CHAPTER 13

Geomorphometry in ILWIS

T. Hengl, B.H.P. Maathuis and **L. Wang**

first steps in ILWIS · main functionalities — what it can and can't do? how to get support · importing and displaying DEMs · derivation and interpretation of land-surface parameters and objects · use of the hydro-processing module to derive drainage network and delineate catchments · use of ILWIS scripts · strong and weak points of ILWIS

1. ABOUT ILWIS

ILWIS is an acronym for Integrated Land and Water Information System, a stand alone integrated GIS package developed at the International Institute of Geoinformation Science and Earth Observations (ITC), Enschede, Netherlands. ILWIS was originally built for educational purposes and low-cost applications in developing countries. Its development started in 1984 and the first version (DOS version 1.0) was released in 1988. ILWIS 2.0 for Windows was released at the end of 1996, and a more compact and stable version 3.0 (WIN 95) was released by mid 2001. From 2004, ILWIS was distributed solely by ITC as shareware at a nominal price. From July 2007, ILWIS shifted to open source and ITC will not provide support for its further development.

> REMARK 1. *ILWIS is an acronym for Integrated Land and Water Information System, a stand-alone GIS and remote sensing package developed at the International Institute of Geoinformation Science and Earth Observations (ITC).*

The most recent version of ILWIS (3.4) offers a range of image processing, vector, raster, geostatistical, statistical, database and similar operations. In addition, a user can create new scripts, adjust the operation menus and even build Visual Basic, Delphi, or C++ applications that will run at top of ILWIS and use its internal functions. In principle, the biggest advantage of ILWIS is that it is a compact package with a diverse vector and raster-based GIS functionality and the biggest disadvantage are bugs and instabilities and necessity to import data to ILWIS format from other more popular GIS packages.

Developments in Soil Science, Volume 33 © 2009 Elsevier B.V.
ISSN 0166-2481, DOI: 10.1016/S0166-2481(08)00013-5.

1.1 Installing ILWIS

As per July 1st, 2007, ILWIS software is freely available ('as-is' and free of charge) as open source software (binaries and source code) under the 52°North initiative (http://52north.org). The ILWIS binaries are very simple to install. Copy the folder in the downloaded zip file. In this folder there is an `Ilwis30.exe` which is the main executable for ILWIS. Double click this file to start ILWIS.

You will first see the main program window, which can be compared to the ArcGIS catalog. The main program window is, in fact, a file browser which lists all ILWIS operations, objects and supplementary files within a working directory (see Figure 1). The ILWIS Main window consists of a Menu bar, a Standard toolbar, an Object selection toolbar, a Command line, a Catalog, a Status bar and an Operations/Navigator pane with an Operation-tree, an Operation-list and a Navigator. The left pane (Operations/Navigator) is used to browse available operations and directories and the right menu shows available spatial objects and supplementary files.

The user can adjust local settings of ILWIS by entering Preference under the main menu. In addition, the user can adjust also the catalog pane by choosing *View* $\mapsto$ *Customize catalog*. This can be very useful if in the same directory we also have GIS layers in different formats. For example, `DEM25m.asc` will not be visible in the catalog until we define `.asc` as external file extension. Note that, although ILWIS provides a possibility to directly write and read from files in external formats, in principle, it is always more efficient to first import all spatial objects to ILWIS format.

> REMARK 2. *There are four basic types of spatial objects in ILWIS: point, segment, polygon and raster maps. Supplementary files include: tables, coordinate systems, scripts, functions, domains, representations, etc.*

1.2 ILWIS operations

ILWIS offers a wide range of vector, raster and database operations that can be often combined together. An overview of possible operations can be seen from the main program window *Help* $\mapsto$ *Map and Table calculation* $\mapsto$ *Alphabetic overview of operators and functions*. For the purpose of land-surface parametrisation, the most important are the map calculation functions including neighbourhood and filtering operations. A special group of specific land-surface modelling operations is included in the module hydro-processing tools.

Note also that a practical aspect of ILWIS is that, every time a user runs an command from the menu bar or operation tree, ILWIS will record the operation in ILWIS command language. For example, you can import a shape file showing the contour lines from the 1:50,000 map by selecting *File* $\mapsto$ *Import* $\mapsto$ *ILWIS import* $\mapsto$ *Shape file*, which will be shown as:

```
import shape(contours50k.shp, contours50k)
```

on the ILWIS command line. This means that you can now edit this command and run it directly from the command line, instead of manually selecting the operations

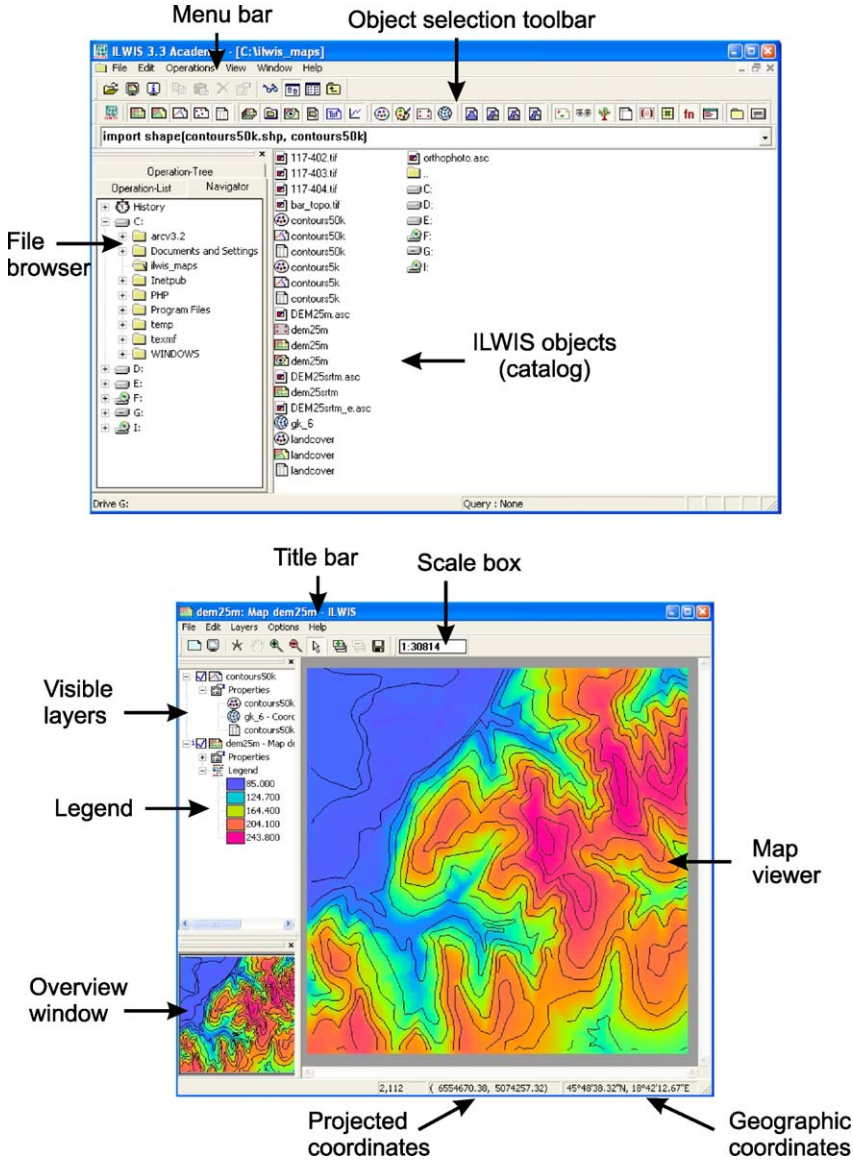

FIGURE 1 The ILWIS main window (above) and map window (below).

from the menu bar. In addition, you can copy such commands into an ILWIS script to enable automation of data analysis (see further Section 3.2).

The most used command in ILWIS is the `iff(condition,then,else)` command, which is often used to make spatial queries. Other commands can be easily understood just from their name. For example command `MapFilter(map.mpr,filter.fil)` will run a kernel filter (`filter.fil`) on an input

map (`map.mpr`). Arithmetical operations can be directly done by typing, for example:

```
mapC = mapA + mapB
```

1.3 ILWIS scripts

An ILWIS script gathers a sequenced list of ILWIS commands and expressions with a limited number of open parameters. Detailed instruction on how to create and run a script can be found in ILWIS 3.0 Academic User's Guide, Chapter 12. Parameters in scripts work similar to replaceable parameters in a DOS batch file. Open parameters can be coded using `%1`, `%2`, `%3`, up to `%9`. Scripts can be edited using the script tab in ILWIS, which offers editing of both commands, input parameters and their default values. If you have more than 9 variables, then you can create one master script that calls a number of sub-scripts. In that way, the number of parameters can be increased to infinity.

Once you create and save a script, you will see that ILWIS creates two auxiliary files: one `.isf` file which carries the definition of script parameters and an `.isl` file showing the list of commands. Both can be edited outside ILWIS using a text editor. When you have created a script and when you click the *Help* button in the *Run Script* dialog box for the first time, an HTML page will also be automatically generated listing all parameter names of the script and a minimal explanation. This HTML file is stored with the same name and in the same directory as the script and can be edited and modified according to your wishes.

> REMARK 3. *All operations in ILWIS can be run from command line using the ILWIS syntax. List of commands can be combined in ILWIS scripts to automate data processing.*

It is useful to know that remarks and comments within the scripts can be added by using the following commands:

- `rem` or `//` — this is an internal comment. All text on the line after `rem` or `//` is ignored from calculation.
- `begincomment endcomment` environment — has the same functionality as `rem` or `//` commands.
- `message` — this will create a text in message box on your screen. After pressing the OK button in the message box, the script will continue.

Comments and instructions can be fairly important because you can explain calculations and provide references.

After you have built and tested a script, it is advisable to copy it to the ILWIS program folder named `/Scripts/`, so that your script will be available from the operation menu every time you start ILWIS. You can customise the operation menu and operation tree to be able to find these operations much faster (see *ILWIS Help* ↦ *How to customize the Operation-tree, the Operation-list and the Operations menu*). To further customize the Operations menu, the Operation-list and the Operation-tree, advanced users may wish to modify the ILWIS `action.def` file that is located under the ILWIS program folder.

A script can be run by double clicking it from the ILWIS catalog or by typing the run script command in the ILWIS command line, e.g.:

```
run scriptname parameter1 parameter2...
```

2. IMPORTING AND DERIVING DEMS

In the most recent version of ILWIS, you can import GIS layers from a wide range of packages and formats. This is possible due to two built-in translation tools: GeoGateway (see list of supported formats at http://pcigeomatics.com) and GDAL (see list of supported formats at http://gdal.org). Elevation data, prepared as shape files and ESRI ArcInfo ascii grids, can be imported without difficulty. In ILWIS is also possible to import `.hgt` (HeiGhT) blocks, but then a general raster import needs to be used. For example, a command line to import the 1×1° SRTM 3 arcsec blocks, which consist of 1201×1201 pixels is:

```
name = map('name.hgt', genras, UseAs, 1201, 0, Int, 2,
SwapBytes)
```

where `name` is the name of the block and `genras` is the general raster map import command.

The following section will explain how to import an existing DEM or derive it from the sampled elevations. First, download the Baranja Hill dataset from geomorphometry.org and save it to a working directory, e.g. `/ilwismaps/` or similar. In this chapter we will work with sampled elevations (contours, height points) digitised from the 1:50,000 topo maps and the 30 m resolution SRTM image. In the case of SRTM DEM, elevations are available at all locations, while in the case of the contour lines, these are just sampled elevations that need to be interpolated to produce a DEM first. Now, import the contour map (`contours50.shp`), point map (`heights50.shp`) with measured heights and a raster mask map (`wbodies.asc`) showing water bodies using the standard import options. Also import the SRTM DEM (`DEM25srtm.asc`), which we will use for further comparison between the DEMs derived from contours and from satellite imagery.

Note that importing a grid file to ILWIS will always create a raster map, a georeference and a coordinate system — you might not need all of these. You can delete redundant coordinate systems and georeferences, but you need to first define in properties of imported maps the replacement grid definition and coordinate system.

2.1 Deriving DEM from sampled elevations

Before you create a DEM from sampled elevations, you need to create a grid definition. Here, you can use either the georeference produced automatically by ILWIS after importing the `DEM25srtm.asc`, or you can create your own grid definition. Use: (*File* ↦ *Create* ↦ *Georeference* ↦ *Corners*) for the output map. By default, we use the following parameters for the grid definition: pixel size of 25 m, and bounding coordinates *X*, *Y* (center of pixel): 6551884, 5070562; 6555609, 5074237. This will give you a raster image consisting of 149 rows and 147 columns.

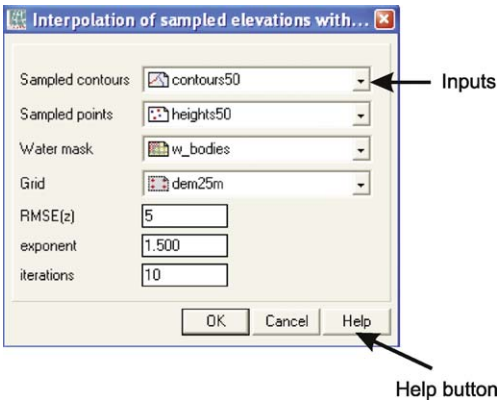

FIGURE 2 Running the DEM interpolation script.

In ILWIS, the default method to interpolate contours is the linear interpolator. The algorithm is described in more detail by Gorte and Koolhoven (1990). This command can be called directly from the *Main Menu* ↦ *Contour interpolation*. A more sophisticated approach is to use the script called `DEM_interpolation`, available from the `geomorphometry.org`. This will interpolate sampled elevations, then detect and filter out the *padi*-terraces and finally adjust elevation for the water bodies. By default, you can run the script using the following command:

```
run DEM_interpolation contours50.mps
heights50.mpp wbodies.mpr dem25m.grf 5 1.5 10
```

where `DEM_interpolation` is the script name, `contours50.mps`, `heights50.mpp` and `wbodies.mpr` are the input maps, `dem25m.grf` is the grid definition, `5` is estimated elevation error, `1.5` is exponent used to adjust for the water bodies and `10` is the maximum number of iterations allowed. A detailed description of the algorithm can be seen by selecting the Help button (Figure 2).

The script works as follows. First the input sampled elevation in segment[1] and point map are rasterized and glued using the target grid:

```
sampled01.mpr = MapRasterizeSegment(contours50.mps, dem25m.grf)
sampled02.mpr = MapRasterizePoint(heights50.mpp, dem25m.grf, 1)
sampled03.mpr = MapGlue(dem25m.grf,
sampled01.mpr, sampled02.mpr, replace)
```

Now we can interpolate the sampled values using:

```
DEM = MapInterpolContour(sampled03.mpr)
```

Of course, the resulting DEM will have many artefacts that will then propagate to land-surface parameters also. We first want to remove the *padi*-terraces, which are absolutely flat areas within the closed contours. These areas can be masked out from the original DEM by using the procedure first suggested by Pilouk and

[1] In ILWIS, segment map is a vector map with no topology, i.e. consisting of only lines.

Tempfli (1992) and further described by Hengl et al. (2004a). First, we need to detect *padi*-terraces using:

```
DEM_TER = iff((nbcnt(DEM#=DEM)>7), ?, DEM)
```

This will detect areas[2] (cut-offs) where more than seven neighbouring pixels have exactly the same elevation and put an undefined pixel "?". Now the medial axes can be detected using the distance operation with the rasterized map of contours:

```
CONT_dist = MapDistance(sampled01.mpr)
MED_AXES{dom=Bool.dom} =
iff((nbcnt(CONT_dist>CONT_dist#)>4), 1, 0)
```

Here the map MED_AXES shows detected valley bottoms and ridges, where value "1" or "True" represents the possible medial axes [Figure 3(b)]. We can attach to these areas some small constant value and then re-interpolate the DEM map. Before we do that, we need to detect which of these medial axes are ridges and which represent bottoms, i.e. which are convex and which concave shapes. Then we can add (concave) or subtract (convex) some arbitrary elevations to the medial axes.

The general shape of the land surface can be detected by using the neighbourhood operation[3]:

```
FORM_tmp{dom=Bool.dom} = iff(DEM>nbavg[2,4,6,8](DEM_TER#), 1,
iff(DEM_TER<nbavg[2,4,6,8](DEM_TER#), 0, ?))
```

The temporary shape map (FORM_tmp) needs to be extrapolated using the map iterations to fill the undefined pixels:

```
FORM_ext = MapIter(FORM_tmp.mpr,
iff(isundef(FORM_tmp), nbprd(FORM_tmp#), FORM_tmp))
FORM = MapIter(FORM_ext.mpr, nbprd(FORM_ext#), 5)
```

The last command is used to smooth the FORM map and reduce possible artefacts (we recommend at least 10 iterations). The derived map of the general land-surface shape can be seen in Figure 3(c). Finally a constant value (RMSE) is attached to the medial axes [Figure 3(d)] and the remaining undefined pixels are interpolated using linear interpolation:

```
DEM_tmp = iff(MED_AXES=True, iff((FORM=True) AND
(isundef(DEM_TER)), DEM+RMSE,
iff(FORM=False, DEM-RMSE, DEM_TER)), DEM_TER)
DEM_L1.mpr = MapInterpolContour(DEM_tmp.mpr)
```

2.2 Filtering of outliers

When using a DEM derived from satellite or airborne imagery (e.g. SRTM DEM), there can often be many artefacts (single pixels or lines) which are erroneous but

[2] This procedure will also detect true cut-offs, i.e. true lakes and water bodies.

[3] The convex land surfaces (ridges) receive value "1" or "True" and concave land surfaces (valleys) value "0" or "False". The *padi*-terrace areas will receive undefined value.

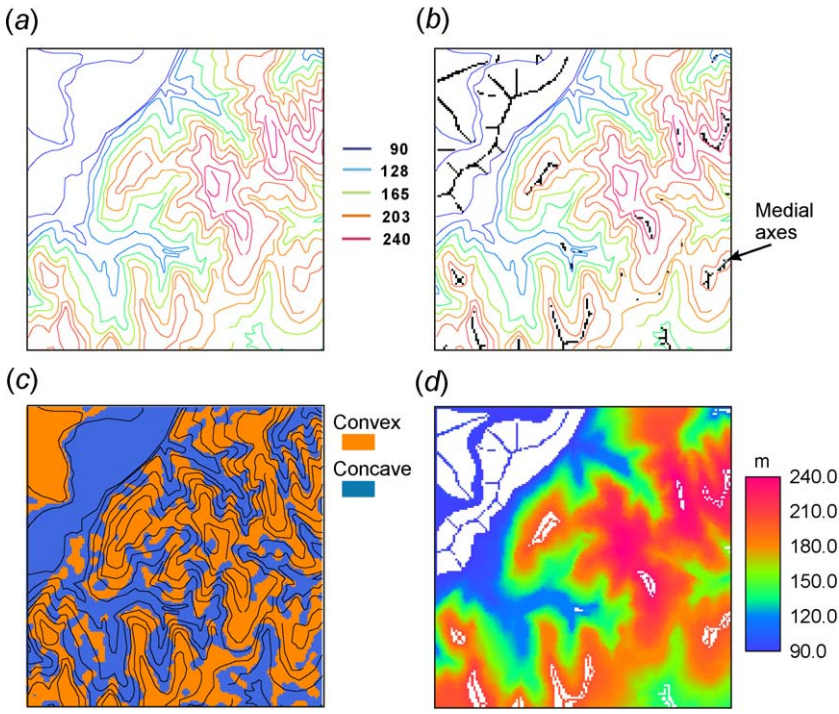

FIGURE 3 Addition of medial axes: (a) original (bulk) contour data; (b) detected medial axes in problematic areas (*padi*-terraces); (c) extrapolated shape of the land surface; and (d) temporary terrace-free map prior to interpolation of the remaining undefined pixels. (See page 722 in Colour Plate Section at the back of the book.)

not easily visible. To detect and filter them, we can apply the statistical procedure explained in Chapter 4. In ILWIS, this can be done using the script called `Filter_outliers`. The script will first predict the central value from the neighbours using the kriging weights calculated in a 5×5 window[4]:

```
Z_PRED = MapFilter(DEM.mpr, zpred.fil, 1)
```

where is the `zpred.fil` is a 5×5 predefined filter matrix calculated for the given covariance function (see Figure 8 in Chapter 4). Because the distances are fixed for the 5×5 window environment, these need to be estimated only once for a given variogram model of elevations. For example, an exponential variogram model with $C_0 = 0$, $C_1 = 1960$ and $R = 1057$ m will give the following weights (see Figure 4 in Chapter 1): $w_A = 0.260$, $w_B = 0.050$, $w_C = -0.025$, $w_D = -0.015$ and $w_E = -0.006$ (see Section 2 in Chapter 4). Now we can calculate the difference between the original and predicted elevation:

```
Z_DIF = DEM - Z_PRED
```

[4] Ideally, one should use a much larger window to filter out the outlier, but this could be computationally demanding for large datasets.

You can display the Z_DIF to see if the values are really normally distributed. The overall average should be 0 and standard deviation should not exceed RMSE(z) as defined for the original dataset. We can then standardise the difference between the predicted and observed value and derive the normal probability of this difference:

```
Z_DIFS = Z_DIF/S_DIF
Z_PROB = (1/sqrt(PI2))*exp(-sq(Z_DIFS)/2)/0.4
```

This probability is then used as the weight function to derive the smoothed DEM:

```
DEM_flt = Z_PROB*DEM + (1-Z_PROB)*Z_PRED
```

where Z_PROB is the normal probability to find a certain value[5] and Z_PRED is the map of elevations predicted from the neighbours.

Note that these filtering steps do not guarantee that all artefacts will be removed (a DEM anyway always carries a measurement error). It is advisable to check the output results and, if needed, digitize extra contours. The above-listed filtering script should also be used with caution because it is possible that also a small number of real features such as small lakes and depressions that can occur naturally will be removed by mistake.

2.3 DEM hydro-preprocessing

A set of built in procedures can be used to prepare the DEM for hydrological processing. These include: (a) removal of sinks, (b) flow optimisation and (c) topological optimisation.

Removal/filling of sinks — Fill sinks operation reduces local depressions (single and multiple pixels). The height value of a single-pixel depression is raised to smallest value of the 8 neighbours of a single-pixel depression and height values of a local depression consisting of multiple pixels are raised to the smallest value of a pixel that is both adjacent to the outlet for the depression and that would discharge into the initial depression. This will ensure that flow direction will be found for every pixel in the map.

ILWIS offers a range of simple procedures to reduce spurious sinks that might not perform equally successful in all areas. More advanced users should consult the work of Lindsay and Creed (2005, 2006).

Note that the flow extraction process allows the occurrences of undefined areas, representing e.g. closed basins, glacial lakes, depressions (sinkholes) within a limestone area or manmade features like reservoirs. These areas are therefore not modified during the fill sink routine. The flow accumulation computation stops at these locations and at a later stage, manually, the topology can be adapted to represent proper flow connection.

> REMARK 4. *ILWIS includes a set of built-in procedures for hydrological processing: removal of sinks, flow optimization and topological optimisation.*

[5] We use the Gaussian function and then simply divide estimated value with the maximum value.

Flow optimisation — This procedure will '*burn*' existing drainage features into a Digital Elevation Model (DEM), so that a subsequent Flow direction operation on the output DEM will better follow the existing drainage pattern. To achieve this, you will need a segment map of the current drainage network and estimate of the smooth drop and sharp drop values. The processed DEM will show (a) gradual drop of (drainage) segments in the output DEM, over a certain distance to the (drainage) segments; (b) gradual raise of (watershed-divide) segments on the output DEM, over a certain distance to the (watershed-divide) segments; (c) additional sharp drop or raise of segments on top of the gradual drop or raise; and (d) simple drop or raise of polygons in the output DEM.

Topological optimisation — Topological consistency can be improved for those areas having undefined DEM values (representing lakes or a reservoir). Before you can perform this operation, you must first prepare a segment map in which the segments connect the inlet(s) of a lake with the outlet(s) of lake (down flow). You can also extract the satellite image based drainage for flat areas and through this manual intervention correct the parallel drainage line occurrences in flat areas. You should first generate a default network, that can be superimposed on a satellite image and then manually adjusted it.

As a result of DEM hydro-preprocessing, you should have a hydrologically consistent raster based elevation representation. Additional modifications might be required as the elevation value assigned to a pixel is an averaged representation only. Furthermore, due to raster resolution in relation to the drainage network or valley width (land-surface discretisation does not allow representation of features smaller than the pixel size) or intrinsic properties of the sensor that acquired the DEM (reflective surface instead of the actual ground surface as is the case with active sensors derived raw elevation models) additional pre-processing is necessary. To overcome the resolution problem, more detailed elevation raster data should be obtained from larger scale aerial photographs or optical stereo satellite images (Aster, SPOT-5 HRS or the ALOS Prism, once operational).

2.4 Visualization of DEMs

In ILWIS, it possible to prepare a 2D visualization of land surface by using a built-in multi illumination angle script. This will produce a colour composite in which the DEM is illuminated from three main directions, the North in red, the North-West in green and the West in blue. The graduated coloured elevation information can be displayed on top of this map as a transparent layer. The example is given in Figure 4.

3. DERIVING PARAMETERS AND OBJECTS

There are two ways to derive LSPs/LSOs in ILWIS: (a) by using built in commands and (b) by building and running scripts. At the moment, ILWIS has only a limited number of built-in land-surface parameterisation algorithms. These are mainly focused on derivation of land-surface parameters related to hydrology.

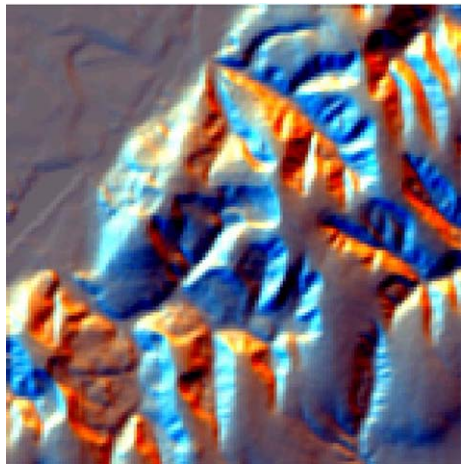

FIGURE 4 Visualization of the DEMs using the multi illuminated angles in ILWIS. (See page 722 in Colour Plate Section at the back of the book.)

3.1 Built-in geomorphometric operations

The version 3.3 of ILWIS has built in hydro-processing module that supports further DEM processing to obtain a full raster and vector based (including topology) schematisation of the (sub-)catchments and drainage network, coupled with additional hydrological relevant parameters (Figure 5). This module is described in detail by Maathuis and Wang (2006) and in ILWIS help files.

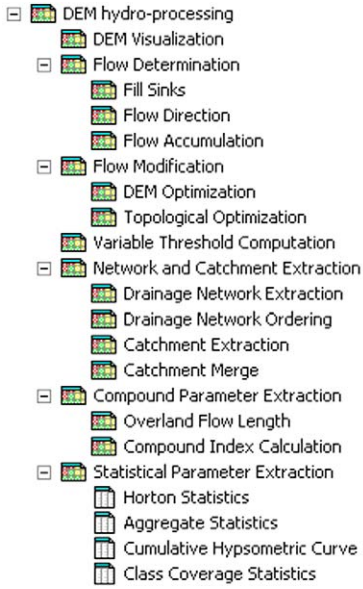

FIGURE 5 ILWIS DEM hydro operations.

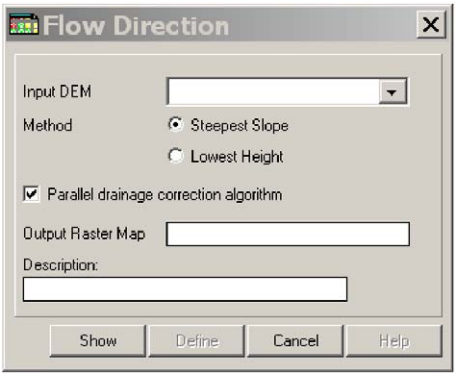

FIGURE 6 ILWIS menu for Flow direction.

The hydro-processing module facilitates various hydrological analysis: extraction of flow direction and flow accumulation, extraction of drainage network and overland flow, extraction of hydrological (flow) indices and statistical parameters. At the moment, only the Deterministic-8 flow model can be used as a built in operation in ILWIS. More advanced models, such as the D-Infinity (Tarboton, 1997) or the Mass Flux algorithm (Section 3.2 in Chapter 7) are under consideration. The raster and vector maps as well as the tables generated in ILWIS can be exported to other formats for incorporation into other software routines.

Flow determination — This module will extract the flow direction (aspect of flow for neighbour pixels — N, NE, etc.) and flow accumulation (cumulative count of the number of pixels that naturally drain into outlets). Optionally a parallel drainage correction algorithm can be incorporated as described by Garbrecht and Martz (1997) to handle the flat areas by imposing an artificial gradient (Figure 6).

Flow direction can be extracted according to the steepest downhill slope between a central pixel and its 8 neighbour pixels[6] or according to the position of the pixel with the smallest elevation.

The current method to address the problem of drainage analysis over flat areas has been implemented in conjunction with the D-8 flow-routine approach. This implementation takes a sink-free DEM as input, and assigns flow direction based on the slope. To the flat areas, where there is no flows (areas with unsolved flow directions), pixels are flagged as *unsolved*. Unsolved directions are resolved by making them to flow toward a neighbour of equal elevation that has a flow direction resolved. The method results in the parallel drainage patterns (as illustrated in Figure 7, left) which requires a lot further manual correction works in order to make a realistic flow.

> REMARK 5. *The true advantage of ILWIS is the possibility of combining vector, raster and database operations together with geomorphometric analysis.*

In the most recent version of ILWIS, a new approach as described by Garbrecht and Martz (1997) is adapted to obtain a more realistic model of flow. This new

[6] To run the multi-direction flow, you can use the script `Flow_indices`, described in Section 3.2.

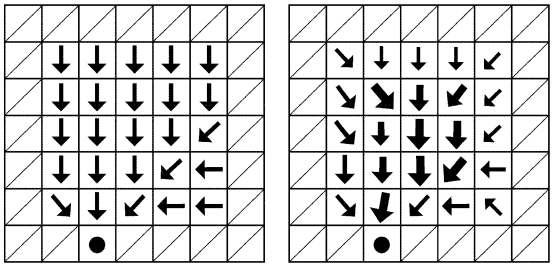

FIGURE 7 Parallel (left) and realistic (right) drainage patterns. The latter is now implemented in the latest version of ILWIS.

approach still makes use of the D-8 flow-routine approach as done in the existing DEM-hydro routine model. Since the D-8 method cannot define the flows over pixels of equal elevations (flat areas), this algorithm will increase the cell elevations of the flat area and produce the desired drainage patterns during the subsequent flow direction assignment processing. This will finally result in the drainage flowing at its outside edge with higher elevations towards lower cells at its centre or middle. Furthermore, the flat area must have at least one outlet so that the down-drainage off the flat area is possible (Figure 7, right).

Note: the adjustment of elevation is applied only to DEM cells within flat areas. No DEM elevations outside flat surfaces are altered. The approach still requires the sink-free DEM data as input as the existing approach done in ILWIS.

> REMARK 6. *Land-surface parameters and objects can be derived by either using built in commands or by building and running scripts.*

In ILWIS, the type of geology and soil can be added to hydrological modelling to emphasises local variability. If a geological or a soil map is available the units of this map can be reclassified to represent flow accumulation threshold values (see ILWIS help). Units with coarse grained sandy soils overlaying deeply weathered sandstones can be assigned higher thresholds compared to thin soils occurring over shales (reflecting the lower permeability and little resistance to erosion).

Drainage network extraction — This operation will extract the basic drainage network (boolean raster map). In this case, a stream threshold (minimum number of pixels that should drain into a pixel examined) need to be defined. Optionally, you can also use the flow direction map and an stream threshold map to allow different thresholds in different parts of the study area. The Flow direction map is used to automatically fill possible gaps between found drainage lines. As a result, you will always obtain a continuous drainage network.

Overland Flow Length — This operation calculates for each pixel the overland distance towards the nearest drainage according to the flow paths available in the Flow Direction map. As input, the Drainage network ordering map need to be derived first. This map shows individual streams within a drainage network and assigns a unique ID to each stream. The short segments can be excluded based on a user defined minimum length threshold. Overland flow length is very use-

ful to quantify the proximity to streams and can be compared with the potential drainage density.

In addition, you can also derive the Flow Length to Outlet and the Flow Path Longitudinal Profile. The output for flow length to outlet will be a value map that will contain for each pixel the down flow distance to the outlet, while outlet pixel will have a value of 0. The Flow Path Longitudinal Profile can derived using the function:

```
outputtabe = TblFlowPathLongitudinalProfile
(LongestFlowPathSegmentMap, SegmentID, Distance, AttributeMap)
```

where `LongestFlowPathSegmentMap` is the input segment map that contains the longest flow path of the catchment, `SegmentID` is the segment ID that you want to use generate the longitudinal profile, `Distance` is the threshold value to obtain the output points at a regular distance along segment and `AttributeMap` is a value map used to obtain the *Y* column in the output table. The output is a table that will contain 3 columns: *X* — point ID extracted from the input segment, *Y* — a value from the input attribute map related to the point and *Coord* — Coordinate of the point:

Flow indices — Based on the DEM and the flow accumulation map, three standard indices can be derived: (a) WTI; (b) SPI; and (c) STI. SPI can be used to identify suitable locations for soil conservation measures to reduce the effect of concentrated surface runoff. STI accounts for the effect of topography on erosion.

In addition to extraction of hydrological LSPs/LSOs, a number of functions are given to provide relevant statistical information of the extracted river and catchment network. The Horton plots show the relationship between Strahler order and total number of Strahler order stream segments for a given order, average length per Strahler order and average catchment area per Strahler order, as well as the bifurcation, channel length and stream area ratio's (by means of a least square regression line). The results can be graphically displayed plotting the Strahler order on the *X* axis and the number of drainage channels, stream length and stream area on a log transformed *Y* axis. According to Horton's law the values obtained should[7] plot along a straight line (Chow et al., 1988), which proves that the parameters used for drainage extraction are properly selected. Especially when performing catchment merge operations using Strahler orders, reference to the original Horton plot might be relevant. All extracted catchments can be crossed with e.g. the elevation model and aggregate statistics (mean, minimum, standard deviation, etc.) are computed and appended to the catchment table. Furthermore drainage network and catchment segmentation can be aggregated — merged using different stream orders (e.g. for more generic up scaling purposes) or by user defined drainage outlet locations and the resulting network can be extracted to provide further hydrological model input. Finally, other raster-based layers, e.g. obtained from a soil map or classified satellite image can be crossed with the catchment map and the cross table shows relevant (aggregated) statistical information as well as Horton plots (Chow et al., 1988).

[7] The error in measuring Horton ratio from DEM extracted stream networks is often fairly high. Severe caution is needed when interpreting these values.

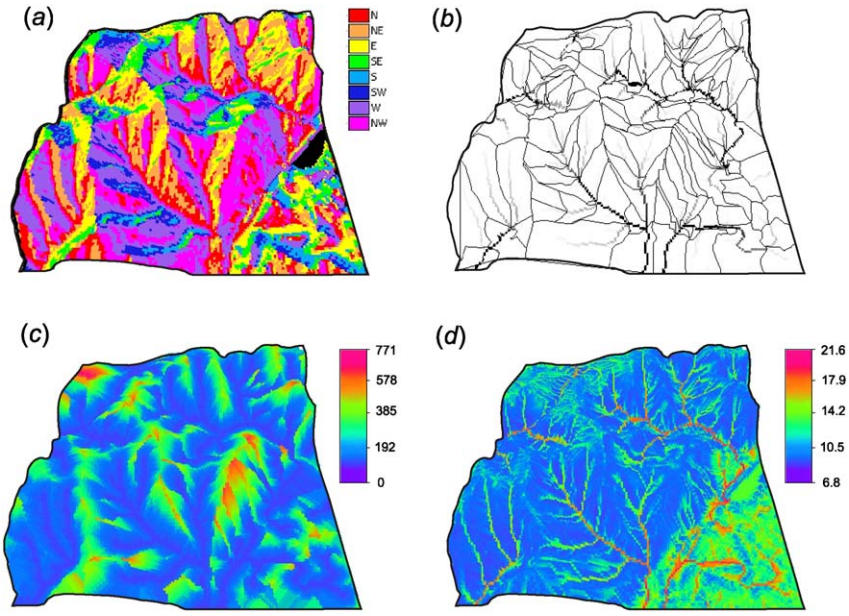

FIGURE 8 Extraction of hydrological parameters and objects using the built-in operations: (a) flow direction, (b) flow accumulation with catchment lines, (c) overland flow length and (d) wetness index. All calculated using the Deterministic-8 algorithm. (See page 723 in Colour Plate Section at the back of the book.)

3.2 Deriving parameters and objects using scripts

In the following section, we will demonstrate how to derive additional number of local morphometric and hydrological parameters using ILWIS scripts. We assume that you have prepared the input DEM using the script described in Section 2.1. Three scripts will be described in more detail:

`LSP_morphometric` can be used to derive local morphometric LSPs: slope in % (SLOPE), aspect (ASPECT), profile curvature (PROFC), planar curvature (PLANC), mean curvature (MEANC), slope-adjusted norhtness (NORTH) and solar insolation for given angles (SOLINS).

`Flow_indices` can be used to derive catchment area and flow indices: topographic wetness index (WTI), stream power index (SPI), sediment transport index (STI) and shape complexity index (SCI), all based on the multiple flow direction algorithm.

`G_landforms` can be used to derive generic landform shapes: channel, ridge, plain (terrace), slope and pit.

`LSP_morphometric` — This script uses the Evens–Young method formulas (see Chapter 6). It differs from similar algorithms in two things. First, the second derivatives (d^2f/dx^2, d^2f/dx^2, $d^2f/dx\,dy$) are smoothed prior to extraction of land-surface parameters in order to produce a more generalized image of land-surface parameters. Second, the local undefined[8] pixels are replaced by iteratively taking the predominant value from the neighbours. The complete script can be seen in Table 1 and the derived land-surface parameters in Figure 9.

TABLE 1 SCRIPT: `LSP_morphometric` — Calculation of local land-surface parameters

REM: Calculation of morphometric land-surface parameters (slope, aspect, curvatures)

```
1  dx.mpr{dom=value.dom;vr=-500.0000:500.0000:0.0001} =
   (%1#[3] + %1#[6] + %1#[9] - %1#[1] - %1#[4] - %1#[7]) /
   (6*pixsize(%1))
2  dy.mpr{dom=value.dom;vr=-500.0000:500.0000:0.0001} =
   (%1#[1] + %1#[2] + %1#[3] - %1#[7] - %1#[8] - %1#[9]) /
   (6*pixsize(%1))
3  //smooth the DEM before deriving second derivatives
4  DEM_s.mpr{dom=value.dom;vr=0.00:5000.00:0.01} =
   %2*(%1#[1] + %1#[2] + %1#[3] + %1#[4] + %1#[6] + %1#[7] +
   %1#[8] + %1#[9])/9 + (1-8*%2/9)*%1#[5]
5  //derive second-order derivates and smooth them to get a more generalised picture
6  d2x_tmp.mpr = (DEM_s#[1] + DEM_s#[3] + DEM_s#[4] +
   DEM_s#[6] + DEM_s#[7] + DEM_s#[9] - 2*(DEM_s#[2] +
   DEM_s#[5] + DEM_s#[8])) / (3*pixsize(DEM_s)^2)
7  d2y_tmp.mpr = (DEM_s#[1] + DEM_s#[2] + DEM_s#[3] +
   DEM_s#[7] + DEM_s#[8] + DEM_s#[9] - 2*(DEM_s#[4] +
   DEM_s#[5] + DEM_s#[6])) / (3*pixsize(DEM_s)^2)
8  dxy_tmp.mpr = (DEM_s#[3] + DEM_s#[7] - DEM_s#[1] -
   DEM_s#[9]) / (4*pixsize(DEM_s)^2)
9  d2x{dom=value.dom;vr =-50.0000:50.0000:0.0001} =
   MapFilter(d2x_tmp, avg3x3)
10 d2y{dom=value.dom;vr=-50.0000:50.0000:0.0001} =
   MapFilter(d2y_tmp, avg3x3)
11 dxy{dom=value.dom;vr=-50.0000:50.0000:0.0001} =
   MapFilter(dxy_tmp, avg3x3)
```

(*continued on next page*)

[8] This usually happens either due to division by zero or because the mapped values are outside a feasible range.

TABLE 1 (*continued*)

REM: Calculation of morphometric land-surface parameters (slope, aspect, curvatures)

```
12 //derive slope, aspect, curvatures (filter them for undefined values using iterations)
13 SLOPE.mpr{dom=value.dom;vr=0.0:5000.0:0.1} =
   100*sqrt(dx^2+dy^2)
14 ASPCT_tmp = raddeg(atan2(dx,dy)+PI)
15 ASPECT{dom=value.dom;vr=0.0:360.0:0.1} =
   MapIter(ASPCT_tmp.mpr, iff(isundef(ASPCT_tmp),
   nbprd(ASPCT_tmp#), ASPCT_tmp))
16 PLANC_tmp = -(dy2*d2x-2*dx*dy*dxy+dx^2*d2y) /
   ((dx^2+dy^2)^1.5)*100
17 PLANC{dom=value.dom;vr=-50.000:50.000:0.001} =
   MapIter(PLANC_tmp.mpr, iff(isundef(PLANC_tmp),
   nbprd(PLANC_tmp#), PLANC_tmp))
18 PROFC_tmp = -(dx^2*d2x-2*dx*dy*dxy+dy^2*d2y) /
   ((dx^2+dy^2)*(1+dx^2+dy^2)^1.5)*100
19 PROFC{dom=value.dom;vr=-50.000:50.000:0.001} =
   MapIter(PROFC_tmp.mpr, iff(isundef(PROFC_tmp),
   nbprd(PROFC_tmp#), PROFC_tmp))
20 MEANC_tmp = -((1+dy^2)*d2x-2*dx*dy*dxy+(1+dx^2)*d2y) /
   (2*(1+dx^2+dy^2)^1.5)*100
21 MEANC{dom=value.dom;vr=-50.000:50.000:0.001} =
   MapIter(MEANC_tmp.mpr, iff(isundef(MEANC_tmp),
   nbprd(MEANC_tmp#), MEANC_tmp))
22 //delete temporary files DEM_s, ????_tmp, d??_tmp
```

`Flow_indices` — This script uses the multiple flow direction algorithm described by Quinn et al. (1991). The theory behind is explained in detail in Chapter 7. The derivation of the catchment area consists of four steps.

(1) Generate the slope-lengths for each diagonal and cardinal direction (8 maps) and their sum:

```
S1 = iff(isundef(DEM#[1]) OR (DEM<DEM#[1]), 0,
     (DEM-DEM#[1])/4)
S2 = iff(isundef(DEM#[2]) OR (DEM<DEM#[2]), 0,
     (DEM-DEM#[2])/2)
...
S9 = iff(isundef(DEM#[9]) OR (DEM<DEM#[9]), 0,
     (DEM-DEM#[9])/4)
SSUM = S1 + S2 + S3 + S4 + S6 + S7 + S8 + S9
```

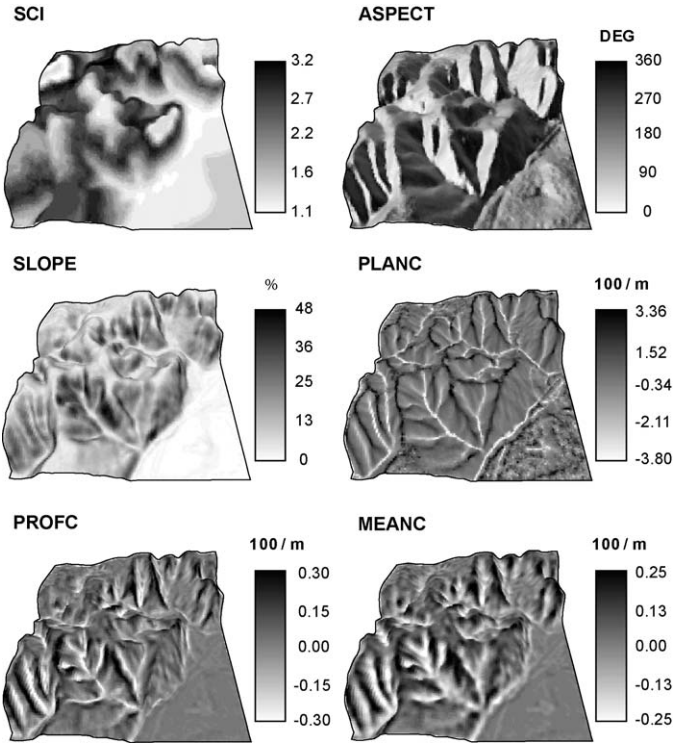

FIGURE 9 SCI (Shape Complexity Index) and other morphometric land-surface parameters derived in ILWIS: ASPECT (0–360°), SLOPE (slope in %), PLANC (plan curvature), PROFC (vertical curvature) and MEANC (mean curvature).

(2) Generate the drainage fraction out of cell for each direction:

```
dA1t = iff(isundef(S1), 0, S1/SSUM)
dA2t = iff(isundef(S2), 0, S2/SSUM)
...
dA9t = iff(isundef(S9), 0, S9/SSUM)
```

(3) Generate drainage fraction into each cell for each direction as a fraction of the contributing cell (Figure 10):

```
dA1 = iff(isundef(dA9t#[1]),0,dA9t#[1])
dA2 = iff(isundef(dA8t#[2]),0,dA8t#[2])
...
dA9 = iff(isundef(dA1t#[9]),0,dA1t#[9])
```

(4) Propagate the total number of contributing cells using n iterations with start map consisting of 1's (Figure 11) and derive the catchment area `CATCH`:

```
start.mpr = iff(isundef(%1), 0, 1)
```

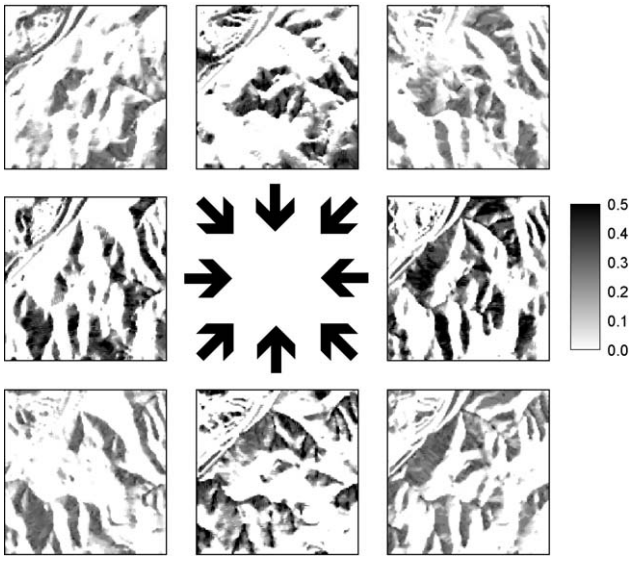

FIGURE 10 Drainage fraction elements in each direction.

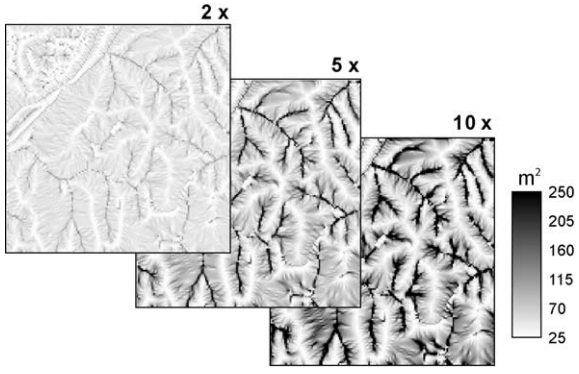

FIGURE 11 Specific catchment area after 2, 5 and 10 iterations.

```
ASUM = MapIter(start.mpr, iff(start<50000, dA1*start#[1]
+ dA2*start# [2] + dA3*start#[3] + dA4*start#[4] + dA6*start#[6]
+ dA7*start#[7] + dA8*start#[8] + dA9*start#[9] + 1, start), %3)
LSUM = pixsize(%1)*(sqrt(2)/4*(iff(dA1>0,1,0)
+ iff(dA3>0,1,0) + iff(dA7>0,1,0) + iff(dA9>0,1,0))
+ 0.5*(iff(dA2>0,1,0) + iff(dA4>0,1,0) + iff(dA6>0,1,0)
+ iff(dA8>0,1,0)))
CATCH_tmp = ASUM*pixarea(DEM)/LSUM
```

The CATCH_tmp map can be iteratively filtered for undefined pixels[9] by taking the predominant value from the surrounding pixels until all zero slopes are replaced:

```
CATCH = MapIterProp(CATCH_tmp.mpr, iff(isundef(CATCH_tmp) and
not(isundef(%2)), nbprd(CATCH_tmp#),CATCH_tmp))
```

This is especially important because in ILWIS the undefined pixels will otherwise propagate. This filtering has the effect of creating pools of high TWI in the plain, which is in general realistic.

> REMARK 7. *At the moment, ILWIS scripts are available to derive dozens of morphometric parameters, flow indices using multiple flow direction algorithm and generic landform shapes.*

Each new iteration will propagate flow by a distance equal to the pixel size or the diagonal pixel size. This should be ideally done until only very few downstream pixels are changed with any new calculation, which can be checked by evaluating a difference map of accumulation after n and after $n + 1$ iterations. In this case we recommend using at least 100 iterations for flow accumulation. Note that the propagation of the drainage fractions can be time consuming.

After the catchment area has been derived, wetness index (TWI), Stream power index (SPI) and Sediment transport index (STI) can be derived using:

```
TWI = ln(CATCH/SLOPE*100)
SPI = CATCH*SLOPE/100
STI = (CATCH/22.13)^0.6*((sin(ATAN(SLOPE/100)))/0.0896)^1.3
```

G_landforms — channel, ridge, plain (terrace), slope and pit (see also Chapter 22) can be derived using the supervised fuzzy *k*-means classification. The input maps needed are the slope in % (SLOPE), planar curvature (PLANC) and anisotropic coefficient of variation (ACV), fuzzy exponent and a table with definition of class centres. In this case, the LF_class.tbt table with the definition of classes looks like this:

	SLOPE	PLANC	SCI	SLOPE_STD	PLANC_STD	ACV_STD
channel	5	-2	0.4	5	0.5	0.25
pit	5	-2	0	5	0.5	0.25
plain	0	0	0.2	5	0.5	0.25
ridge	5	2	0.4	5	0.5	0.25
slope	25	0	0.2	5	0.5	0.25
peak	5	2	0	5	0.5	0.25

These are just approximated class centers and variation around the central values (SLOPE_STD, PLANC_STD and ACV_STD) that will probably need to be adjusted from an area to area (see also Figure 8 in Chapter 22). There can be quite some overlap between the pits and streams (see further Section 5.1 in Chapter 22). Other classes seems to be in general easier to distinguish, although there is obviously overlap between streams-plain and ridges-plain.

[9] Division by zero — locations where LSUM = 0.

The script runs as follows. It will first calculate[10] distances from the central value to the attribute band per each class and standardise them according to the standard deviation:

```
t_d11 = abs(%4-TBLVALUE(%1, "SLOPE", 1)) /
TBLVALUE(%1, "SLOPE_ STD", 1)
t_d12 = abs(%5-TBLVALUE(%1, "PLANC", 1)) /
TBLVALUE(%1, "PLANC_STD", 1)
...
t_d63 = abs(%6-TBLVALUE(%1, "ACV", 6))/TBLVALUE(%1, "ACV_STD", 6)
```

where `%1` is `LF_class.tbt` table and `%4`, `%5`, `%6` are SLOPE, PLANC and ACV maps. Then, it will calculate sum's of distances for each class:

```
sum_dc1 = t_d11^2+t_d12^2+t_d13^2
sum_dc2 = t_d21^2+t_d22^2+t_d23^2
...
sum_dc6 = t_d61^2+t_d62^2+t_d63^2
```

and the fuzzy factors per each class:

```
sum_d1 = (sum_dc1)^(-1/(%2-1))
sum_d2 = (sum_dc2)^(-1/(%2-1))
...
sum_d6 = (sum_dc6)^(-1/(%2-1))
sum_d = sum_d1+sum_d2+sum_d3+sum_d4+sum_d5 +sum_d6
```

where `%2` is the fuzzy exponent. Finally, memberhsips for each class as can be derived as `sum_dc/sum_d`:

```
GLF_Channel{dom=Value, vr=0.000:1.000:0.001} = sum_d1/sum_d
GLF_Ridge{dom=Value, vr=0.000:1.000:0.001} = sum_d4/sum_d
GLF_Slope{dom=Value, vr=0.000:1.000:0.001} = sum_d5/sum_d
GLF_Plain{dom=Value, vr=0.000:1.000:0.001} = sum_d3/sum_d
GLF_Pit{dom=Value, vr=0.000:1.000:0.001} = sum_d2/sum_d
GLF_Peak{dom=Value, vr=0.000:1.000:0.001} = sum_d6/sum_d
```

You might also try to classify an area using some other generic landforms, such as *pool* or "*poolness*", *pass*, *saddle*, etc. These would, of course, require somewhat different clustering of attribute space (see Chapter 9 for more details). The final classification map can be produced by taking the highest membership per cell (Figure 12). In the case of the Baranja Hill dataset, it seems that the most dominant landforms are slopes and plains, while pits occur only in a small portion of the area.

4. SUMMARY POINTS AND FUTURE DIRECTION

ILWIS has many advantages, from which the biggest are the accessibility and richness of GIS operations. For example, next to the elevation data set itself, also information acquired from remote sensing images can be incorporated and up scaling

[10] Note that, in ILWIS, it is possible to run arithmetic operations using raster maps and table values in the same line.

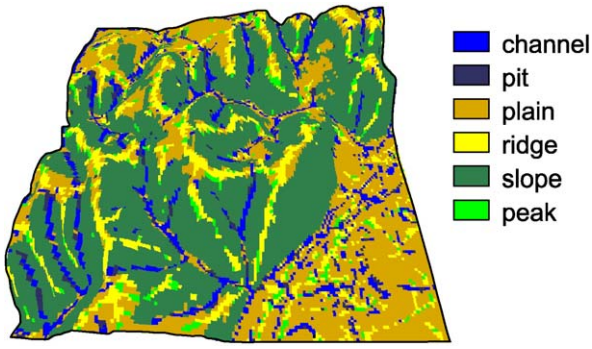

FIGURE 12 Study area classified into the generic landforms. (See page 723 in Colour Plate Section at the back of the book.)

for comparison with data derived from low resolution (meteo) satellites could be facilitated. Relevant features that represent actual topology can also be extracted from satellite images (through screen digitising) and the DEM may be adapted at these locations. It is also possible to improve the assignment of drainage direction over flat surfaces in raster elevation models in order to prevent the occurrence of parallel drainage according to the procedure proposed by Garbrecht and Martz (1997). All this can be achieved because ILWIS already offers a substantial capability for GIS-RS data processing. Also the drainage network and catchment tables generated can be easily linked using common table ID columns and can be exported and incorporated in other packages. The amount of information that can extracted from DEMs is high and can be even extended by building new scripts.

Still, the fact is that the number of ILWIS users is relatively limited to former ITC students and collaborators. There are several probable reasons for this. Number one reason is that the transfer from different packages to ILWIS is still limited. Import/export operations still contains some bugs and can lead to inaccuracies or artefacts in maps. ILWIS needs to import GIS datasets from various popular formats (like ArcInfo ascii, Erdas' `.img` or shape files) to the unpopular ILWIS format which many do not like to do. ILWIS also does not have a website where the users can exchange scripts and user-built modules (compare with ArcGIS, SAGA or GRASS that all have user groups), but only a mailing list.

In addition, the command line is rather user-un-friendly. Unlike in ArcGIS, the user has to already know how are specific functions used and which are input/output parameters. ILWIS will not assist you in running a command directly from the command line or warn you about what is wrong in your command, which usually leads to many tests and trials. Also the neighbourhood operations are fairly limited in ILWIS. For example, unlike ArcInfo DOCELL function, ILWIS is limited to working with 3×3 window environment and further neighbours can not be pin-pointed within an ILWIS scripts.

When displaying multiple raster images, all images need to have the same georeference. Unlike in ArcGIS where the user can overlay literally any GIS layer. On one way, this limitation prevents from creating seamless maps, but does not al-

low exploration of overlap and position of adjacent maps or maps belonging to different grid definitions. Furthermore, the 3D viewer in ILWIS is practically unusable. Draping large raster images is slow and static, therefore not suggested for large datasets. Similarly, ILWIS is not a professional software to prepare final map layouts.

With its limited support and many known and unknown bugs, ILWIS will continue to be rather a scientific than a commercial product. Still, with its rich computational capabilities can be attractive to users with limited funds interested to learn and modify land-surface parameterisation methods. At least now anybody has a chance to obtain the original code and produce an improved version of the package.

IMPORTANT SOURCES

Maathuis, B.H.P., Wang, L., 2006. Digital elevation model based hydro-processing. Geocarto International 21 (1), 21–26.

Unit Geo Software Development, 2001. ILWIS 3.0 Academic User's Guide. International Institute for Geo-Information Science and Earth Observation (ITC), Enschede, 530 pp.

Unit Geo Software Development, 1999. ILWIS 2.1 Applications Guide. International Institute for Geo-Information Science and Earth Observation (ITC), Enschede, 352 pp.

www.itc.nl/ilwis/ — ILWIS home page.

www.ilwis.org — ILWIS users' home page.