The Explorations in Digital Cartography Discussion Paper Series is a continuation of the CISRG Discussion Paper Series. This series is edited by:

Mahes Visvalingam
Reader Emeritus
Dept of Computer Science
University of Hull
HULL HU6 7RX, UK

Explorations in Digital Cartography

DISCUSSION PAPER 4

Geometric Data for Testing Implementations of Point Reduction Algorithms: Case study using Mapshaper v 0.2.28 and previous versions

by

Mahes Visvalingam

Explorations in Digital Cartography Discussion Paper Series,
Department of Computer Science, University of Hull

List of publications

ABSTRACT

There are several open source and commercial implementations of the Visvalingam algorithm for line generalisation. The algorithm provides scope for implementation-specific interpretations, with different outcomes. This is inevitable and sometimes necessary and, they do not imply that an implementation is flawed. The only restriction is that the output must not be so inconsistent with the intent of the algorithm that it becomes inappropriate.

The aim of this paper is to place the algorithm within the literature, and demonstrate the value of the teragon-test for evaluating the appropriateness of implementations; Mapshaper v 0.2.28 and earlier versions are used for illustrative purposes. Data pertaining to natural features, such as coastlines, are insufficient for establishing whether deviations in output are significant. The teragon-test produced an unexpected loss of symmetry from both the Visvalingam and Douglas-Peucker options, making the tested versions unsuitable for some applications outside of cartography. This paper describes the causes, and discusses their implications. Mapshaper 0.3.17 passes the teragon test. Other developers and users should check their implementations using contrived geometric data, such as the teragon data provided in this paper, especially when the source code is not available. The teragon-test is also useful for evaluating other point reduction algorithms.

CONTENTS

1. Introduction

Visvalingam (2015a) explained why the specification of Visvalingam's algorithm (Whyatt, 1991; Visvalingam and Whyatt, 1992, 1993) was not overly prescriptive and open to variations in implementation. The primary aim of this paper is to alert implementers and package users of the scope for variations and potential errors in the implementation and use of Visvalingam's algorithm. David Luebke (May 2015, personal communication) pointed out that Schroeder et al (1992) had proposed a similar approach in 3D graphics, which they called decimation, for reducing the vertices in 3D meshes. Their and subsequent research are covered in the surveys by Cignoni et al. (1998), Garland (1997) and Luebke et al. (2002).

Visvalingam's algorithm was popularised by Bloch's (2015) open source Mapshaper program, by Bostock's (2012) demonstrator and Javascript code; and, by republication of the original paper by Visvalingam and Whyatt (1993) in Field and Kent (2014). Visvalingam and Whelan (2014) found that Mapshaper's weighted Visvalingam area metric provided more pleasing simplifications of coastlines but noted some differences in output between Mapshaper's Visvalingam effective area option and Visvalingam's implementation of the algorithm.

Visvalingam (2015b) explored whether the Mapshaper implementation had any unexpected errors over and above expected variations in implementation). It focused on whether implementations conformed to the specification of the algorithm, so as to assess their suitability for intended uses. It was not overly concerned with the properties of the algorithm or with metrics and measures. This paper provides an update based on dialogues with others, especially Matthew Bloch (the author of Mapshaper), and includes an extended background, a description of possible sources of error, and a revised discussion and conclusion. It suggests that similar implementation issues may arise with other geometric algorithms, using a figure in Garland and Zhou (2004) as an example. More recent versions of Mapshaper pass the teragon-test. Other developers and users could check their own implementations of geometric algorithms using the teragon-test, especially when the source code is not available.

2. Background

This section starts by explaining how Visvalingam's algorithm for point reduction differs from its predecessors with respect to its aims and approach (Section 2.1). It then describes the algorithm and its usage (Section 2.2) and reviews some similar algorithms in Computer Graphics and Pattern Recognition (Section 2.3).

2.1 Line generalisation versus line approximation

As Garland and Zhou (2004, p 3) stated "*The study of curve simplification has a much longer history than surface simplification".* In cartography, curve simplification is commonly referred to as line (polyline) simplification. Heckbert and Garland (1997, p 2) defined decimation as a "*fine-to-coarse approach starting with an exact fit, and discarding details to create less and less accurate approximations*". Schroeder et al (1992) had used the term **decimation** to refer to the reduction of triangle meshes by iterative elimination of the vertices until some threshold condition was reached. Visvalingam's algorithm (Visvalingam and Whyatt 1992; 1993) involved the iterative elimination of points (vertices) from polylines.

Since Visvalingam's algorithm was not included in the surveys provided by Heckbert and Garland (1997) and others, this paper explains where it fits. Jenks' (1981) levels of generalisation are useful for understanding the cartographic reasons for vertex reduction. His primary distinction was between output lines which were no longer recognisable after generalisation, and those which were. The latter he classed into a) those which were perceived as essentially the same as the original (referred to here as approximations) and b) those which were seen as distinctly different versions of the original (generalisations). The former were the product of line **approximation** algorithms, which removed superfluous points, errors and some insignificant features. The same algorithms were, and still are, taught and used with much larger tolerances for line simplification and generalisation. However, manual **line generalisation** involves the removal of scale-related features in their entirety; this was difficult to achieve with available approximation algorithms. Visvalingam found by trial-and-error that the method of iterative point removal, driven with the concept of effective area, could even achieve caricatural generalisations in which the original line can be portrayed with very few points. Whyatt (1991) included Visvalingam's algorithm in his evaluation of line generalisation algorithms. The algorithm was formally presented in Visvalingam and Whyatt (1992, 1993).

These two aims of polyline reduction are discussed below to explain the position of the algorithm within Digital Cartography, before reviewing the Visvalingam and similar algorithms.

2.1.1 Line approximation

In the early days, the main concern in digital cartography, pattern recognition and in 3D computer graphics was the ***approximation*** of polylines/polygons and surfaces – not generalisation. Manual digitisation of the lines on maps and algorithmic extraction of boundaries of polygons on scanned images gave rise to superfluous points and errors. Different approaches were explored to remove errors and to represent polylines by a reduced number of vertex points. Koning's (2012) *psimpl* is a lightweight header-only C++ library which includes some of these early algorithms, which were reviewed by McMaster (1987). Ignoring Nth point sampling, the other point reduction algorithms tend to fall into two classes, namely those based on:

- **Sequential search**. These were partly influenced by the restrictions of the technology of the day and include some of the early algorithms which incrementally worked through the line, from start to finish, **rejecting** any points which fell outside a tolerance band (sometimes called a sleeve or envelope) defined on different criteria within a search region. Some sequential point elimination algorithms used other criteria, such as the curvature of the line at a given point (See Weibel, 1997).

  Similar sequential algorithms were reported in Pattern Recognition. Leu and Chen (1988) and Boxer et al (1993) are widely cited by others seeking to improve on the quality and/or speed of the sequential approach. Such sequential algorithms which retain points by a process of elimination are referred to as **merge algorithms** since the algorithm connects the two segments of the original line after a vertex or a segment of the line are eliminated. This class of algorithms includes piecewise linear approximations of curves, which is outside the scope of this paper.

- **Global search**. The algorithm proposed by Ramer (1972) and by Douglas and Peucker (1973), now known as the RDP algorithm, belongs to this category. Unlike the sequential algorithms, which were focused on rejecting points, the RDP algorithm and its variants simplify by recursive or iterative selection of points. The algorithm selects the point with the maximum offset from the line joining the start and end points, and divides the original line into two parts at this point. These two parts are recursively subjected to the same procedure. In Pattern Recognition, the RDP algorithm is classed with **split algorithms**, i.e. those which split the original line into smaller and smaller segments, which may be independently processed.

Maps consist of complex detail and mapping agencies provide access to maps at **multiple levels of detail (LoD)**. For example, maps of Britain can be viewed at different scales (see OS, 2015a). Luebke et al's (2002) classification of LoD differentiates between discrete and continuous LoD. Decimations which reduce data until a terminating condition deliver snapshots at discrete levels of detail. A convenient feature of the RDP algorithm is that the vertices can be tagged with the offset values, which led to their selection, as implemented by Wade (Fortran source in Whyatt and Wade, 1988; see discussion in Visvalingam, 2015a). The weighted vertices could be interactively filtered or animated in continuous LoD and snapshots taken as and when needed. This filter value was usually user-specified, but it can be computed to match the display resolution.

The RDP algorithm is now widely available in open source and commercial GIS packages. Hershberger and Snoeyink (1992) provide complete C code for their convex hull based speeded up version, which has been converted to C++ by Sunday (2012); see also Wikipedia (2015). McMaster (1987, p 108) promoted the RDP algorithm as "mathematically and perceptually superior" to the others he compared. He believed that perceptually it tended to select points which closely matched those selected by humans, citing White (1983); and, that it produced the least areal and vector displacement from the original line, and best preserved its angularity. In their extensive review, Heckbert and Garland (1997) noted that the Douglas-Peucker algorithm is probably the most commonly used curve simplification algorithm and described some variants of the method. Their survey also included reference to terrain simplification in cartography.

Visvalingam and Whyatt (1990) pointed out that the RDP algorithm was designed for polyline/polygon ***approximation*** and that it is insufficient for the type of caricatural ***generalisation*** undertaken by cartographers. Line generalisation involves the omission of less important features, deliberately causing the generalised line to deviate from the original. So, McMaster's measures of goodness of fit were only relevant for line approximation and inappropriate for line generalisation.

2.1.2 Line generalisation by repeated elimination of points

Within cartography, Visvalingam's algorithm (see Pseudocode) offers a **global search and merge** algorithm to complement the **global search and split** RDP algorithm. Visvalingam and Whyatt (1992; 1993) demonstrated that it was able to achieve generalisation, and especially caricatural generalisation, as never done before. It is a merge algorithm, in that it focuses on elimination but it involves global, not sequential, searches. It differs fundamentally from algorithms reviewed in Section 2.1.1 in that it was conceived for different reasons – for caricature and not approximation; the latter was widely regarded as solved by the RDP algorithm. The algorithm was designed to displace the line from the original so as to progressively eliminate scale-related features. It undertakes a global search for the least important point and its specification deliberately avoids reference to 'error terms' and similar nomenclature to emphasise that the algorithm was not designed for approximation but for generalisation, i.e. when a line is perceived as a representation of the original even though it departs from the original.

Also, Visvalingam (2015a) explained that the algorithm can be driven by any metric, and that the Effective Area (EA) she first proposed in Visvalingam and Whyatt (1993) is a heuristic measure designed for facilitating research into caricatural and model-based generalisation. Heuristic measures are indicators and are not guaranteed to provide optimal solutions in all situations. Visvalingam and Brown (1999) pointed out that the offset metric is more suitable when Visvalingam's algorithm is used for line approximation. The algorithm is restated below (Section 2.2) and similar algorithms in related fields are briefly reviewed in Section 2.3.

2.2 Visvalingam's algorithm

The Visvalingam algorithm for polyline generalisation is very simple. "*It consists of repeated elimination of the point which is <u>least significant</u> in a given line and treating the remainder as forming the new input line.*" Visvalingam (2015a) described how it can be expressed in different but consistent ways to suit different circumstances and purposes. She noted why some expressions have limitations and explained why she favoured the specification published in Visvalingam and Whyatt (1993); Whyatt (1991) only needed to implement a part of it for comparing the performance of generalisation algorithms using individual lines. The full specification enables the filtering of a set of lines on maps with a single filter threshold. The versatility of the algorithm was also demonstrated by using it to sieve terrain profiles with multiple filters to identify significant convex and concave forms for P-stroke sketching (Visvalingam and Dowson, 1998). The algorithm was designed primarily to facilitate heuristic research on line segmentation, structuring and modelling of complex lines, such as coastlines, which with their sometimes convoluted shapes, remain a research challenge in Digital Cartography (see Visvalingam, 2015a).

```
[1] Let previous = 0.0
[2] Calculate EA for all internal points of the input line
[3] While there are internal points {
[4]   Find the point with the least EA
[5]   if (EA of this point <= previous)  EA = previous
[6]       else previous = EA
[7]   Record the EA of this point and note its rank (adjusted if and when needed)
[8]   Recalculate EA for the two neighbouring points
[9] }
```

Pseudocode: The Visvalingam algorithm

The algorithm itself is not prescriptive but the Pseudocode includes some implementation-specific features. It does not specify how the importance of a point should be measured – this depends on the application. Any metric can be used, but Visvalingam (2015a) explained why the Effective Area (EA) was chosen initially and why EA functions as a heuristic indicator. The EA in line 2 starts off as the triangular areal displacement which would occur if the point was to be dropped; this is the calculated metric. In the original paper, it was noted that the metric could be weighted and this was explored by Zhou and Jones (2004) and Harrower and Bloch (2006). There may be several points with the same minimal value line 4 and the next point to be eliminated should be chosen in array access order (Visvalingam, 2015a).

The value of the metric is changed if the condition in line 5 is true, which usually indicates the presence of a line configuration suggestive of a feature. Visvalingam (2015a) demonstrated how without lines 5 and 6, lines filtered on unaltered values will not correspond to the rank order of the points. This modification of EA is quite important since it can lead to a cascade of points being eliminated with the same EA (or rank if needed) on thin elongated features. Please note that Visvalingam (2015a) favoured the conditional operator (<=) in line 5 over the original (<) operator published in Visvalingam and Whyatt (1993).

Lines 5 to 7 in the Pseudocode are implementation specific and are not an integral part of the basic generalisation algorithm (see Visvalingam, 2015a). They facilitate the tagging of vertices with the value of the metric which led to their removal in a once only process. User-specified filter tolerances are used to select desired levels of generalisation as in Figure 7 of Visvalingam and Whelan (2014). Lines 5 to 7 also provide scope for filtering a whole map, consisting of several lines with a single tolerance value (Visvalingam and Williamson, 1995). It is also possible to use multiple filter values as demonstrated by Visvalingam and Dowson (1998) when needed. A full analysis of the algorithm identifying opportunities for further research will be provided in a separate paper.

2.3 Other Global Search and Merge Algorithms

Similar algorithms to Visvalingam's were reported in other fields at about the same time. The following references are indicative and sufficient to suggest that the implementation issues discussed in this paper may be of wider concern.

2.3.1 Computer Graphics

The application of global search and merge in higher dimensions than 2D involves a more complex decision space, including a) the choice of entity for removal (for example, vertex or edge); b) the metric to be used to select the next entity for deletion from a candidate list; c) the procedure for merging; and, d) whether there should be constraints, such as topological constraints. Given the immense scope for choice and the prodigious literature in computer graphics, only some directly comparable global search and merge algorithms are included here.

Turk (1992) refers to two contemporary projects which used global search and merge. In 1992, Novins had developed a method for removing vertices in relatively flat portions of a polygonal object. The user specified the number of vertices to be retained. His program queued vertices for removal based on the surface normals of triangles linked to each. Whenever a vertex was removed, the hole was re-triangulated. Turk also referred to the approach adopted by Schroeder et al (1992) for vertex reduction or decimation, which used different criteria (distance from the plane approximating the surface near the vertex) and topological constraints on vertex removal and re-triangulation.

In their catalogue of useful algorithms, Luebke et al (2003, p 122) included Garland and Heckbert (1997) who proposed *quadric error metrics,* "which strikes perhaps the best balance between speed, robustness, simplicity, and fidelity". Garland and Heckbert (1997) proposed simplification by repeatedly contracting a pair of vertices into a single vertex, reconnecting the incident edges and removing degenerate edges and faces. The metric used for selecting the pair for contraction was their plane-based error quadric. As in Schroeder et al (1992) their approach could enforce constraints; contraction could be disallowed or penalised to preserve boundaries and to prevent face inversion. Garland (1998) provides an example of contraction and "planes" in 2D, to illustrate how pair contraction uses new approximating points to replace the vertex pairs. Garland and Zhou (2005) generalised the quadric error metric to propose a new simplification method that produces approximations in any dimension. Figure 7 in Garland and Zhou (2005) includes approximating points that do not lie on the spiral arms. This significant departure from vertex elimination as undertaken by Schroeder et al (1992) and Visvalingam is outside the scope of this paper. However, an implementation-specific issue, which falls within the theme of this paper, will be discussed later.

2.3.2 Pattern Recognition

Pikaz and Dinstein (1995) proposed a global search and merge algorithm, which is very similar to the partial implementation of Visvalingam's algorithm by Wyatt (1991); see Visvalingam (2015a) for more details. The test data they used (the outline of a key) were not as demanding as convoluted coastlines. They advanced several propositions and formal proofs to draw inferences about this algorithm but as Visvalingam and Brown (1999) demonstrated their conclusions relate to their aim of approximation and did not extend to generalisation. Pikaz and Dinstein considered two metrics, namely the offset distance for shape preservation and areal displacement for smoothing and their illustrations are based on the latter. Visvalingam and Williamson (1995) had demonstrated that even though Visvalingam's algorithm could be used for approximation with the offset distance, offset based global search and split methods, such as RDP, are better for weeding (removing superfluous and digitising errors) and for approximation. They also pointed out that Visvalingam's algorithm with EA is not suitable for generalising height contours since it has a tendency to cut curves and this would also apply to Pikaz and Dinstein's test data. The expression of Visvalingam's algorithm was designed to adapt it to application and data requirements.

Pikaz and Dinstein also recalculated the areal displacement of the neighbours when a vertex is eliminated. They noted that the error metric (e.g. area) will not necessarily increase monotonically and that it could give rise to suboptimal results. However, their data prompted them to disregard this as causing minor aberrations, which occurred at the early stages of decimation, mainly along unsmoothed data. Like Whyatt (1991) they undertook the decimation until a terminating condition. With this approach there was no need for Visvalingam's special case (lines 5 to 7) which is essential for compute-once and filter later during interactive exploration of the pre-tagged data. Visvalingam (2015a) illustrated the unacceptable results which would be obtained if the data were filtered on EA without implementing the special case in lines 5 to 7 of the pseudocode.

Visvalingam and Whyatt (1991) discussed variations in the implementation of cartographic algorithms; these included rounding and digitising errors and the presence of equal metric values. Even using limited and less exacting data, Pikaz and Dinstein noted that there may be problems when a number of points have the same minimal error; and, that an arbitrary selection of the elimination order may result in a slightly different polygonal approximation. They considered and then rejected as unnecessary the following modification to obtain a unique solution: "*while the minimal error is less than the threshold, at each iteration eliminate all the points with minimal errors"*. They felt that in practice, the error terms for their data were not significantly different and that it was more important that the approximation had the same geometrical meaning as the input. Visvalingam and Brown (1999) demonstrated that the order of selection can generate shapes which appear distinctly different. However, this may not be an issue with all data, such as those used by Pikaz and Dinstein, nor in all applications, e.g. it may be unimportant in fleeting 3D computer graphics animations.

3. Some implementations of the Visvalingam Algorithm

Release of free topographic data by various government agencies and by corporations, such as Google and OpenStreetMap, has promoted the use of point reduction algorithms in GIS research and applications. Given the huge amounts of high resolution vector data that can be browsed, the subject of cartographic generalisation has become topical. There are a growing number of implementations of both the RDP and Visvalingam's algorithms for line approximation and generalisation – some available as open source software and others have been incorporated within commercial GIS and mapping software. Although the observations made in this paper are also applicable to implementations of the RDP and other algorithms, the following list is limited to representative examples of the use of Visvalingam's algorithm.

- Vivid Solutions (2001) posted a Java implementation on sourceforge.net.for users of its JTS Topology Suite.
- The Zhou and Jones (2004) implementation is used by the Ordnance Survey for generalising coastlines (see Revell, 2011). Ordnance Survey is now using the 1Spatial 1Generalise implementation. Zhou (2014) attempted to segment coastlines using RDP and then use Visvalingam's algorithm to generalise the segments. This is one example of the split-and-merge approach. He provided a link to the download site for his demonstrator program, which includes the Java source.
- Yang et al (2004; 2007) adapted Visvalingam's algorithm with a constraint to remove a vertex, only when its triangle does not contain other vertices. Their subsequent publications on progressive transmission of vector data for web-based applications refer back to these papers for their simplification method.
- McMaster et al (2005) and Schroeder and McMaster (2007) reported on the 5-year NSF-funded project, based on the US Bureau of Census TIGER data, to create a comprehensive National Multiscale Database for the free to use National Historical Geographic Information

System (NHGIS, 2011). After pre-filtering with the RDP algorithm to weed and simplify the raw vector data, multiple databases for target scales were automatically generalised using the Visvalingam algorithm with weighted EA. Other extensions were included for specific purposes (Jonathan Schroeder, June 2015, personal communication; see Schroeder, 2010).

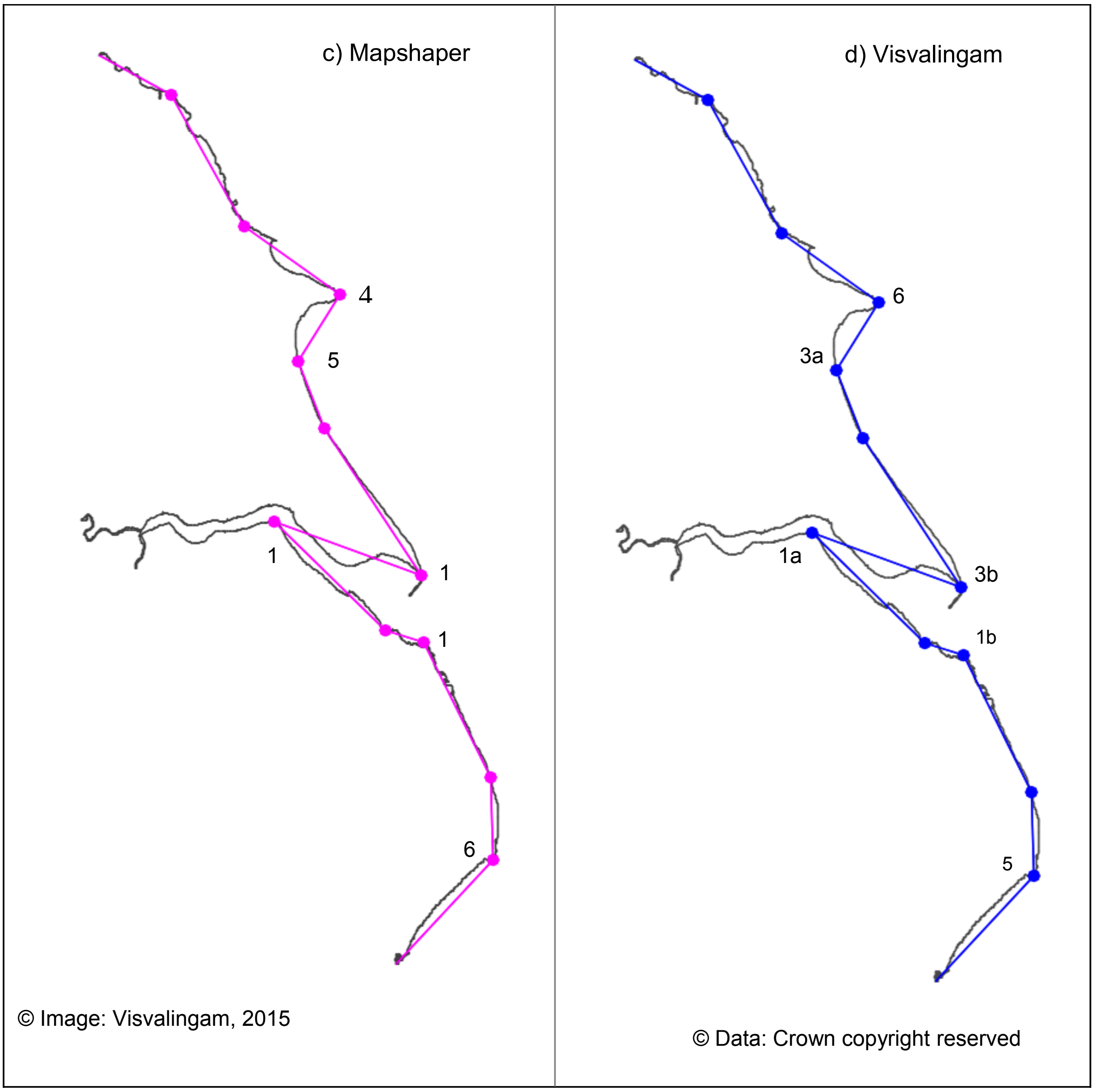
- Ariza-Lopez et al (2005) used Visvalingam's algorithm to tag vertices of roads with EA. They then traced the profile of EA against the distance between vertices and used the RDP algorithm to segment the trace and the corresponding roads. Garcia-Balboa and Ariza-Lopez (2009) suggested parameter values for automating this process. This is an example of a merge and split approach.
- Harrower and Bloch (2006) announced Mapshaper; for the latest version, see Bloch (2015).
- Ahmed et al (2010) used it for progressive vector transmission in mobile GIS.
- Bostock (2012) posted a demonstrator using his Javascript implementation.
- Davies (2012) modified Bostock's implementation to preserve topological relationships between polygons.
- Kaefer (2012) implemented the C++ version within Mapnik.
- Aisch (2012-14) generated compact SVGs using Visvalingam simplification in the free to use version of Kartograph.
- Weifang and Li (2012) adapted Visvalingam's algorithm for progressive transmission of a dendritic river system across the internet.
- Frye (2013) illustrated the use of the algorithm to simplify and compress auto traced coastlines by NASA.
- Steinarsson (2013) adapted the algorithm for down sampling financial time series data.
- Mapbox Studio (2014) is an open source desktop software for designing maps. It uses Mapnik, which includes the Visvalingam algorithm, for rendering maps. Mapbox is not entirely free and has raised substantial funding to compete with Google Maps (see Kolodny, 2013).
- Gaborit (2014) provided a Python implementation of Bostock's code.
- Reimer and Kempf (2014) used a self-intersection-free implementation of Visvalingam's algorithm for caricaturing the outlines of urban settlements, derived from large scale maps, for display at substantially reduced scales.
- Dufilie and Grinstein (2014) used the Visvalingam algorithm for progressive transmission of vector data in web applications.
- ACM (2014) SIGSPATIAL Cup 2014 was on generalising maps with the emphasis on the preservation of topological relationships. The constrained vertex removal used by Chen et al (2014, 3rd prize winners) drew on Visvalingam's algorithm and is similar to the approach of Yang et al (2004).
- IGN (2014) included the algorithm in their generalisation suite.
- OSGeo.org (2015) has a thread on implementation of Visvalingam's algorithm within this Open Source Foundation.
- Oracle Spatial and Graph (12.1) implements the basic algorithm which works on single polylines at a time and also a topology-based constrained simplification (Siva Ravada, personal communication, July 2015).
- PostGIS (2015) closed the thread on their implementation.
- Others are using the algorithm in applications beyond cartography; for example, for simplifying data visualizations for display on mobile devices. (Daniel Cascais, personal communication, 2014) and as already noted in the field of Pattern Recognition (see citations of the work by Pikaz and DInstein (1995).

This project tested just the implementation in Mapshaper for the following reasons. Bloch's open source Mapshaper has been instrumental in promoting the use of Visvalingam's algorithm. It inspired the demonstrator by Bostock (2014), which has in turn inspired Davies (2012) and several others. Mapshaper's weighted area option produced aesthetically pleasing simplifications (Visvalingam and Whelan, 2014). Matthew Bloch (personal communication, 2014)

Figure 1: Comparison of points selected by Bloch (mauve) and Visvalingam (blue); (a) 22 points (0.9%); (b) 13 points (0.5%); order of removal of the last 6 points by c) Mapshaper v0.2.19; (d) Visvalingam

noted our observations on version 0.2.0 and has changed the function for the weighted effective area option in subsequent versions. Also, Mapshaper is sometimes used for exporting generalised shape files for use within some commercial GIS packages, which do not include the Visvalingam algorithm. His source code has been used by others.

The issues identified by Visvalingam (2015b) have now been resolved in Mapshaper – see Section 5 below. This paper provides sufficient background, data and sample output to enable others to test their implementations, using at least the black-box approach adopted here. Any implementation which meets a specified purpose is valid so long as it eliminates the least important point on each iteration in a systematic order (see Visvalingam, 2015a). The following section shows that it is not easy to reach definitive conclusions with coastline data. Geometric patterns, such as fractals, can be more revealing.



4. Observations

At the start of the Visvalingam and Whelan (2014) project, Whelan downloaded the source of Mapshaper v 0.2.0 and checked it using data for the section of a road used by Visvalingam and Williamson (1995). He noticed some differences in the values for EA. Visvalingam found that the discrepancies tended to occur on curved sections of lines, such as at a roundabout and at a filleted road junction. At these places, Mapshaper was picking a different point to that selected by Visvalingam's implementation, especially where these two points had the same EA.

Visvalingam and Whelan (2014) used two stretches of coastlines, namely the SWURCC data and the OS VMD data, as described in their paper. These data sets can be downloaded from https://hydra.hull.ac.uk/resources/hull:9040, which provides information on the sources of these free copyrighted data, maps and the co-ordinates of the coastlines.

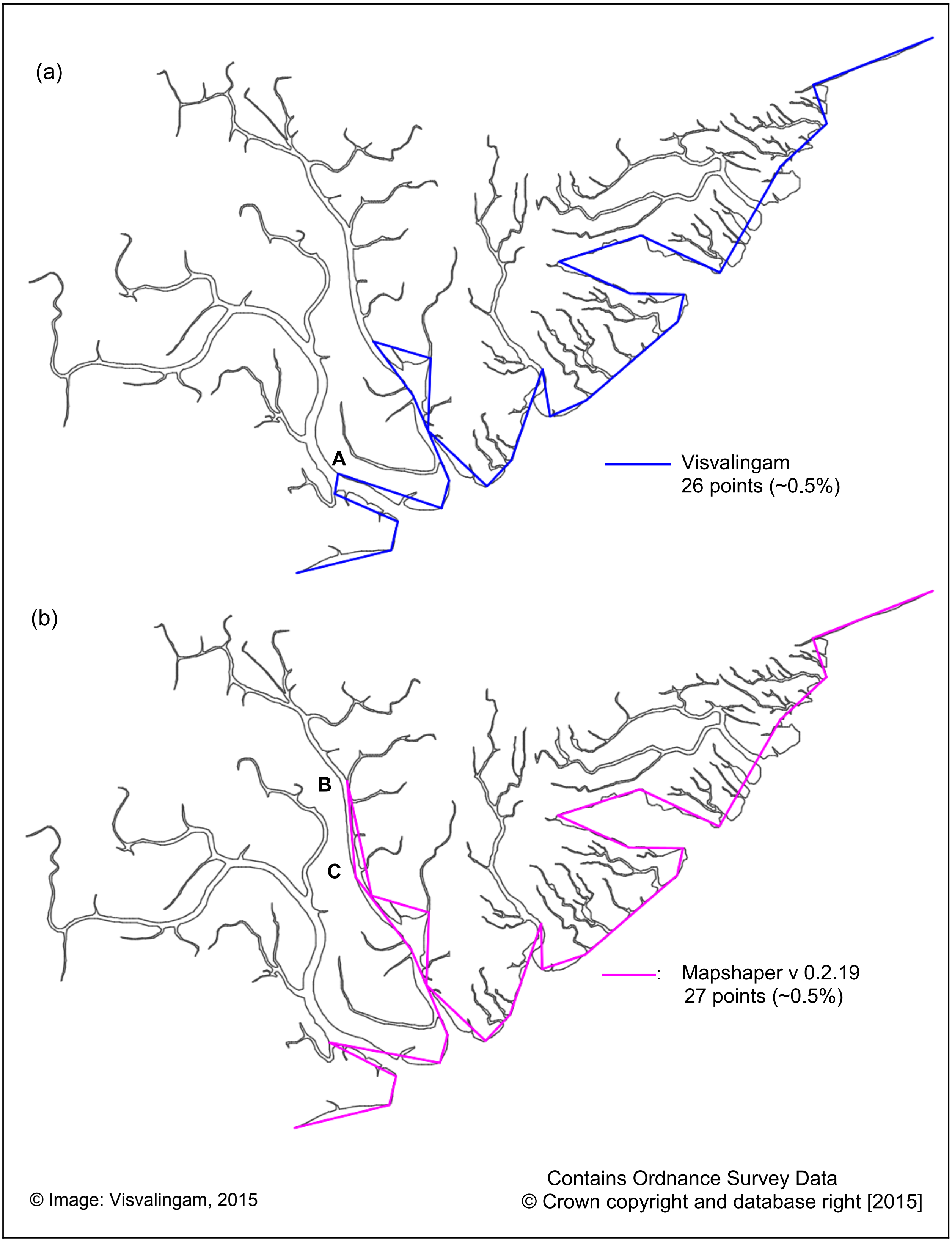


Figure 2 : Filtering by (a) Visvalingam's implementation and (b) Mapshaper

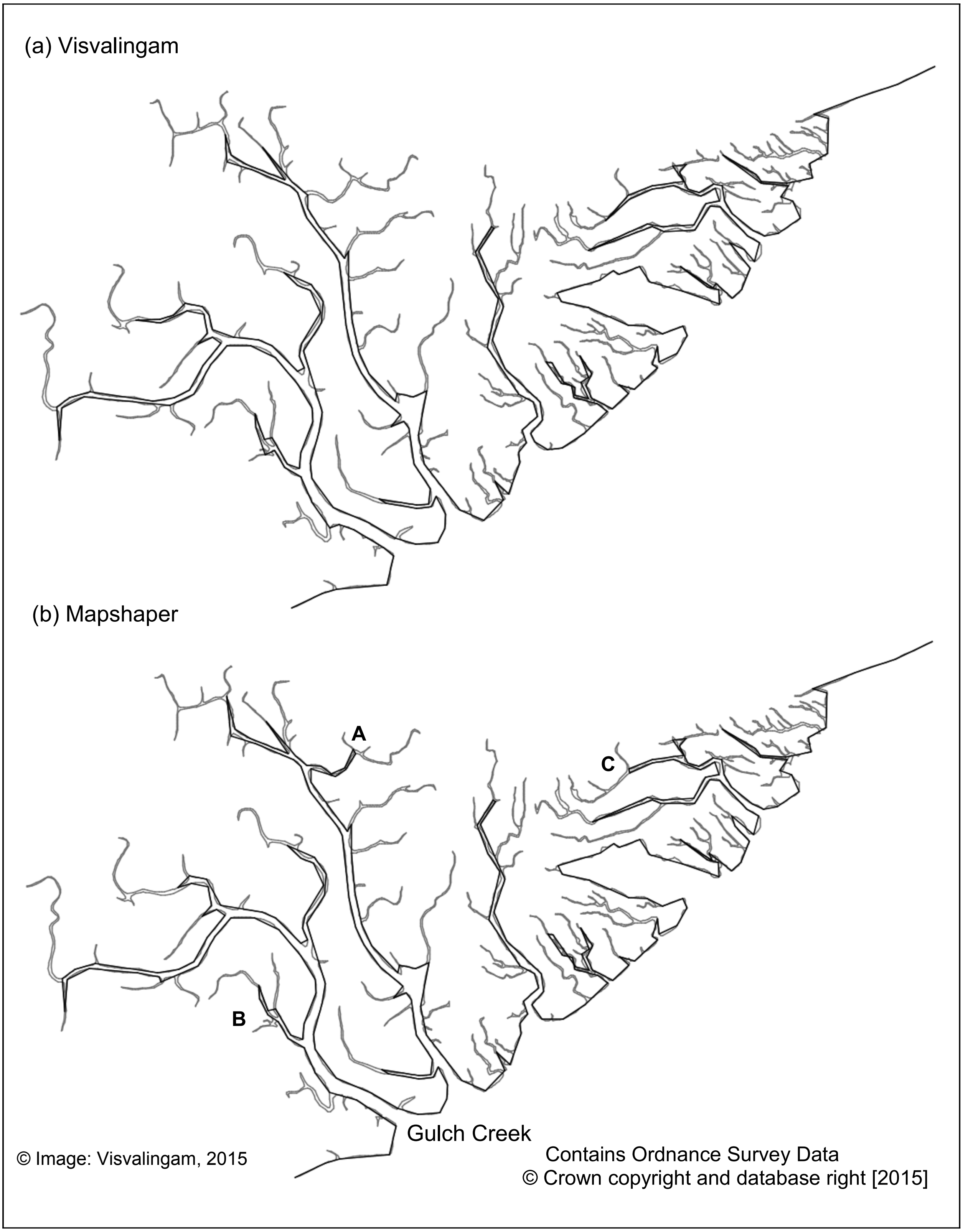


Figure 3 : Five percent of points retained by (a) Visvalingam and (b) by Mapshaper

4.1 1:50000 SWURCC Data

This was the main data set used by Visvalingam and Whelan (2014). Mapshaper produced comparable results to Visvalingam's implementation for this data set.

More recent investigations by the author suggest that the differences were ***partly*** related to the special case (statements 5 and 6 in Pseudocode). When an EA was less than that of the previously eliminated point, Mapshaper did not always pick the point with minimum EA. It picks the first point which fulfilled this condition on some but not all occasions (see explanation in Section 5a). This can have a knock-on effect on the choice of some subsequent points. Different implementations of the RDP algorithm can produce different, but equally valid, results as observed and explained by Visvalingam and Whyatt (1991). So, for reasons of consistency, Visvalingam and Whelan (2014) used Mapshaper v 0.2.0 to compare maps produced with the standard weight of 1 and Bloch's weighted EAs. The subsets of points drawn for a given percentage of points were very similar and often identical. In Figure 1a, there is a difference of just one point in the 0.9% of points retained. Mapshaper picks a point which gives a better shape, while Visvalingam's implementation picks a point which produces the chopped effect discussed in Visvalingam and Whelan (2014). Both implementations produce the same 0.5% subset of 13 points in Figure 1b. Stepwise visualization of the elimination of the points in Figure 1b showed that the two implementations were eliminating points in a different order, especially after encountering the special case. Figures 1c & 1d show the last six points to be eliminated in their order of removal.

4.2 1:25000 OS VectorMap® District data (OS VMD Data)

This data related to an area known as The Scalp in Lincolnshire, which consists of wetlands drained by a complex of meandering creeks (see Ordnance Survey, 2015a). Visvalingam and Whelan (2014) only compared output without and with weighting of EA using maps with 1.4% of points or more. Figure 2 shows the lack of correspondence between Mapshaper and Visvalingam's output when only 0.5% of points were retained. Visvalingam introduces A in Figure 2a before B & C in Figure 2b. At some levels of filtering, Mapshaper produces better results. In Figure 3, most of the retained features are remarkably similar – but there are some notable exceptions. The depiction of tributaries A and B by Mapshaper look more appropriate, even if C looks chopped. Gulch Creek and the trident shape created by the retention of A, make the streams instantly recognizable. This seemed to suggest that Bloch's implementation of the pseudocode could be preferable.

Again, the output for coastlines showed that differences tended to occur a) when there were two or more candidate points with equal EA, and b) when the special case (statement 5 in Pseudocode) was triggered. The impact of equal-valued EAs on Mapshaper was investigated next using fractals.

4.3 Using fractals as test data

It is possible to abstract a range of unexpected patterns from even the level 1 teragon of the rectangular (quadratic) Koch island as demonstrated by Visvalingam and Brown (their Figure 2), not just by using different algorithms, but also by driving Visvalingam's algorithm with different metrics and in different directions. However, of those metrics tested with Visvalingam's algorithm, only EA was able to recover the original square initiator for teragons of orders 1 to 3 of the rectangular Koch curve.

Figure 4 shows the teragon and the generalisations produced by Visvalingam's program using EA. The data for the teragon is provided in the Appendix. As pointed out by Visvalingam and Brown (p 164 -165), Visvalingam's algorithm retains the four-fold symmetry in the teragon and emulates the give-and-take rule used in manual cartography (Maling, 1989). They illustrated how rounding errors in implementation and the use of inappropriate start/end points can lead to a loss of symmetry. Visvalingam and Herbert (1999) used coastlines and the quadratic Koch data to demonstrate that there were problems with the Arc/Info implementation of the Bendsimplify algorithm, which uses Visvalingam's idea of iterative elimination to remove bends instead of individual points.

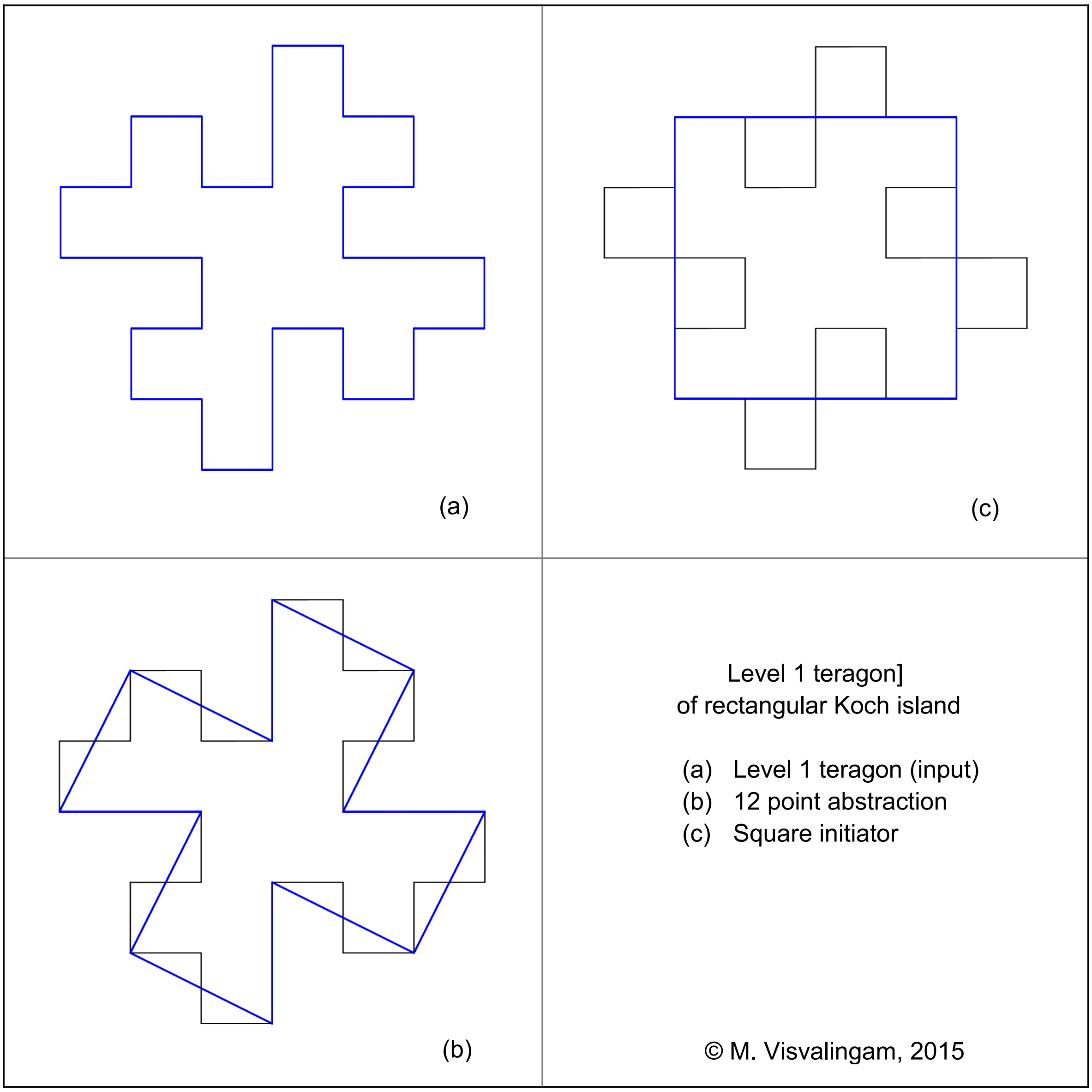


Figure 4: Visvalingam's implementation retains the 4-fold symmetry of the Koch island

Bloch's implementation of Visvalingam's algorithm produced rather unexpected results (see Figure 5). It was unable to retain the symmetry of the Koch island. Visvalingam's implementation only outputs one figure between the teragon and the initiator. Mapshaper outputs several but does not recover the initiator (Figure 5 only shows some of the intervening simplifications). Mapshaper mimics the give-and-take rule in the bottom half of Figure 5a, but produces a different simplification in the top half of the figure; Figure 6a shows the difference. This inconsistency results in unbalanced and unacceptable results on further simplification (5b – c). Visvalingam and Brown (1999, Figure 3a) produced symmetric generalisations of the level 2 Koch island as well. Mapshaper produced very unbalanced output from the same data, and the reasons for this were not immediately obvious. A screen image of Mapshaper's output with 50% of points filtered from the Level 2 Koch island is shown in Figure 6b. Davies (2014) used a pair of quadratic Koch islands to demonstrate his approach to preserving the topology during Visvalingam simplification. His output was also unbalanced. Visvalingam and Brown (1999) noted that like the RDP algorithm, the Visvalingam algorithm is also sensitive to start and end points. The unbalanced output may partly be the result of the insertion of topological nodes to split the polygons into three polylines. However, this does not fully explain the lack of symmetry at even low levels of simplification.

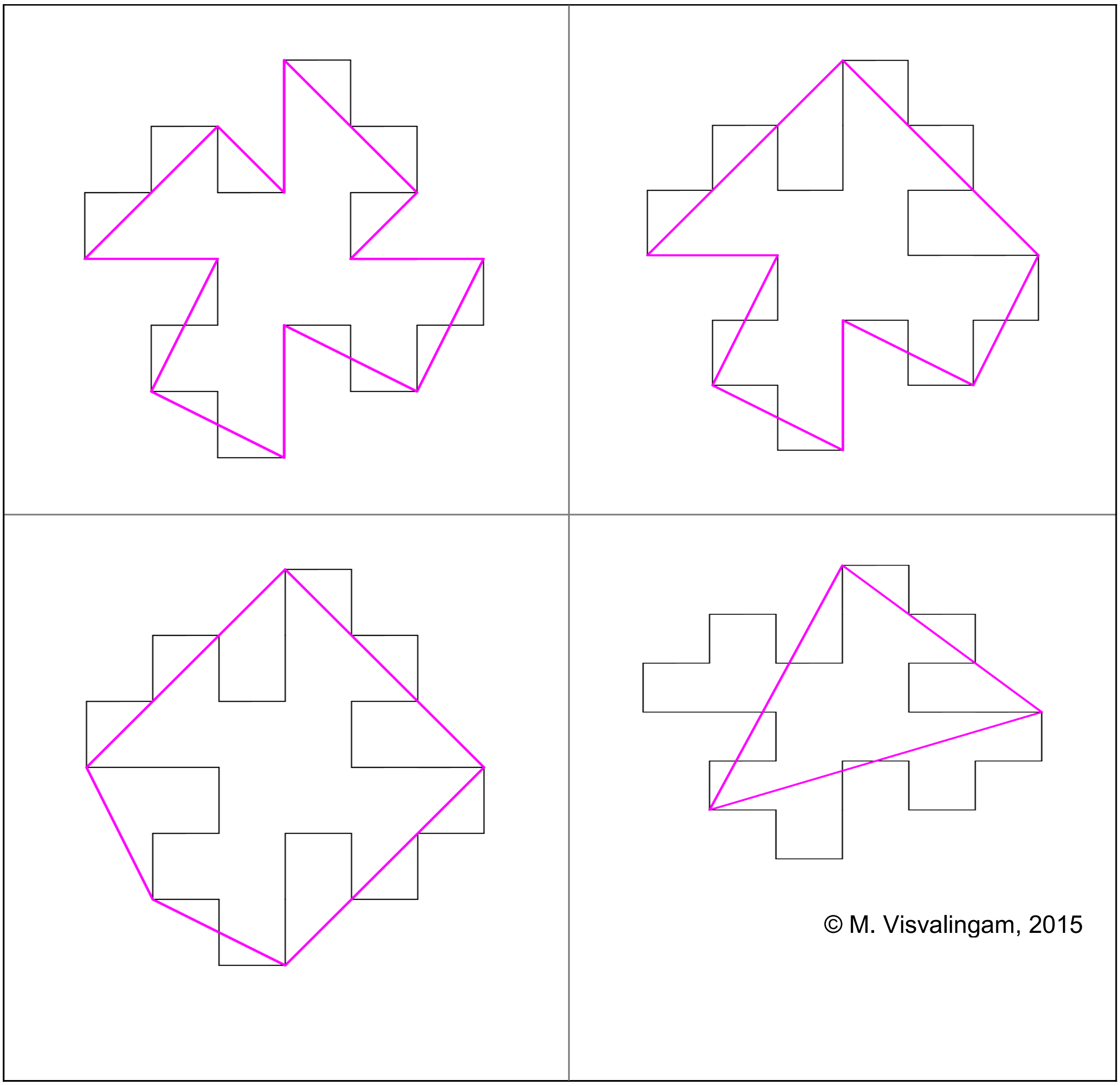


Figure 5 : Sample output from Mapshaper v 0.2.19

The Koch data were run through Mapshaper's ***Douglas-Peucker*** option. Philip Wade's original Fortran programme (listed in Whyatt and Wade, 1988) produced 2 sets of five symmetrical figures (see Figure 2 in Visvalingam and Brown, 1999). Mapshaper produces many more unbalanced figures, including that in Figure 7, which shows varying levels of detail on different wings of the figure.

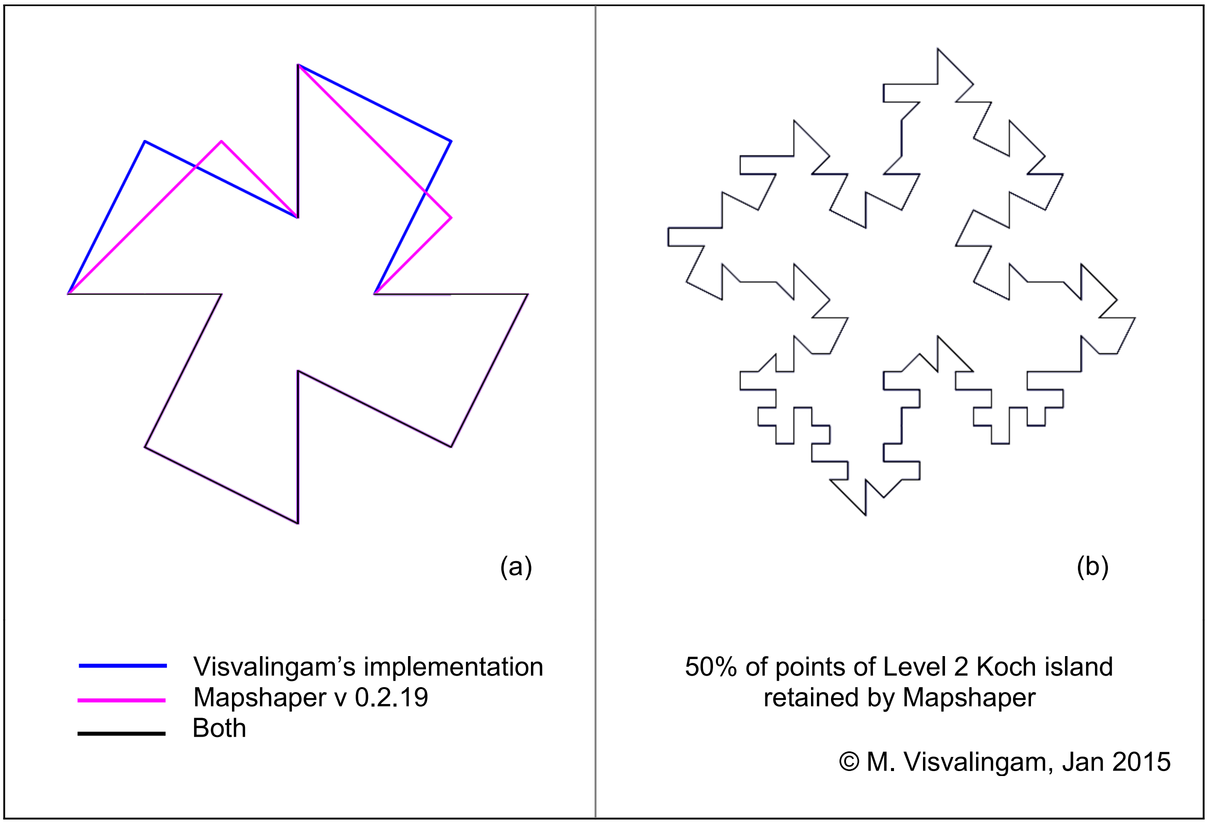


Figure 6 : Comparison of figures abstracted by Visvalingam's implementation and Mapshaper

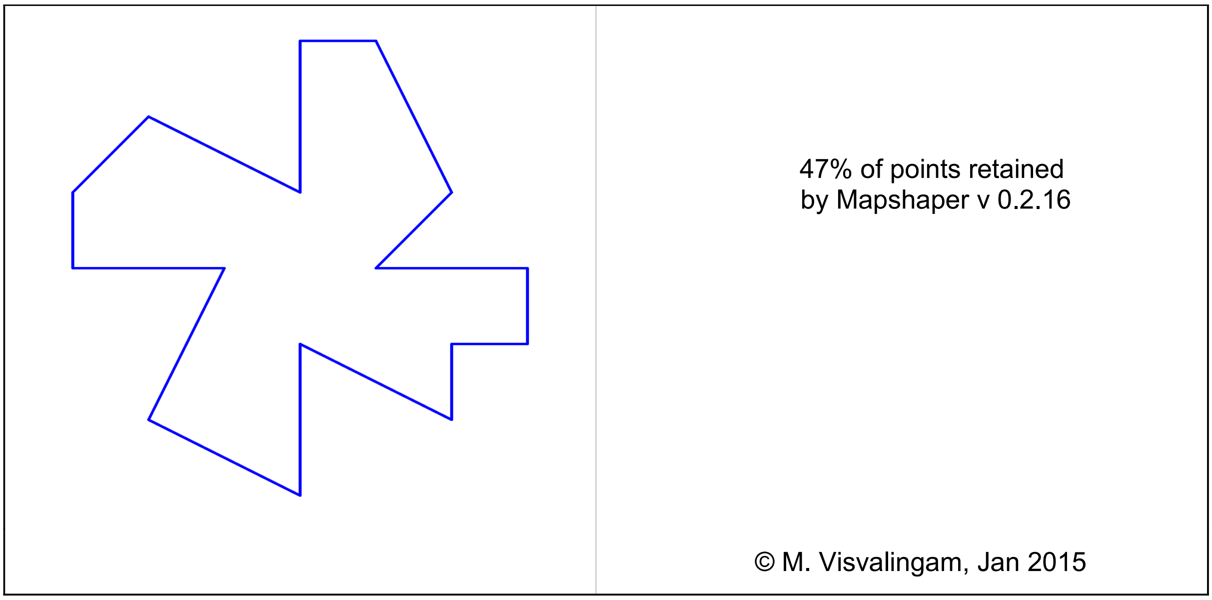


Figure 7 : Mapshaper's output for the RDP algorithm

5. Explanations

The following extracts are from Mathew Bloch's personal communication (7 March 2016) in response to a revised version of Visvalingam (2015b) and to comments on the latter by others:

a. **Treatment of neighbouring points.** "*Until very recently, mapshaper followed a slightly different procedure than the pseudocode included in (Visvalingam & Whyatt 1992). According to mapshaper's original implementation, after a vertex is removed and the effective area of adjacent points recalculated, if an adjacent point's updated EA value is less that that of the removed point, mapshaper would set the EA value associated with a lesser-value adjacent point to be the same as that of the removed point. Not uncommonly, both adjacent points would be set to the same value (the value of the removed point). These points would subsequently be removed in an arbitrary order*". His implementation had been changed in version 0.3.10 (in October 2015) to match the original pseudocode; his updated implementation still did not generate symmetrical output for teragons for the following two reasons.

b. **Selection of the least important point**: Visvalingam (2015a, Section 2.4) used observations in Visvalingam and Brown (1999) to explain why she did not specify precisely how the least important point should be selected. All implementations should output similar, even if not identical, generalisations. "*The only restriction is that the output must not be so inconsistent with the intent of the algorithm that it looks inappropriate for its intended purpose. This can happen if the point is picked from a candidate set with equal-values without due regard to its position along the line. This may not matter when coastlines are only simplified to a modest extent.*" This conforms to Jenk's (1981; see Section 2.1) view that generalisations, being representations, can depart (and vary) from the original as long as they are recognisable.

   (Matthew Bloch, March 2016) noted that "*Mapshaper's Visvalingam implementation uses a min-heap to sort vertices, and the order in which equal-value vertices are removed from the heap is not defined".* He included *"an additional constraint to the heap, such that when the heap contains several vertices of equal weight, the vertex having the lowest array index is removed before the others. With this new constraint, the teragon figure becomes symmetrical when simplified, just like in your implementation."* However, even with this correction, the output remained unbalanced due to the following.

c. **Assumptions about the co-ordinate reference system.** Mapshaper auto-detects whether a dataset has latitude-longitude or projected coordinates. ***"****Mapshaper's current behavior is to assume unprojected geographical coordinates when the bounding box of the data falls within the typical range of decimal degree coordinates* (-180, -90 to 180, 90)*. Mapshaper uses 3D simplification by default when it thinks the dataset is unprojected. …two triangles that have equal-area planar coordinates will most likely no longer have equal areas when their coordinates are interpreted as decimal-degree data. … Mapshaper's behavior works fine for almost any geographical dataset. It only breaks down when simplifying toy datasets designed for testing, which look like lat-long datasets according to Mapshaper's heuristic. With toy datasets, testers need to specify planar simplification explicitly…. A planar default will cause many users unwittingly to apply planar simplification to their unprojected data, and I believe that most of the time it is preferable to use 3D simplification on unprojected data. … The simplification command for Mapshaper's command line interface has a "cartesian" option, which disables auto-detection of geographical coordinates. I just added a checkbox to the simplification settings menu of the web interface".* Currently, in version 0.3.17, this check box only appears when Mapshaper assumes lat-long unprojected co-ordinates.

   Whether the simplification should be applied to unprojected or projected co-ordinates by default is open to discussion (see Section 6.2). Ticking the checkbox for ***planar*** co-ordinates in the simplification user interface box does correct the north-south bias but it did not redress the loss of symmetry in mid-March 2016 for the reason noted in 5b above but it does so as of 27 March 2016 in v 0.3.17. The newly installed settings button, to the left of the slider, enables users to set and change the simplification options at will.

6. Discussion

Of the issues raised in Section 5 above, only 5a is specific to the Visvalingam algorithm. The issues described in 5b and 5c are discussed below since they can affect the behaviour of other geometric algorithms, such as of the RDP algorithm as demonstrated in this paper and the Garland and Zhou (2005) algorithm as noted below.

6.1 The problem of equal areas

Visvalingam and Whelan were not unduly concerned that the output of Mapshaper v 0.2.0 was not identical to theirs. Mapshaper is targeted at simplifying topographic data. Generalised depictions are inevitably subjective in manual cartography and some variation in digital generalisations is equally acceptable. The processes involved in the production of cartographic data introduce a margin of error. Visvalingam and Whyatt (1991) pointed out that cartographic data are inexact and representative. Given that digitising errors are much greater than rounding errors and that data are rounded to the nearest precision for dissemination, they can make pedantic stipulation of how to choose a point from a set of equal-valued candidates somewhat academic. Equal-valued may be fortuitous and the choice of the point for elimination can vary, not just with test data but also with the order in which data are presented as explained by Visvalingam (2015a). What is interesting is that despite the loose specification of the least important point (line 4 of pseudocode), Mapshaper outputs good results for coastlines, suggesting that the method of iterative elimination is fairly robust when applied to topographic data pertaining to natural irregular features.

However, the teragon-test has highlighted that a systematic choice is needed for applications outside of cartography, such as Pattern Recognition and CAD. Unlike coastlines, fractals are by definition self-similar and any generalisation has to retain the symmetry, which defines their self-similarity. Mapshaper was never intended to generalise fractals. However, until it was updated, it would have produced unsuitable results for artificial geographies, such as the man-made coastline and census boundaries of Florida and the USA, illustrated by McMaster *et al* (2005). Nor would it have been suitable for applications outside of cartography, if they require that simplified depictions of engineered components should reflect the symmetry and balance in their original design. Shape distortion will also impede the segmentation of in-line features and parts for model-based generalisation, which has applications in pattern recognition and not just cartography.

Other geometric figures, such as the spiral used by Garland and Zhou (2005, Figure 7) can also enable users to assess whether the implementation is appropriate for them. They noted that the spiral arms could be drastically simplified using Garland's pair contraction algorithm without substantial loss of shape but pointed out the loss of symmetry. The loss of symmetry causes unnecessary congestion in the centre of the spiral. Different cartographers may use different algorithms (even Nth point sampling) to filter the arms and may opt to retain a different number of vertices. But, having decided on the algorithm and the number of vertices to retain, each cartographer would generalise all arms in a consistent way. This makes the author wonder whether the varying number of vertices retained on different arms was due to the implementation and not necessarily a product of Garland's algorithm.

Pikaz and Dinstein (1995, see Section 2.3.2 above) considered eliminating all equal-valued candidates on an iteration but dismissed this idea. This can make Visvalingam's algorithm implementation independent, which may suit some data and applications. However, it can exacerbate the algorithm's inherent tendency to shortcut curves, as noted by Visvalingam and Williamson (1995); this can also make the problem of self-intersection worse.

6.2 Interpretation of the input data co-ordinates

Geometric algorithms, such as point reduction algorithms, are usually conceived and expressed using planar Cartesian x, y co-ordinates. They are often demonstrated using projected data, such as the Ordnance Survey digital topographic map data used in Figures 1- 3 above. The x, y data co-ordinates refer to the Eastings and Northings of the National Grid of Great Britain (Ordnance Survey, 2015b, Section 7).

As noted by Ordnance Survey (2015b, p 14), computations in geodesy are performed on the 3D co-ordinates of spheroids and are only converted to projected map co-ordinates if and when a visual display is needed. GIS tend to perform calculations on map co-ordinates, often by reprojecting data to a common co-ordinate reference system. So, the distinction made by Bloch (see Section 5c) between unprojected and projected data is pertinent. Users should be mindful that the Mapshaper web service may assume unprojected degree data by default and use a simple spherical model of the earth for calculations (Bloch, personal communication, 8 April 2016); this may or may not be appropriate. Also, Mapshaper only uses a rectilinear Cartesian planar projection for display purposes at present (Bloch, personal communication, 8 April 2016). When spherical co-ordinates are assumed for calculation and then displayed using a Cartesian projection, the output can be inappropriate and confusing, as demonstrated by the simplified teragons. The display of input and unprocessed data on the surface of a virtual 3D sphere would alert the user that Mapshaper was assuming lat-long co-ordinates – but this will introduce other issues which are outside the scope of this paper.

Given the complexity of this subject and the variety of spatial referencing systems in use (Ordnance Survey, 2015b; ESRI, 2016), the onus is on users to check any assumptions being made by GIS developers and to request the inclusion of a planar option if required and if none exists. Mapshaper users should explicitly specify the planar option when necessary and if in doubt.

7. Conclusion

There are several independent implementations of the Visvalingam algorithm (Section 3). Most users would find black-box testing with teragons easier than code inspection. It quickly highlighted that there were some problems in the implementation of Mapshaper 0.2.28 and previous versions. Although the source code for Mapshaper is available, this is not the case with all implementations. Visvalingam's algorithm will often form only a small part of a complex GIS system and its output can be affected by other ancillary functions.

Only one source of variability in output was directly related to the implementation of Visvalingam's algorithm. This has been addressed (see Section 5a). A further problem arises from the use of unconstrained sorts to select the least/most important point. Despite the ambiguity in line 4 of the pseudocode, hand working would select a point from a set of candidates in array access order. The use of unconstrained sorts to choose minima and maxima can result in points being chosen in an arbitrary unintended order. This has been corrected in Mapshaper (see Section 5b).

Also, the unbalanced generalisation of fractals was due to the assumption that the input co-ordinates were spherical, when they were planar. This was confusing since National Grid co-ordinates had been treated as planar by Mapshaper. Now that this problem has been highlighted and a planar option included in the Mapshaper 0.3.17 web interface, users have to specify the appropriate referencing system if and when necessary. Other open source software, such as QGIS (2016), provide many more projection options. As the Ordnance Survey (2015b, p 3) guide pointed out, "*Users of coordinates are often unaware that this subject exists, or that they need to know some fundamental geodetic concepts in order to use coordinates properly*." As explained in Section 2, Visvalingam's algorithm was designed for cartographic generalisation and not for line approximation. Generalisation is performed normally on projected map data for human processing. Generalised data, especially caricatures, are unsuitable for calculations. Equally, data pre-filtered by approximation algorithms may not be optimal for deriving caricatures. This is especially so when the data has also been subjected to weighting, smoothing or other modifications during the derivation of multiscale cartographic products.

It must be stressed that Mapshaper 0.3.17 passes the teragon-test when used with the planar option. The onus is on others to check their implementations.

Acknowledgements

The data used in this project are covered by Crown Copyright. The SWURCC data were provided for research purposes and contains public sector information licensed under the Open Government Licence v2.0. The data for The Scalp is Ordnance Survey data © Crown copyright and database right [2015].

This unfunded project by Mahes Visvalingam (a retired, independent research worker - IRW) relied on input from many others. Thanks to Matthew Bloch for freezing Mapshaper v 0.2.0 and for continuing to evolve and maintain his unfunded web service; Chris Brown who generated the co-ordinates for the various levels of teragons using the *lsys* program written by John Leech (published in Prusinkiewicz and Lindenmayer, 1990); John Griffiths (a retired former colleague and fellow IRW) for writing an interactive graphics program, PolylineViewer, for visualising the output of the author's program in camaraderie and friendship; and John Whelan whose contributions to Visvalingam and Whelan (2014) provided the impetus for this research. Thanks also to the implementers who kindly checked and adjusted the text about their work in Section 3.

The author would also like to thank Morgan McGuire and David Luebke for their helpful suggestions and for picking out the landmark papers in Computer Graphics, which are most relevant to the theme of this paper.

References

ACM (2014) ACM SIGSPATIAL Cup 2014 News & FAQ
http://mypages.iit.edu/~xzhang22/GISCUP2014/faq.php [Accessed July 2015]

Aisch, G (2012-14) "What is Kartograph?" http://kartograph.org/ [Accessed Jan 2015]

Ariza-López, F J, Garcia-Balboa, J L and Reinoso-Gordo, J F (2005) "Segmentation of lines by means of Douglas Peucker applied to effective areas of the Visvalingam and Whyatt algorithm". Proceedings XXII International Cartographic Conference (ICC2005). La Coruña (Spain). http://www.cartesia.org/geodoc/icc2005/pdf/poster/TEMA9/FRANCISCO%20J.%20ARIZA%20LOPEZ1.pdf [Accessed June 2015]

Bloch, M (2014) An updated version of the original Mapshaper tool v 0.2.0
http://mapshaper.org/Visvalingam2014 [Accessed 2014]

Bloch, M (2015) Mapshaper v. 0.2.28, http://mapshaper.org/ [Accessed June 2015]

Bostock (2012) Line Simplification – demonstrator http://bost.ocks.org/mike/simplify/
Javascript source: http://bost.ocks.org/mike/simplify/simplify.js
Discussion: https://groups.google.com/forum/#!topic/d3-js/uPOao98T9mg] and
https://news.ycombinator.com/item?id=4055445 [Accessed Jan 2015]

Chen, Y, Wang, Y, Chen, R, Chen, H, Zang, B, Wang, Z (2014) "Greedy Map Generalization by Iterative Point Removal", **ACM SIGSPATIAL 2014**
http://ipads.se.sjtu.edu.cn/_media/publications/sigspatialcup14.pdf [Accessed October 2015]

Cignoni, P, Montani, C, and Scopigno R (1998) “A comparison of mesh simplification algorithms”, **Computers & Graphics 22**, 1, 37–54.

CISRG (2015) **Cartographic Information Systems Research group (CISRG) Discussion Papers Series,** University of Hull; **T**hese preprints of our journal papers and are available at: https://hydra.hull.ac.uk/resources?f%5Bcreator_name_ssim%5D%5B%5D=Visvalingam%2C+Maheswari

Davies, J (2012) “Quadratic Koch Island Simplification”, http://www.jasondavies.com/simplify/koch/ [Accessed Jan 2015]

Douglas, D H and Peucker, T K (1973) “Algorithms for the reduction of the Number of Points Required to Represent a Digitised Line or its Caricature”, **The Canadian cartographer 10 (2) : 112 – 122**

Dufilie, A and Grinstein, G (2014) “Feathered Tiles with Uniform Payload Size for Progressive Transmission of Vector Data”, in (Pfozer, D and Li, K-J, eds.) **Web and Wireless Geographical Information Systems**. Springer Berlin Heidelberg, 2014. 19-35. http://www.cs.uml.edu/~adufilie/feathered-tiles.pdf [Accessed Jan 2015]

ESRI (2016) ArcGIS Runtime SDK for .NET : Spatial references https://developers.arcgis.com/net/desktop/guide/spatial-references.htm [Accessed April 2016]

Field, K. and Kent, A. (2014) **Landmarks in Mapping: 50 Years of The Cartographic Journal,** British Cartographic Society, 416 pp, ISBN: 978 1 909662 38 4

Frye, S (2013) “NASA’s Earth Observing One (EO1) Satellite Advanced Land Imager Observations and Products for Disaster Management”, Presented at the **15th Congres de l’AQT** 26 September 2013 http://www.laqt.org/congres2013/presentations/Frye.pdf [Accessed Jan 2015]

Gaborit, M (2014) visvalingam 0.3.0, https://pypi.python.org/pypi/visvalingam/0.3.0 The source is at: https://github.com/Matael/Visvalingam-Wyatt [Accessed Jan 2015]

Garcia-Balboa, J L and Ariza-López, F J (2009) "Sinuosity pattern recognition of road features for segmentation purposes in cartographic generalization", **Pattern Recognition 42 (9)** 2150-2159.

Garland, M (1988) “Fast and effective polygonal surface simplification” http://mgarland.org/files/talks/QSlim-Talk.ppt [Accessed Jun 2015]

Garland, M and Heckbert, P S (1997) “Surface Simplification Using Quadric Error Metrics” In **Proceedings of SIGGRAPH 97**, ACM SIGGRAPH, 209–216. http://mgarland.org/papers/quadrics.pdf [Accessed Jun 2015]

Garland, M and Zhou. Y (2005) “Quadric-based Simplification in any Dimension”, **ACM Transactions on Graphics, 24(2)**, 209 – 239. Preprint available as Tech Report UIUCDCS-R-2004-2450. http://mgarland.org/papers/TR-2004-2450.pdf [Accessed Jun 2015]

Harrower, M and Bloch, M (2006) “MapShaper.org: A Map Generalization Web Service”, **IEEE Computer Graphics and Applications 26** (4): 22-27; accessible as Bloch, M and Harrower, M (2006) “MapShaper.org: A Map Generalization Web Service”, **Proc AutoCarto 2006.** www.cartogis.org/docs/proceedings/2006/bloch_harrower.pdf [Accessed Jan 2015]

Heckbert, P and Garland, M (1997) “Survey of Polygonal Surface Simplification Algorithms”. In **SIGGRAPH 97 Course Notes**: **Multiresolution Surface Modeling.** [PDF] http://www.cs.cmu.edu/~./garland/Papers/simp.pdf [Accessed Jun 2015]

Hershberger, J and Snoeyink, J (1992) "Speeding Up the Douglas–Peucker Line-Simplification Algorithm", **Proc 5th Symp on Data Handling**, 134–143 (1992). UBC Tech Report TR-92-07 available at http://www.cs.ubc.ca/cgi-bin/tr/1992/TR-92-07 [Accessed June 2015]

IGN (2014) “Class VisvalingamWhyatt”, Laboratoire COGIT, IGN, France http://oxygene-project.sourceforge.net/apidocs/index.html?fr/ign/cogit/cartagen/genealgorithms/polygon/VisvalingamWhyatt.html [Accessed July 2015]

Jenks, G F (1981) 'Lines, Computers, and Human Frailties', **Annals of the Association of American Geographers 71** (1), 1 - 10.

Jenks, G F (1989) “Geographic Logic in Line Generalisation”, **Cartographica 26 (1)**, 27 – 42.

Kaefer, K (2012) " Simplification of geometries using the Visvalingam-Whyatt algorithm", https://github.com/mapnik/mapnik/pull/1385 . See also http://mapnik.org/ and http://dbsgeo.com/sotm-pdx/ [Accessed May 2015]

Koning, E (2012) “psimpl: generic n-dimensional polyline simplification”, version 7 http://psimpl.sourceforge.net/index.html [Accessed June 2015]

Kolodny, L (2013) Google Maps Open-Source Challenger Mapbox Raises $10M From Foundry Group http://blogs.wsj.com/venturecapital/2013/10/16/google-maps-open-source-challenger-mapbox-raises-10m-from-foundry-group/ [Accessed Jan 2015]

Maling, D H (1989) **Measurements from maps: principles and methods of cartometry**, Pergamon Press, Oxford

Mapbox Studio (2014) Mapbox/carto documentation https://github.com/mapbox/carto/blob/master/docs/latest.md#polygon-simplify-algorithm-keyword [Last accessed Jan 2015]

McMaster, R B (1987), “Automated Line Generalization”, **Cartographica 24** (2), 74 111.

McMaster, R B, Galanda, M, Schroeder, J and Koehnen, R (2005) "The Creation of a National Multiscale Database for the United States Census", **Proceedings of the International Cartographic Conference, A Coruna, Spain** http://www.cartesia.org/geodoc/icc2005/pdf/oral/TEMA9/Session%207/ROBERT%20MCMASTER.pdf [Last accessed Jun 2015]

NHGIS (2011) “The National Historical Geographic Information System” https://www.nhgis.org/ [Last accessed Jun 2015]

Ordnance Survey (2015a) **OS GB maps ©2015: Viewer,** http://www.ordnancesurvey.co.uk/opendata/viewer/index.html [Accessed Jun 2015]

Ordnance Survey (2015b) **A guide to coordinate systems in Great Britain,** http://www.bnhs.co.uk/focuson/grabagridref/html/OSGB.pdf [Accessed Mar 2016]

OSGeo-org (2015) Thread on implementation of Visvalingam's algorithm Geo, http://osgeo-org.1560.x6.nabble.com/Sprinting-td5180250.html#a5180696 within the Open Source Geospatial Foundation http://www.osgeo.org/content/foundation/about.html [Last accessed Jan 2015]

Oracle Spatial and Graph (12.1) SDO_UTIL.SIMPLIFYVW in SDO_UTIL Package (Utility), http://docs.oracle.com/database/121/SPATL/sdo_util.htm#CIHCCFCD [Last accessed July 2015]

Pikaz A and Dinstein, I (1995) "An algorithm for polygonal approximation based on iterative point elimination. **Pattern Recognition Letters 16 (6)** 557-63.

PostGIS (closed 2015) "Visvalingam line generalisation function", https://trac.osgeo.org/postgis/ticket/2227 Last accessed Mar 2016]

Prusinkiewicz, P and Lindenmayer, A (1990) **The Algorithmic Beauty of Plants**, Springer, New York.

QGIS (2016) Working with Projections https://docs.qgis.org/2.2/en/docs/user_manual/working_with_projections/working_with_projections.html [Last accessed April 2016]

Ramer, U (1972) "An Iterative Procedure for the Polygonal Approximation of Planar Curves", **Computer Graphics and Image Processing 1,** 244 – 256.

Reimer A, Kempf, C (2014) " Efficient derivation and caricature of urban settlement boundaries for 1:250k", 17th ICA Workshop on Generalisation and Multiple Representation, Vienna, Austria, 23rd September 2014, http://generalisation.icaci.org/images/files/workshop/workshop2014/genemr2014_submission_12.pdf

Revell, P, Regnauld, N, and Bulbrooke, G (2011) OS VectorMap® District: Automated Generalisation, Text Placement and Conflation in Making Public Data Public, in: **Proceedings of 25th International Cartographic Conference**. Paris, France. 3-8 July 2011. http://icaci.org/documents/ICC_proceedings/ICC2011/Oral%20Presentations%20PDF/D3-Generalisation/CO-358.pdf [Accessed May 2014]

Schroeder, J P, and McMaster, R B (2007)" The creation of a multiscale national historical geographic information system for the United States Census", **Proceedings of the 23rd International Cartographic Conference**. Moscow, Russia, August 4-10, 2007. http://icaci.org/files/documents/ICC_proceedings/ICC2007/documents/doc/THEME%2010/Oral%203/The%20Creation%20of%20a%20Multiscale%20National%20Historical%20Geographic%20.doc [Accessed Jun 2015]

Schroeder, J P (2010) "Extensions to the Visvalingam-Whyatt line simplification algorithm for improved boundary characterizations" http://users.pop.umn.edu/~jps/SchroederJP_ICC_2011_Abstract.doc Accessed Jun 2015]

Schroeder, W J, Zarge, J A and Lorensen. W E (July 1992) "Decimation of triangle meshes", Computer **Graphics (SIGGRAPH '92 Proc.) 26 (2),** 65–70. http://fab.cba.mit.edu/classes/S62.12/docs/Schroeder_decimation.pdf [Accessed Jun 2015]

Steinarsson, S (2013) "Downsampling Time Series for Visual Representation", 30 ECTS thesis submitted in partial fulfilment of an MSc degree in Computer Science, University of Iceland. http://skemman.is/stream/get/1946/15343/37285/3/SS_MSthesis.pdf

Sunday, D (2012) “Polyline decimation” http://geomalgorithms.com/a16-_decimate-1.html [Accessed May 2015]

Turk, G (July 1992) “Re-tiling polygonal surfaces”, In **ComputerGraphics (SIGGRAPH ’92 Proceedings), volume 26**, pp 55 – 64.

Visvalingam, M (1990) “Trends and Concerns in Digital Cartography”, **Computer-Aided Design 22 (3)**, 115 - 130. doi:10.1016/0010-4485(90)90070-S

Visvalingam, M (2015a) “The Visvalingam Algorithm - Metrics, Measures and Heuristics”, **Explorations in Digital Cartography Discussion Paper 2**, University of Hull, 11 pp https://hydra.hull.ac.uk/resources/hull:10596 Revised version to be published in **The Cartographic Journal**.

Visvalingam, M (May 2015b) “Testing Implementations of Visvalingam’s Algorithm for Line Generalisation”, **Explorations in Digital Cartography Discussion Paper 3**, University of Hull, https://hydra.hull.ac.uk/resources/hull:10874

Visvalingam, M and Brown, C I, (1999) “The Deconstruction of Teragons into Decogons”, **Computers & Graphics 23** (1), 155 – 167

Visvalingam, M and Dowson, K, (1998) Algorithms for Sketching Surfaces, **Computers & Graphics 22 (2&3),** 269 – 280. Preprint (1997) at https://hydra.hull.ac.uk/resources/hull:8347

Visvalingam, M and Herbert, S P (1999) "A Computer Science Perspective on the Bendsimplify Algorithm", **Cartography and Geographical Information Systems**, 253 - 270.

Visvalingam, M and Whelan, J C (2014) “Implications of Weighting Metrics for Line Generalisation with Visvalingam’s Algorithm”, **Explorations in Digital Cartography Discussion Paper 1**, University of Hull, 24 pp. https://hydra.hull.ac.uk/resources/hull:10064 Revised version to be published in **The Cartographic Journal** (2016)

Visvalingam, M and Whyatt, J D (1990) "The Douglas-Peucker algorithm for line simplification : re-evaluation through visualization" **Computer Graphics Forum 9** (3), 213 - 228. Preprint (1989) at: https://hydra.hull.ac.uk/resources/hull:8330

Visvalingam, M and Whyatt, J D (1991) “Cartographic Algorithms : problems of implementation and evaluation and the Impact of digitising errors”, **Computer Graphics Forum 10 (3),** 225 – 235. Preprint (1990) at: https://hydra.hull.ac.uk/resources/hull:8336

Visvalingam, M and Whyatt J D (1992) "Line Generalisation by Repeated Elimination of Points**",** Cartographic Information Systems Research Group Discussion Paper **10,** University of Hull**,** 16 pp at: https://hydra.hull.ac.uk/resources/hull:8338

Visvalingam, M and Whyatt J D (1993) "Line Generalisation by Repeated Elimination of Points**", Cartographic J., 30** (1), 46 – 51 https://hydra.hull.ac.uk/resources/hull:11210

Visvalingam, M and Williamson P (1995) "Simplification and Generalisation of Large-Scale Data for Roads : a comparison of two filtering algorithms**", Cartography and Geographical Information Systems 22** (4), 264 – 275 DOI:10.1559/152304095782540249 .

Vivid Solutions (2001) VWSimplifier.java http://sourceforge.net/p/jts-topo-suite/code/HEAD/tree/trunk/jts/java/src/com/vividsolutions/jts/simplify/VWSimplifier.java [Accessed Jan 2015]

Weibel, R (1997) "Generalization of spatial data: Principles and selected algorithms". In **Algorithmic Foundations of GIS**, M. V. Kreveld, J. Nievergelt, T. Roos, and P. Widmayer, Eds. Lecture Notes in Computer Science, vol. 1340. Springer-Verlag, Berlin, 99–152.

Weifang, Y, and Li, J (2012) "A method for rapid transmission of multi-scale vector river data via the Internet", **Geodesy and Geodynamics 3**, Issue 2, 34–41 http://www.sciencedirect.com/science/article/pii/S1674984715301166 [Accessed May 2015]

Whyatt, J D (1991) "Visualisation and Re- evaluation of Line Simplification Algorithms", Unpublished PhD Thesis, University of Hull, 259 pp

Whyatt, J D., and Wade, P R (1988) "The Douglas Peucker line simplification algorithm", **Society of University Cartographers Bulletin 22 (1)** 17-25.

WikipediA (June 2015) "Ramer–Douglas–Peucker algorithm", http://en.wikipedia.org/wiki/Ramer%E2%80%93Douglas%E2%80%93Peucker_algorithm [Accessed May 2015]

Yang, B S, Purves, R S and Weibel, R (2004) "Implementation of Progressive Transmission Algorithms for vector Map Data Web-based Visualisation", **The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, Vol. 34**, Istanbul, Turkey, 12 – 23. http://www.isprs.org/proceedings/XXXV/congress/comm4/papers/310.pdf [Accessed July 2015]

Yang, B S, Purves, R S and Weibel, R (2007) "Efficient Transmission of Vector Data over the Internet", **Int Journal of Geographical Information Science, 21(2**) 215-237. [Accessed July 2015]

Zhou, S (2014) "Partition and conquer: Improving WEA-based coastline generalisation" **GISRUK 2014**, Glasgow, 16-18 April 2014: http://www.researchgate.net/publication/261723727_Partition_and_conquer_Improving_WEA-based_coastline_generalisation [Accessed May 2014].
The original demonstrator and the Java source code is at: http://www.researchgate.net/publication/262487021_WEADemo-original [Accessed Aug 2014]

Zhou, S & Jones, C B (2004) Shape-aware line generalisation with weighted effective area, Presented at: 11th International Symposium on Spatial Data Handling, Leicester, UK, 23 - 25 August 2004. Published in: Fisher, P. ed. **Developments in Spatial Data Handling**. Berlin: Springer, pp. 369-380. http://users.cs.cf.ac.uk/C.B.Jones/Zhou_JonesSDH04.pdf

Appendix : Output from Visvalingam's program

X	Y	EA
0	0	9.00
0	1	0.50
1	1	-0.50
1	2	-2.00
0	2	0.00
-1	2	2.00
-1	3	0.50
0	3	-0.50
0	4	8.00
1	4	0.50
1	3	-0.50
2	3	-2.00
2	4	0.00
2	5	2.00
3	5	0.50
3	4	-0.50
4	4	8.00
4	3	0.50
3	3	-0.50
3	2	-2.00
4	2	0.00
5	2	2.00
5	1	0.50
4	1	-0.50
4	0	8.00
3	0	0.50
3	1	-0.50
2	1	-2.00
2	0	0.00
2	-1	2.00
1	-1	0.50
1	0	-0.50
0	0	9.00

KEY

X & Y Co-ordinates
EA Effective Area