PDW Documentation

a program written by

Robert H. Gardner

Appalachian Laboratory
University of Maryland Center for Environmental Science
301 Braddock Road
Frostburg, MD 21532

Introduction

Remote imagery is an essential element for landscape studies. Because remote imagery comes in a variety of formats, each varying in spectral detail and dimensions (i.e., grain or resolution and spatial extent or areal coverage), images from diverse sources must often be combined to form a single complete map. For instance, SPOT imagery (10 m resolution) might be inserted into LANDSAT imagery (30 m) to "fill in" missing portions of a map or increase the temporal resolution of land cover change assessments. This aggregation processes may introduce two types of errors: 1) a shift in abundance, *p*, of each of the map cover type; and 2) a change in geometry of map objects (i.e., habitat patches, rivers, roads, etc.). The extent of these errors depends on the decision rule used to determine how the cover type of an aggregated pixel will be determined from a mixture of cover types encountered in heterogeneous sets of fine-grained pixels. We know that the random selection of a cover type eliminates the first type of error (change in *p*), while all aggregation rules affect the geometry of map objects (shape, edge and size). The program introduced here, PDW, takes advantage of this property of the "random rule" for map aggregation and extends the method to consider either a decrease (aggregate) or increase (divide a single pixel into multiple units) in map resolution. In addition, combining PDW with Qrule (Gardner and Urban in press), which generates multiple iterations of neutral landscapes maps, allows the errors in map reformatting to be assessed (Gardner et al. manuscript).

Method

PDW uses a four-step process for changing map resolution (Figure 1). The first step locates the center point, c_{ij}, of the pixel of the new map (the crosshair in Fig. 1a(1)). The location of c_{ij} is expressed in real dimensions (e.g., decimal fractions of meters) rather than as integer values (e.g., grid coordinates). The new map being formed may be at either a coarser (Fig. 1a) or finer (Fig. 1b) resolution, *r*, than the original. The second step places a set of *n* sampling points on the original map, with the network of points centered at location c_{ij} (Figure 1b). Although the network of points may be of any number, geometry and resolution (*r*, distance between points in the sampling net), Figure 1 illustrates a rectangular network with $n = 9$ points and *r* equal to the resolution of the finer-grained map. If $n = 1$ then the sampling net is composed of a single sampling point located at c_{ij}. This "minimum net" is the most commonly used scheme, especially when filling in missing data. The third step enumerates the types and frequency of land cover sampled and calculates the corresponding cumulative frequency distribution (Fig. 1a(3)), *f*. The final step sets the cover type for the pixel located at c_{ij} by a random selection

from *f*. These four steps are repeated for each pixel of the new map. The process is equivalent for either decreasing (Fig. 1a) or increasing (Fig. 1b) map resolution.

Under most circumstances it will be desirable to weight the points in the sampling net by their distance from the c_{ij}. For instance, the corners of a square grid illustrated in Figure 1 are a greater distance from the center then those in the four cardinal directions. PDW allows for three alternative distance-weighted sampling methods to be employed: no weighting (all points have equal effect); simple inverse distance weighting; or weights based on the inverse squared distance. The use of this sampling scheme is reflected in the program name, PDW: a point-centered, distance weighted, moving window method for changing map resolution.

Generality of Approach

Random selection is an unbiased estimation procedure that preserves *p* during rescaling. There are two innovations incorporated into PDW. The first is the sampling net which form the cumulative frequency distribution, *f*, from the neighborhood of pixels around the central point, c_{ij}. Because the number of points in the net, *n*, the resolution of the net (*r*, distance between points), and the distance-weighting function can all be varied, many alternatives exist for the construction of *f*. The second innovation is the use of real coordinates, allowing an infinite range of changes in resolution. For instance, one may combine data from SPOT and Landsat (resolutions of 10 m and 30 m, respectively) to form a new map with resolution of 15 m. PDW makes this possible.

The generality of the method opens the potential for reformatting of images from the familiar rectangular lattices to any other 2-dimensional geometry – triangles, hexagons, etc. Changes in geometry of the lattice has not, at present, been implemented in PDW, but would be an interesting addition.

Program Execution

PDW was written as a companion tool with Qrule (obtain a copy of Qrule and related documentation at http://www.al.umces.edu/Qrule.htm). Te assumption that Qrule would be available to analyze results allowed PDW to focus on a simple set of tasks: reading in a map(s); transforming according to the parameters specified by the userl and outputting a new map(s) as space delimited ASCII files readable by Qrule for analysis. Map visualization may also be done via Qrule which produces maps readable by ArcView (Figure 2).

Executables for running PDW under either Windows or LINUX are provided in this distribution. Windows and LINUX have somewhat different characteristics, but PDW behaves the same in both cases. Table 1 lists the key variables affecting PDW execution and the following detailed example assumes execution in a Windows environment.

Download the PDW Zip file (**PDWdos.zip**) and place in a directory of your choosing (we'll call this directory **\example**). Uncompress the zip file. The file PDW.exe should now be present in this directory along with a map file **anti.map**. Go to the Windows start menu, click on the "run" icon, type in "cmd" and hit the enter key. Navigate to the directory "\example" and you are ready to begin. Now type **PDW.exe** and answer the questions in a dialog that looks like:

insert dialog

An alternative is to create a text file with all these responses prepackaged. An example file is included called **example.scr**. If you type:

PDW.exe < example.scr > example.chk

PDW will run the above example, reading input from the file **example.scr** and sending output to the file **example.chk**. The contents of **example.scr** are:

```
anti.map
229 157             ! rows and columns
28.5                ! old map resolution
-9999               ! nodata
1                   ! iterations
d                   ! downscale
10                  ! new map resolution
9                   ! net points
0                   ! net resolution
1                   ! weight
-17161817           ! random number seed
```

This script file answers the series of questions asked by PDW causing the original map, anti.map, to be reformatted from 28.5 m to 10 m. The '0' for net resolution causes PDW to set *r* equal to the resolution of the finer map (check **example.chk** to confirm that this is the case.

Execution of PDW creates an map output file readable by Qrule. Every run of PDW creates a file with the same name: **resmap.map**. If you rerun PDW for another case, this file will be overwritten with the new data. So before executing PDW for a second time, save the data by renaming the file or moving it to another directory. Analysis of this map can be performed by Qrule. The following script, **antiq.scr**, makes this easy. Just type:

Qrule.exe < antiq.scr > antiqout.chk

and Qrule will perform the analysis and produce an output map, **arcgrid.map,** which can be read into ArcView for analysis. This file is a space delimited ASCII file with the appropriate header. Remember that spatial analyist must be installed in ArcView to read in this integer ASCII file.

Program limitations

Several program limits are set at the time the program is compiled, including the number of rows and columns of the maps (maxprm), the maximum of land cover types considered (maxhabs), and the maximum size of the sampling net (maxcord). These limits (Table 1) can be changed within the source code (**PDWcode.zip**) by editing module.f90 and changing these values.

PDW reads input maps from a file specified by the user. This file must be an ASCII file of integer values representing different land cover types. The maximum value of land cover types (not the total number) can not exceed the value set by maxhabs.

Negative values within these files (e.g., the no data value of –9999) are treated as map boundaries and are not analyzed by PDW.

Multiple maps may be analyzed by PDW. It is assumed that these maps are separated in the input map file by a blank line between maps. This format is consistent with that used by Qrule, allowing maps be efficiently created by Qrule, transformed by PDW, and reanalyzed by Qrule. This process is the basis of the analysis described by Gardner et al. (*manuscript*).

Linking Qrule and PDW

The following example shows how to use Qrule to create multiple maps, run PDW to increase the resolution of the maps, and then run Qrule to analyze the effects of these changes. The script files for running this example, and the check files (***.chk**) are included in the file **inout.zip.**

Step 1: Run Qrule to create 30 multifractal maps at 30 m resolution.

Qrule.exe < make30.scr > qmake30.chk

Step 2: Run PDW to increase the resolution to 10 m.

PDW.exe < trans30.scr > trans30_10.chk

Step 3: Run Qrule on the transformed maps.

Qrule.exe < newmaps_10.scr > newmaps_10.chk

The combined use of Qrule and PDW allows the effects of map transformations to be systematically examined. Further explanations and examples of this linkage can be found in Gardner et al. (*manuscript*).

Potential developments

The nature of the transformation – either an increase or decrease in resolution – will depend on many factors; the imagery available and the purpose of the analysis are, perhaps, the two most important. The range of choices allows for innovative selection from many possibilities. These multitude of options have yet to be fully explored.

Literature Cited

Gardner, R. H., T. Lookingbill, P. A. Townsend, and J. Farrari. manuscript. A general approach for the analysis of multiple resolution landscape imagery.

Gardner, R. H., and D. L. Urban. in press. Neutral models for testing landscape hypotheses. Landscape Ecology.

Press, W. H., S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. 1992, Numerical Recipes. Cambridge, Cambridge University Press.

Table 1 Description of key program variables setting program limits and controlling program execution.

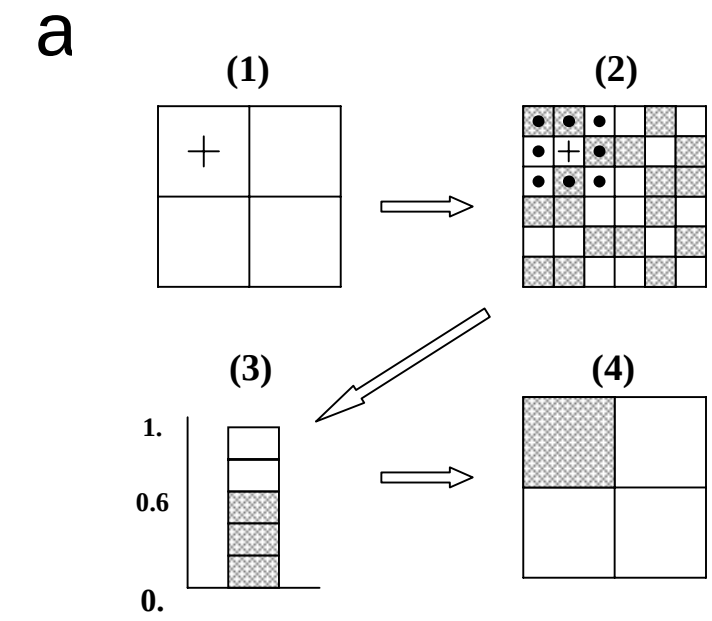
Variable name	Units	Definition	Nominal value	Range	Defined in routine
maxprm		Maximum number of rows and columns (constant)	2048	*	module
maxhabs		Maximum number of cover types in maps (constant)	36	*	module
maxcord		Maximum number of points in window (constant)	48	)	module
nk		Actual number of points in window (variable)	9	1-48	input
MapName		Name of original map (this is the map that will be rescaled) (variable)	character	60 characters	inmap
method		Define if to upscale (a) or downscale (d)	character	a, b	input
OldRes	m	Resolution (grain) of original map	variable	½ map extent	inmap
NewRes	m	Sets the resolution (grain) of rescaled map	variable	½ map extent	input
NetRes	m	Sets *r*, the resolution of the sampling net	'0' sets to resolution of finer map	½ map extent	input
iseed	integer	random number seed	negative value	$< 2^{10}$	driver
Wtexp		Control value affecting weighting scheme	1	0, 1, 2	input
weight		Vector of weighting values with values determined by Wtexp	0 – no weight 1 – linear 2 – exponential		NetDefine
resmap.map		Name of file containing rescaled map	output		input

Table 2

Zip file	Contents	File type/purpose
PDWdos.zip	PDW.exe	DOS executable
	anti.map	map file for 1st example
	example.scr	input file for 1st example
	example.chk	output file for 1st example
	antiq.scr	input file for Qrule analysis
	antiqout.chk	output file for Qrule analysis
PDWLINUX.zip	PDW	LINUX executable
	anti.map	map file for 1st example
	antiq.scr	input file for Qrule analysis
	driver.f90	main
	example.scr	input file for 1st example
	input.f90	input
	Makefile	make file for program compilation (requires a fortran compiler)
	module.f90	program limits
	NetDefine.f90	configuration of sampling net
	netset.f90	locating sampling net
	ran1.f90[1]	random number generation
	setnew.f90	moving sampling net
PDWcode.zip		fortran routines
	driver.f90	main
	input.f90	input
	module.f90	program limits
	NetDefine.f90	configuration of sampling net
	netset.f90	locating sampling net
	ran1.f90[1]	random number generation
	setnew.f90	moving sampling net
PDWinout.zip		Files for running second example
	Qrule	Qrule executable
	make30.scr	input to Qrule, step 1
	qmake30.chk	output file from Qrule
	trans30.scr	input to PDW, step 2
	trans30_10.chk	output file from PDW
	newmaps_10.scr	input to Qrule, step 3
	newmaps_10.chk	output file from Qrule

[1] Random number generation routine from Press et al. (1992).

Figure 1. Illustration of the method PDW uses to place a net of sampling points on an existing map to randomly estimate the cover type of a new map with a coarser (A) or finer (B) resolution. See text for details.



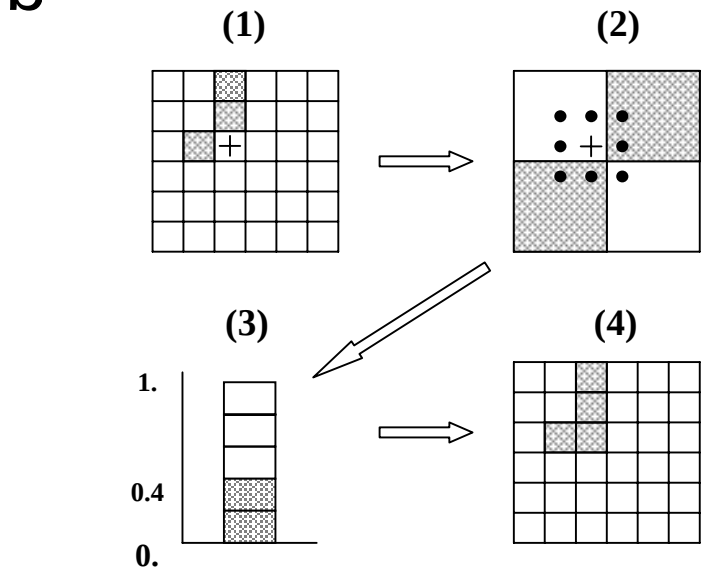


Figure 2. The three panels show a multifractal map generated by Qrule (A) at a resolution of 15m; an aggregated map (B) created with PDW with resolution of 30 m using a sampling net of 9 point, wtexp =1; and (C) a map with higher resolution (5 m) created by PDW with the same sampling net.