit my ability to attract other programers to the Whitebox project, which would be unfortunate. However, overall, Rust was the best option for this particular application.

12.8 Do I need Rust installed on my computer to run WhiteboxTools?

No, you would only need Rust installed if you were compiling the *WhiteboxTools* codebase from source files.

12.9 How does WhiteboxTools' design philosophy differ?

Whitebox GAT is frequently praised for its consistent design and ease of use. Like *Whitebox GAT*, *WhiteboxTools* follows the convention of *one tool for one function*. For example, in *WhiteboxTools* assigning the links in a stream channel network their Horton, Strahler, Shreve, or Hack stream ordering numbers requires running separate tools (i.e. *HortonStreamOrder*, *StrahlerStreamOrder*, *ShreveStreamMagnitude*, and *HackStreamOrder*). By contrast, in GRASS GIS[1] and ArcGIS single tools (i.e. the *r.stream.order* and *Stream Order* tools respectively) can be configured to output different channel ordering schemes. The *WhiteboxTools* design is intended to simplify the user experience and to make it easier to find the right tool for a task. With more specific tool names that are reflective of their specific purposes, users are not as reliant on reading help documentation to identify the tool for the task at hand. Similarly, it is not uncommon for tools in other GIS to have multiple outputs. For example, in GRASS GIS the *r.slope.aspect* tool can be configured to output slope, aspect, profile curvature, plan curvature, and several other common terrain surface derivatives. Based on the *one tool for one function* design approach of *WhiteboxTools*, multiple outputs are indicative that a tool should be split into different, more specific tools. Are you more likely to go to a tool named *r.slope.aspect* or *TangentialCurvature* when you want to create a tangential curvature raster from a DEM? If you're new to the software and are unfamiliar with it, probably the later is more obvious. The *WhiteboxTools* design approach also has the added benefit of simplifying the documentation for tools. The one downside to this design approach, however, is that it results (or will result) in a large number of tools, often with signifcant overlap in function.

[1] NOTE: It's not my intent to criticize GRASS GIS, as I deeply respect the work that the GRASS developers have contributed. Rather, I am contrasting the consequences of *WhiteboxTools*' design philosophy to that of other GIS.