Generalization Transformations

- Conversion of data collected at higher resolutions to lower resolution
- Change (reduction) in extent due to scale change (e.g. zoom)
- Less data and less detail
- Simplicity -> clarity
- Can be lossless or lossy
Less detail and fewer features
Generalization:
Line to line transformations

- Problem of "line character"
- Algorithmic resampling i.e. reduce # of points in finite sample
- Algorithmic reconstruction
- Enhancement
Algorithms (Reviewed by McMaster)

- N-th Point retention
- Equidistant Resampling
- Douglas-Peucker
Douglas-Peucker aka Ramer–Douglas–Peucker algorithm, the iterative end-point fit algorithm or the split-and-merge algorithm.
function DouglasPeucker(PointList[], epsilon)
    //Find the point with the maximum distance
    dmax = 0
    index = 0
    for i = 2 to (length(PointList) - 1)
        d = OrthogonalDistance(PointList[i], Line(PointList[1], PointList[end]))
        if d > dmax index = i dmax = d end
    end
    //If max distance is greater than epsilon, recursively simplify
    if dmax >= epsilon
        //Recursive call
        recResults1[] = DouglasPeucker(PointList[1...index], epsilon)
        recResults2[] = DouglasPeucker(PointList[index...end], epsilon)
        // Build the result list
        ResultList[] = {recResults1[1...end-1] recResults2[1...end]}
    else
        ResultList[] = {PointList[1], PointList[end]}
    end
    //Return the result
    return ResultList[]
end
Douglas-Peucker for Michigan Counties

Example (Using Animation) Courtesy of Brad Allen and Waldo Tobler.
Enhancement: adding detail back!

- **Lines**
  - Splines
  - Bezier Curves
  - Polynomial Functions
  - Trigonometric Functions (Fourier-based)
  - Fractals

- **Surfaces**
  - Fractal
  - Fourier
  - Manual
Splines

Tolerance = 0.5

Tolerance = 0.2

Tolerance = 0.0
Bezier curves: Match points and guide points

Continuous derivative at switching point

\[ P_0 \cdot (1 - t)^3 + P_1 \cdot 3 \cdot t \cdot (1 - t)^2 + P_2 \cdot 3 \cdot t^2 \cdot (1 - t) + P_3 \cdot t^3 \]
Polynomial curves
Fourier Series
Fractals: Fractional Geometry

Fig. 3. Enhancement of a 13-point chain, showing the effects of varying SD.
Surfaces: Fractal 2.75
Fourier surfaces

Figure 7

Figure 8
Algorithms for Areas: Overlay

1. Intersections
2. Chain splitting
3. Polygon reassembly
4. Labeling and attribution
Volume-to-Volume

- Common conversion between two major data structures, vector (TIN) and grid.
- Often via points and interpolation
- Problem of VIPs
Vector to Raster and Back Again

- Efficient V->R-> V has eliminated vector-raster debate, BUT is a major source of error
- Major consumer of processing power

**Vector to Raster**
- Easy compared to inverse, a form of resampling
- Grid must relate to coordinates (extent, bounds, resolution, orientation)
- Rasters can be square, rectangular, hexagonal.
- Resample at minimum r/2
- Both structures may be tiled
- Problem: What value goes into the cell?
- Separate arrays for dimensions and binary data?
- Index entries & look up tables
Consider drawing a line on a raster grid where we restrict the allowable slopes of the line to the range $0 \leq m \leq 1$

If we further restrict the line-drawing routine so that it always increments $x$ as it plots, it becomes clear that, having plotted a point at $(x,y)$, the routine has a severely limited range of options as to where it may put the next point on the line:
• It may plot the point $(x+1,y)$, or:
• It may plot the point $(x+1,y+1)$.

So, working in the first positive octant of the plane, line drawing becomes a matter of deciding between two possibilities at each step.
Resampling Problems

North America
200 rows 400 columns

125 rows 250 columns

50 rows 100 columns
Issues in Resampling

- Drop out
- Broken lines
- Fat lines
- Jaggies
- Moire patterns
Algorithm (e.g. rasterize)

- Convert form of vectors (e.g. to slope intercept)
- Thin fat lines
- Compute implicit inclusion (anti-alias)

**Figure 25-11**
Binary skeletonization. The binary image of a fingerprint, (a), contains ridges that are many pixels wide. The skeletonized version, (b), contains ridges only a single pixel wide.
Raster to Vector

- Much harder, more error prone.
- May involve cartographer intervention (e.g. Laserscan)
- Importance of alignment
- Can do points, lines, area
WinTopo
DXF file created
Algorithms

- Skeletonization and Thinning
  - Peeling/Erosion
  - Dilation
  - Medial Axis
- Feature Extraction
- Topological Reconstruction
- User assisted update
Vector lines

Raster scan of lines

Thinned Image: Endpoints located

Conversion to topological vectors

Left to right edges

Top to bottom edges
Data Structure Transformations

- Scale transformations are lossy
- (re)storage produce error
- Algorithmic error, systematic and random
- Types are:
  - scale
  - structural (data structure)
  - dimensional
  - vector-to-raster
JPG using lossy wavelet compression
100% to 1%
Data structure transformations

<table>
<thead>
<tr>
<th>Entity by Entity</th>
<th>Topological</th>
<th>TIN</th>
<th>Grid</th>
</tr>
</thead>
<tbody>
<tr>
<td>Entity by Entity</td>
<td>Scaling Dimensional</td>
<td>Structural</td>
<td>Structural</td>
</tr>
<tr>
<td>Topological</td>
<td>Structural</td>
<td>Scaling Dimensional</td>
<td>Structural</td>
</tr>
<tr>
<td>TIN</td>
<td>Structural</td>
<td>Structural</td>
<td>Scaling Dimensional</td>
</tr>
<tr>
<td>Grid</td>
<td>Raster to vector</td>
<td>Raster to vector</td>
<td>Structural</td>
</tr>
</tbody>
</table>
The Role of Error

- Kate Beard: Source error, use error, process error
- Morrison: Method-produced error
- Error is inherent, can it be predicted, controlled or minimized?

\[ XT = X' \]

\[ X' T^{-1} = X + E \]
Traditional View of Error

- Measurement error is random and Gaussian
- Many errors are also systematic
- Type 1 and Type 2: Omission and Commission
- Omission: Something real on the ground was missed
- Commission: Something NOT on the ground was captured

<table>
<thead>
<tr>
<th></th>
<th>Feature Present</th>
<th>Feature Absent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mapped</td>
<td>True Positive</td>
<td>False positive</td>
</tr>
<tr>
<td>Not mapped</td>
<td>False negative</td>
<td>True negative</td>
</tr>
</tbody>
</table>
Map error

- positional
- attribute
- systematic
- random
- blunder (e.g. 8 vs. 5)
- known
- uncertain
Avoiding error

- Errors can be attributed to poor choice of transformations
- Chain of transformations: Which contributes most error?
- Worst case vs average
- Incompatible sequences of T's (non-invertible)
- "Hidden" Error = use error, not process error
- Blunders and misinterpretations: Design!
- Errors on maps can have major consequences!
Summary

- Line-to-line algorithms resampling vs. modeling
- Also applies to surfaces
- Generalization algorithms examined, can they be inverted?
- Saw examples of data structure transformations
- Resampling and generalization can cause map errors
- Showed issues surrounding raster to vector conversion
- Data structure transformations can be lossy
- Errors on maps can be caused by sequences of transformation
- Errors on maps can be blunders, random or systematic