Downloaded from orbit.dtu.dk on: Dec 16, 2017

Technical University of Denmark

Conversion of contours to cartesian grids

Mann, Jakob; Broe, Brian Riget

Publication date:
2007

Document Version
Publisher's PDF, also known as Version of record

Link back to DTU Orbit

Citation (APA):
Mann, J., & Broe, B. R. (2007). Conversion of contours to cartesian grids. Risø National Laboratory. (Denmark. Forskningscenter Risoe. Risoe-R; No. 1564(EN)).

DTU Library
Technical Information Center of Denmark

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal

If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

Risø-R-Report

RISØ

Conversion of contours to cartesian grids

Jakob Mann and Brian Riget Broe

Risø-R-1564(EN)

Risø National Laboratory
Roskilde, Denmark
July 2006

Author: Jakob Mann and Brian Riget Broe
Title: Conversion of contours to cartesian grids
Department: Wind Energy Department

Risø-R-1564(EN)
July 2006

Abstract (max. 2000 char.):

A robust and efficient method of calculating a cartesian grid of heights or roughnesses from contour line maps is developed. The purpose of the grids is to serve as input for atmospheric flow solvers such as WAsP Engineering or EllipSys3D. The method builds on Delaunay triangulation constrained to include all contour segments in the triangulation. It is furthermore refined to avoid spurious flat areas produced by the Delaunay triangulation. Robust ways to extrapolate beyond the convex hull of the map points are provided.

ISSN 0106-2840
ISBN 87-550-3527-2

Contract no.: STVF26-02-0015

Group's own reg. no.: 1105058–1

Sponsorship:

Cover :

Pages: 16
Tables:
References: 8

Information Service Department
Risø National Laboratory
P.O.Box 49
DK-4000 Roskilde
Denmark
Telephone +45 46774004
bibl@risoe.dk
Fax +45 46774013
www.risoe.dk

Risø–R–1564(EN)

Conversion of contours to cartesian grids

Jakob Mann & Brian Riget Broe

Risø National Laboratory, Roskilde, Denmark
July 2006

Abstract A robust and efficient method of calculating a cartesian grid of heights or roughnesses from contour line maps is developed. The purpose of the grids is to serve as input for atmospheric flow solvers such as WAsP Engineering or EllipSys3D. The method builds on Delaunay triangulation constrained to include all contour segments in the triangulation. It is furthermore refined to avoid spurious flat areas produced by the Delaunay triangulation. Robust ways to extrapolate beyond the convex hull of the map points are provided.

ISBN 87-550-3527-2
ISSN 0106–2840

Print: Pitney Bowes Management Services Denmark A/S · 2006

Contents

1 Introduction

We describe a robust method of calculating a rectangular grid of height or roughness values from height or roughness contours to be used in two flow calculation programs.

The first program is WAsP Engineering. The purpose of this program is to estimate various aspects of the wind relevant for loads on a wind turbine or other civil engineering structures. The wind properties the program predict is extreme wind speeds, wind shears and profiles, and turbulence. The input is an elevation and roughness map, the positions and heights of the wind turbines, together with climatological data representative for the region. The flow model is a so-called linearized model, which in this case means that it runs very quickly on a pc. The program is commercially available (`www.waspengineering.dk`) and is described in more detail in Mann, Ott, Jørgensen and Frank (2002).

The fact that WAsP Engineering is based on linearized flow equations imposes some limitations. The program does not model properly recirculation zones which appears in too steep terrain.

The purpose of the STVF project "Computational methods in wind power meteorology" (26–02–0015) is to test the performance of the second program, EllipSys3D, in steep terrain. The program is a Reynolds averaged Navier-Stokes solver (RANS) developed by Risø and the Danish Technical University (Sørensen 1995), but can also be run as a detached eddy simulator, also known as DES or hybrid RANS/LES (Hansen, Mann, Johansen and Sørensen 2006). This program also needs a cartesian grid of heights and aerodynamic roughnesses as input.

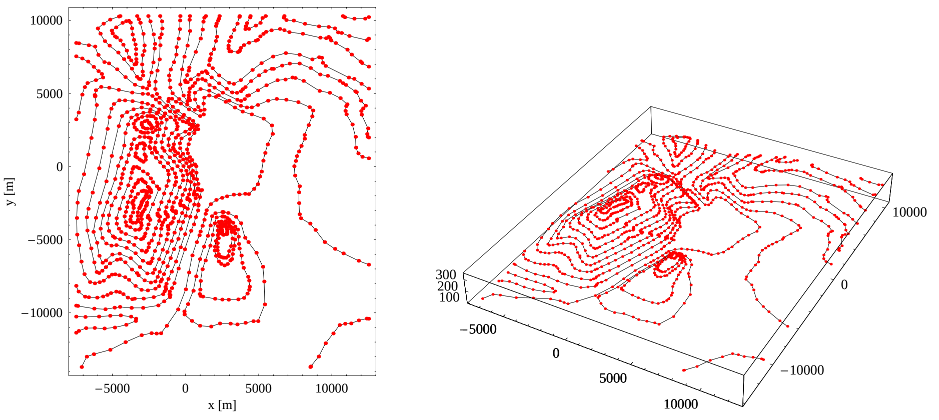

Figure 1. The Waspvale height contours.

The current implementation of the conversion from contours to a grid in WAsP Engineering does not always have a satisfactory behavior. In figure 1 the artificial landscape "Waspvale", which is an example file in WAsP Engineering, is shown. Each height contour is an ordered set of two-dimensional points $(x_1,y_1),(x_2,y_2),...,(x_n,y_n)$ where n can be any positive integer. A height is given for each of these contours. In the figure the points are connected with straight lines and plotted in perspective.

In figure 2 a cartesian grid of heights (200 by 240) is shown to the left. A careful examination shows line-like discontinuities radiating in various directions. Calculating the derivative in the East–West direction $\partial h(x,y)/\partial x$, as shown in the right graphics, enhances these unpleasant features. Also noisy derivatives are seen in the northwestern valleys and along the north shore of the lake in the center of the map.

The gradient of h acts as a source of velocity perturbations in WAsP Engineering, so, at least close to the surface, spurious velocities will occur. In Ellipsys3D even small, spu-

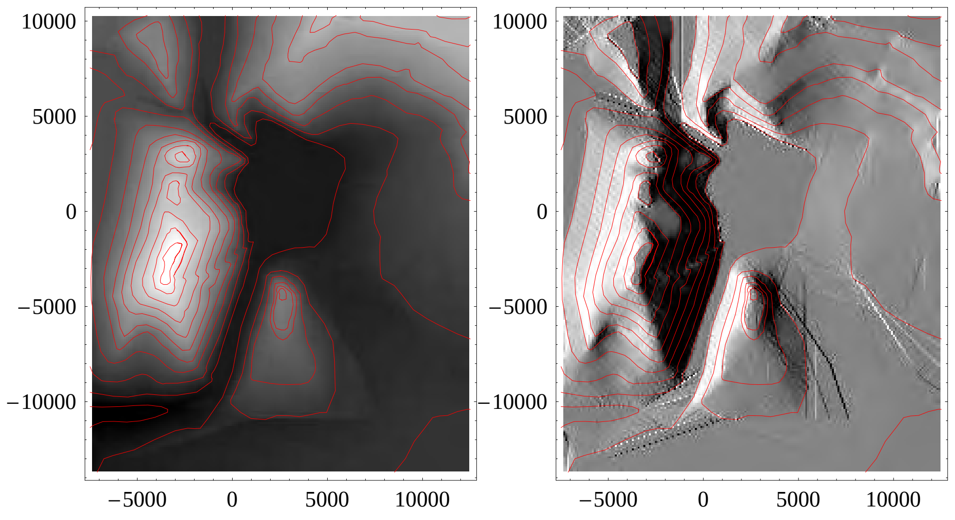

Figure 2. The Waspvale height grid (left) and derivative $\partial h/\partial x$ (right) as made by the original version of WAsP Engineering. The lighter the higher values of either h or $\partial h/\partial x$.

rious terrain discontinuities are unacceptable because they may generate unreal turbulent wakes.

The purpose of this work is to make a better contour line to grid conversion and implement it in WAsP Engineering.

2 Delaunay triangulation

Several properties of the height contour to grid conversion have to be met:

- The interpolation has to be *exact*. This means that if a grid node coincides with a contour point the interpolation gives the height value of that point. Many interpolations introduce some smoothing, but here we consider the input contour lines to be trustworthy, so their height values have to be reproduced exactly.
- *Gradients* should be relatively limited. There should be no allowance for sudden jumps, unless as a consequence of very closely spaced height contours, because this could affect the flow considerably.
- *Extrapolation* has to be implemented. Maps of islands, for example, do not necessarily have contour lines engulfing the requested computational grid.
- The interpolation has to be able to cope *efficiently* with contour line files of many megabytes. The interpolation should not be significantly slower than the original implementation in WAsP Engineering.

The second point is not very precisely formulated and some subjectivity in the choice of algorithm is unavoidable.

2.1 Standard methods

In order to gain a feeling for the problems of interpolation we look at seven "canned" interpolation routines as provided by *Surfer* (Surfer 2002).

In figure 3 seven different Surfer interpolations are shown. We have chosen to show all exact interpolations with standard settings of the parameters, if any. The grid is, as in figure 2, 200×240. Clearly, the methods *inverse distance squared, Modified Shepard*, and

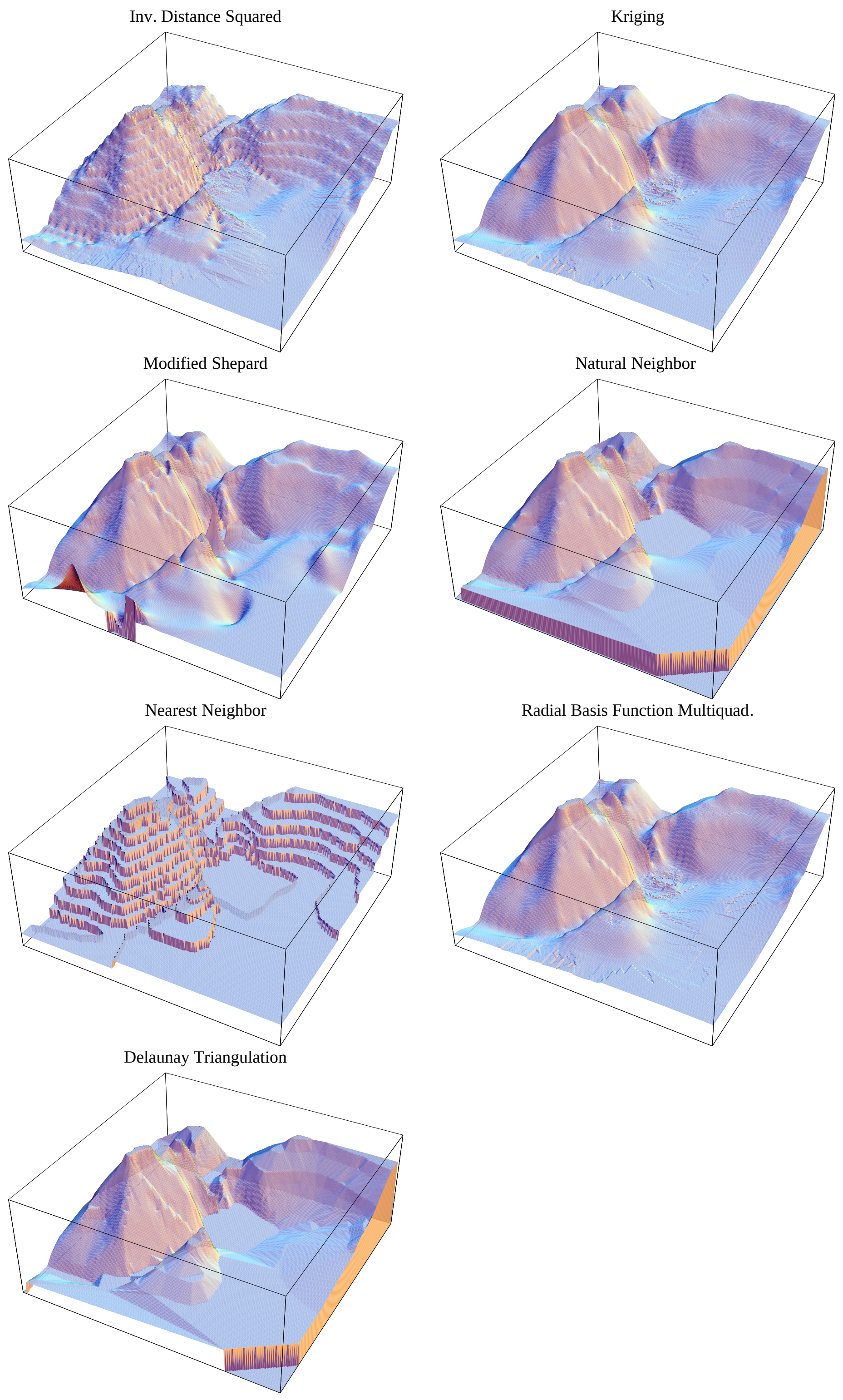

Figure 3. Surfer interpolations.

Nearest Neighbor are all unsatisfactory. *Kriging* is, apart from being prohibitively slow for large maps, nice looking, but introduces small scale irregularities. The same is true for the *radial basis function* method, where irregularities and undulations are introduced at the lake in the middle of the domain. This leaves us with the methods *Natural Neighbor* and *Delaunay triangulation with linear interpolation*.

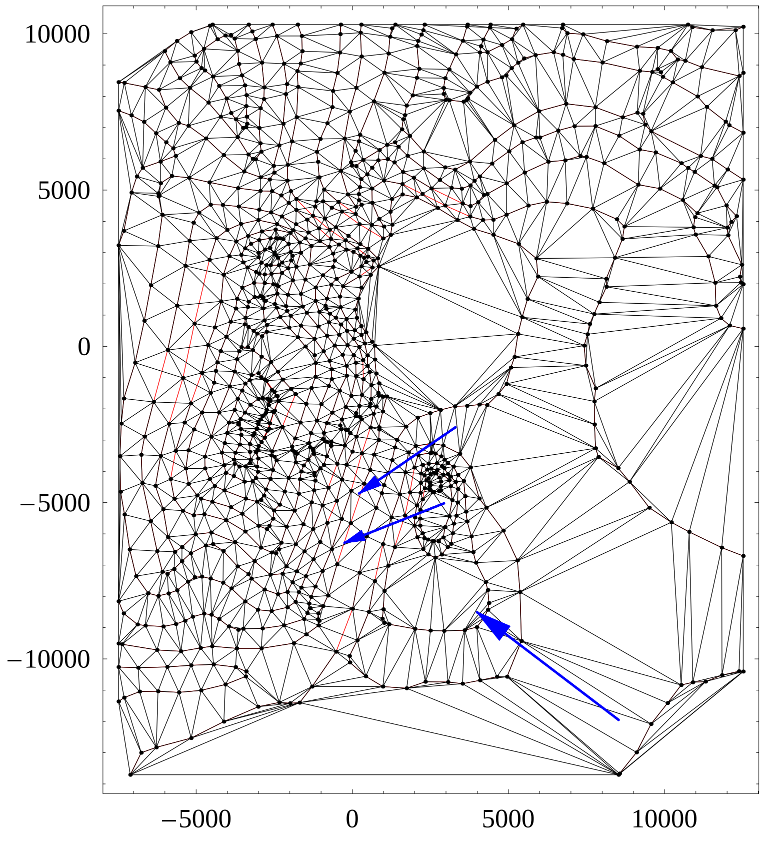

Figure 4. Delaunay triangulation of the points in the Waspvale map. The contours are first plotted in red and the triangulation is shown in black on top. Blue arrows point to problematic areas of the triangulation, which is also shown in figure 5.

The two last methods, as well as the nearest neighbor method, use triangulation. Triangulation of a set of points, in our case the points where heights are given, means connecting as many pairs of points as possible with straight, non-crossing lines. This can be done in many ways, and the result is a collection of contiguous triangles that covers the set of points. A particular beautiful triangulation is due to Delaunay (O'Rourke 1998). The Delaunay triangulation makes the triangles most "round" in the sense that any other triangulation would have a distribution of angles shifted toward the sharper ones. A Delaunay triangulation based on the points in the Waspvale map is shown in figure 4. The piecewise linear, closed curve engulfing all triangles is called the *convex hull*. In Surfer's Delaunay interpolation algorithm the heights on each triangle are interpolated linearly.

The natural neighbor method (Watson 1985) uses the Voronoi diagram, also called the nearest neighbor graph. It is intimately related to the Delaunay triangulation (O'Rourke 1998). It appears a bit smoother than the Delaunay triangulation. We do not pursue this powerful algorithm further because it is not quite clear how it should be modified for non-Delaunay triangulations, which we will use later.

All the methods do not apply outside of the convex hull, so ways to extrapolate must also be devised.

2.2 Problems with Delaunay

Close inspection of the figure 4 and 5 reveals some problems with the triangulation:

1. The smaller arrow in figure 5 points to spurious ridges. Inspecting figure 4 the problem seems to be that the height contour itself is not a part of the triangulation. As it stands, the information on how the individual height points are connected is in fact

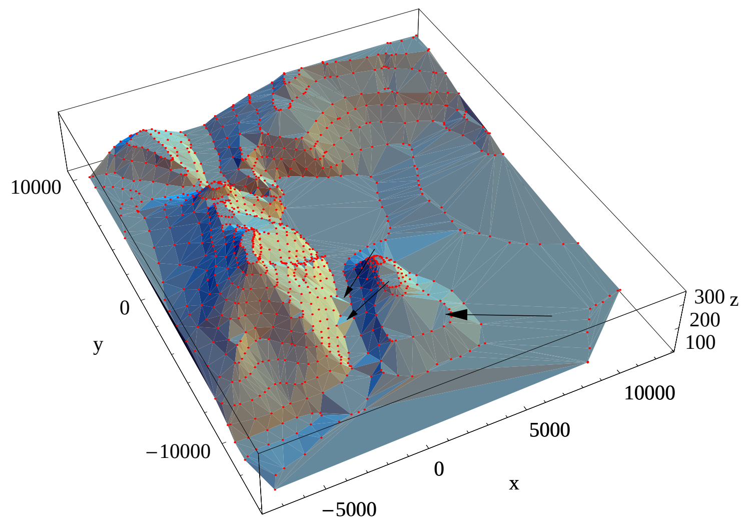

Figure 5. Same as figure 4 but in perspective view. All numbers are in meters.

not used at all.

2. The larger arrow points to an area which appears flat in figure 5, which, judging from the contour lines in figure 4, should not have been flat.

3 Refinements and implementation

In this section we improve the triangulation and describe some implementation and performance issues.

3.1 Constrained Delaunay triangulation

The first problem described in section 2.2 is that the contour lines are closer together than the spacing between the points on each contour. This makes the Delaunay triangulation "jump over" contour lines. The solution to this problem is *constrained Delaunay triangulation*. This is implemented in a very efficient and robust C code by Shewchuk (1996). Constrained Delaunay triangulation is an ordinary Delaunay triangulation followed by insertion of the missing height contour segment. Each segment is inserted by removing the triangles it overlaps, and then retriangulating each region on either side of the segment (Shewchuk 2002). In this way no new points will be introduced. The resulting triangulation will be Delaunay, but the triangles are as "round" as possible given the constraint.

3.2 Rearrangement of triangles

The Askervein Hill, shown in figure 6, has been used for experiments to study atmospheric flow over topography (Taylor and Teunissen 1983, Taylor and Teunissen 1985, Walmsley and Taylor 1996). Any atmospheric flow model will use the Askervein experiment for validation, so it is natural to use it here as well.

In figure 7 a part of the triangulation is shown. The height contours are in red and appear as edges in the triangulation because it is constrained to do so. All other edges are shown in blue. Some triangles are shown colored, for example 3590 and 3589 on the left plot. These are "problematic" in the sense that they touch two different height contours and connect two points on the same contour that are not neighbors, implying

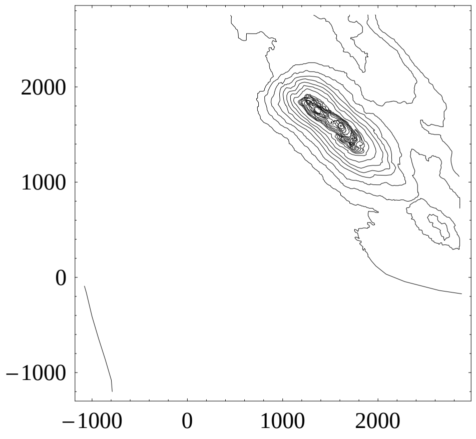

Figure 6. Height contours of the Askervein Hill

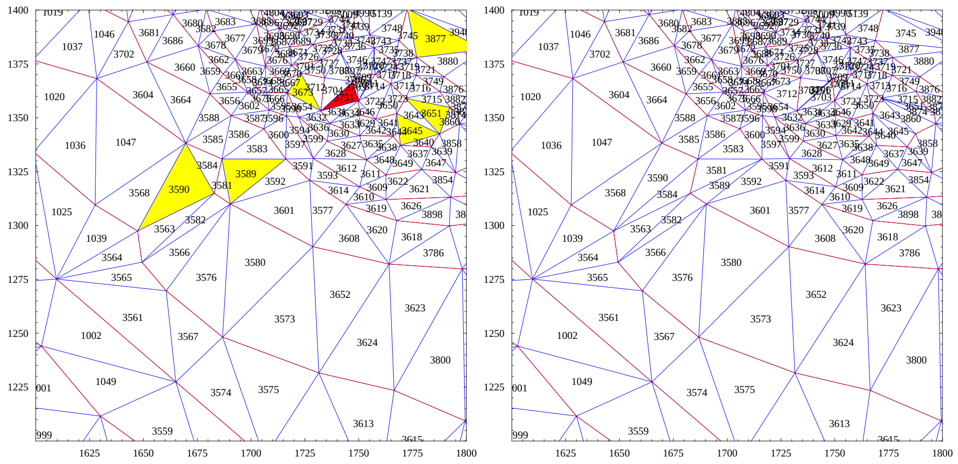

Figure 7. Detail of Askervein Hill triangulation of the Southwest slope. Left: *Original constrained Delaunay triangulation.* Right: *Triangulation after rearrangement.*

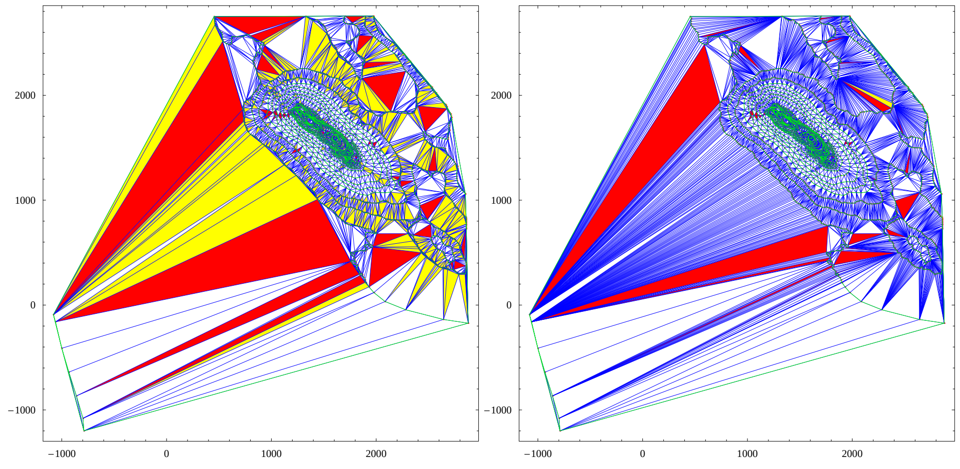

Figure 8. Askervein Hill triangulation before (left) and after rearrangement (right).

that they produce triangles, in these cases 3584 and 3581, that have the same height at each corner. The result is the occurrence of spurious "rice paddies" as also seen at several positions in figure 5, most notably at the large arrow. The right plot of figure 7 shows how the triangles are rearranged in order to avoid "rice paddies." The problematic triangle (for example 3590 in the left plot) and the triangle facing the problematic edge (triangle 3584) form a quadrangle. If this quadrangle is convex it can be cut into two new triangle: 3584 and 3590 on the right plot. The new triangles are tested again and the procedure is continued until no more problematic triangles can be rearranged. This means that we may end up with un-fixable triangles as shown in the right plot of figure 8.

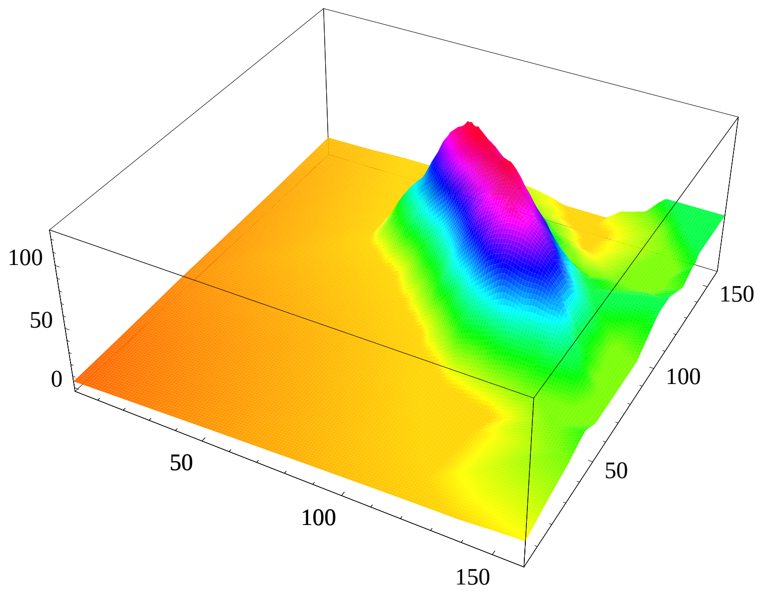

Figure 9. Final 3D interpolation of the Askervein map. The vertical axis is stretched.

Figure 8 shows the constrained Delaunay triangulation of the entire Askervein map (figure 6) before (left) and after (right) the rearrangement. "Problematic" triangles are colored read or yellow. The final interpolated grid is shown in figure 9. Also shown in the result for the Waspvale map (figure 10), which should be compared to figure 2. The smoother appearance is clear.

3.3 Linear Interpolation

The linear interpolation is made by the equation

$$h_{grid} = h_1 - \frac{\mathbf{n} \cdot \mathbf{p}_{grid}}{n_z}, \tag{1}$$

where h_{grid} is the interpolated height at a grid point within a triangle, h_1 is the height at the first point of the triangle, $\mathbf{n}$ is the normal vector to the plane spanned by the 3D-coordinates of the triangle (blue section in figure 11), n_z is the vertical component of the normal vector, and $\mathbf{p}_{grid}$ is the vector from p_1 to the grid point. Δh in figure 11 is given by the last term in (1). The same equation is also used for interpolating roughnesses.

3.4 Roughness

The roughness contours are split into two contours. One that has the value of the roughness to the right in the direction of the contour, and one with the value to the left. These

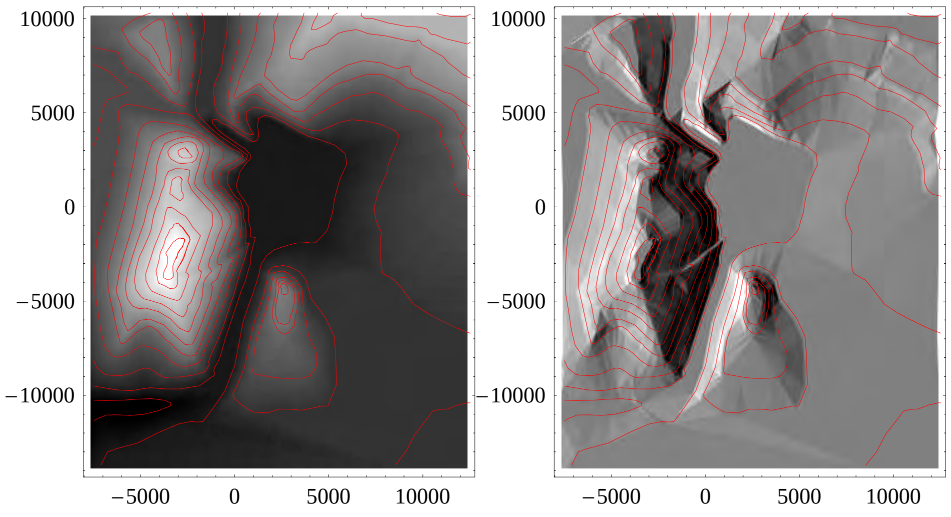

Figure 10. The Waspvale height contours and grid with rearranged triangles.

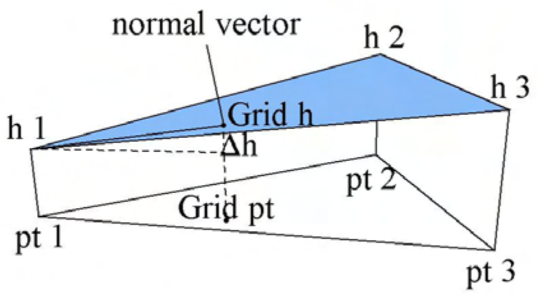

Figure 11. Illustration of simple, linear interpolation.

two contours are separated with a distance arbitrarily set to 0.1 m. The roughness contours are transformed to triangles by a constrained Delaunay triangulation. Without further processing these triangles are used for linear interpolation in the same way as for the height contours. An example of an very difficult roughness map is shown in figure 12. The map has open roughness lines towards the West and erroneous line approximately at (8800, 4200) where the inside roughness of the closed line has been swapped with the outside. The new method treats such inconsistencies more gently. However, such errors should ideally be removed from the maps.

3.5 Extrapolation

Extrapolation is needed when a grid point is outside of the convex hull spanned by the map. The average of all points is used as the center of the convex hull. A point at the convex hull is given the value of a linear interpolation of the two nearest hull points. A grid point outside the convex hull is given the value at the hull where a line from the center to the grid point crosses the hull (see figure 13).

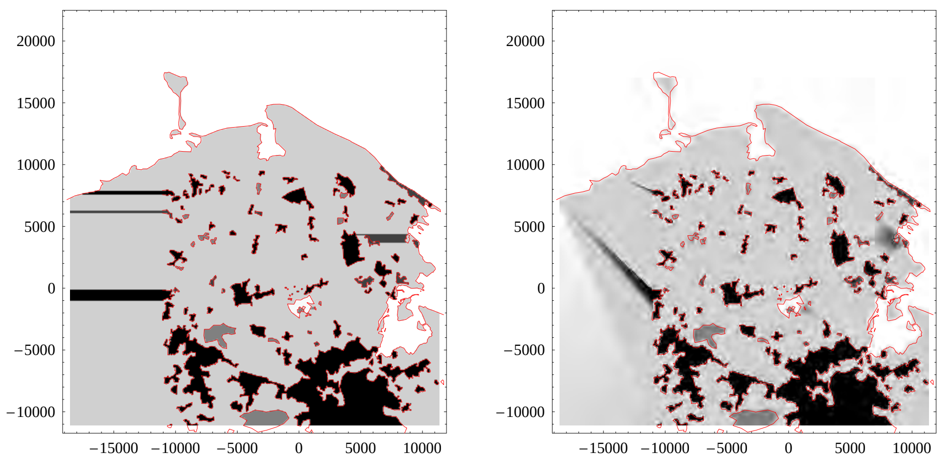

Figure 12. Roughness lines from the northern part of Fyn shown in red together with the grided roughnesses in shades of gray. The existing WAsP Engineering gridding to the left and the new procedure to the right.

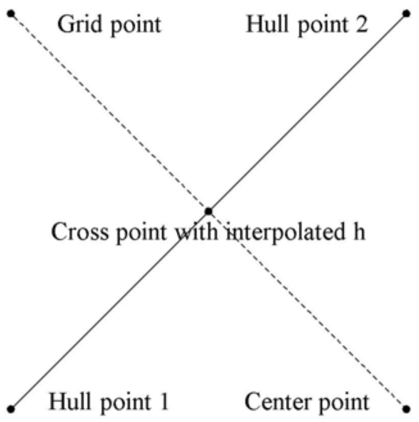

Figure 13. Sketch of the extrapolation procedure. The grid point is given the value h.

If the selected area is chosen away from the points in the map the output will have a default value of -9999 for all grid points. If the number of points is greater than three, then the output will have interpolated and extrapolated values in all grid points.

3.6 Further possibilities

Several possibilities for a more smooth interpolation than the linear have been investigated. These include cubic splines and Bezier surfaces. Two general problems emerged from these attempts:

- Even though the interpolated grid appears smoother almost everywhere, spurious ridges are often produced. This could seriously harm the subsequent flow calculation.
- Lakes, which of course are flat, always appear slightly curved.

Problems with eliminating especially the first item made us stick to the simple and robust linear interpolation. We envisage that a modification of Watson's (1985) natural neighbor interpolation could make the interpolation smoother while still solving the problems stated in section 2.2. Exactly how this could be done is not clear.

4 User interface

Two different versions of the software are made, the only difference being the output. The first produces a regular, cartesian grid suitable as input to WAsP Engineering. The second does not interpolate on a grid, but merely returns the triangles. This is used for the terrain description in EllipSys3D.

4.1 Regular Grid

When the executable is run from a DOS-prompt eight arguments have to be given. It is important that all arguments are stated otherwise the program will stop and no output will be given. The call of the program is

```
Map2Grid_regular <outtype> <filename> <nx> <ny> <lowx> <highx>
<lowy> <highy>
```

These arguments are in order: the type of output (`hght` or `rgh`), map filename with extension (so far it has to be ASCII), the number of points in the x-direction, the number of points in the y-direction, the lowest x-coordinate, the highest x-coordinate, the lowest y-coordinate, and the highest y-coordinate. If the map file is not in the current directory, the full path has to be given, e.g `C:\mapfiles\test.map`. The four last arguments are the corners of the rectangular region in study and they can either be within or go beyond the limits of the contours. An example of a call from a DOS-prompt could be

```
Map2Grid_regular hght C:\test\test.map 301 301 0 10000 1000 11000
```

Here the file with orography and roughness information is `test.map` in the directory test. The output will be a two-dimensional array of heights in Surfer's GRD format (Surfer 2002). 301 points in both directions are chosen. The x-coordinates range from 0 to 10000 and the y-coordinates range from 1000 to 11000. The user should be careful that the second x-coordinate is larger than the first. The program will not crash but give unreasonable output. The same is true for the y-coordinates.

A similar call

```
Map2Grid_regular rgh C:\test\test.map 301 301 0 10000 1000 10000
```

will give an array of roughnesses as output in GRD format.

In the previous examples of calls the output arrays will be printed at the screen. If the user want the output in a file the call is

```
Map2Grid_regular <outtype> <filename> <nx> <ny> <lowx> <highx>
<lowy> <highy> > <outputfilename>
```

where the output file name is chosen by the user.

The structure of the output arrays is the same for both height and roughness: Surfer's GRD format. The output array $h(i, j)$, where $1 \leq i \leq \mathtt{nx}$ and $1 \leq j \leq \mathtt{ny}$ is written with the first index running fastest. The coordinates of the grid points of the output is given implicit by the corners of the region and the number of points in each direction. We have chosen to print at maximum seven interpolated heights per line in the ASCII file.

4.2 Speed performance

The time spend by the computer program is split up into 4 parts. These parts are

1. reading of the map and preparing for the triangulation
2. constrained triangulation
3. reshuffling of triangles,

4. interpolation on a regular grid.

For roughnesses grid generation item 3 is not used. The exact execution time is dependent on the size of the map file, the size of the part of interest, the complexity of the orography, and the number of boxes chosen (resolution of output). Furthermore it is dependent on the quality of the digitized map. If anchor points (special for roughness) are not made carefully and the contours are not closed properly (both height and roughness) more time will be spent to make the output as good as possible. The running time will increase if many of the grid points have to be extrapolated. In short the quality of the output is dependent on the quality of the input. Examples of the execution times of “La Plana” and Waspvale are presented and discussed.

The size of the ASCII map file for La Plana is 10.0 Mb, with a total execution time of 10.5 s for heights and 3.5 s for roughnesses, both generating 301×301 points. The difference in running is due to the fact that La Plana have more height contours than roughness lines. In the case of heights, the reading of map and reshuffling of triangles take about 3.5 s each, and the triangulation takes about 2.5 s. The interpolation takes 1.0 s. This is typical for a large file with many contour lines. The execution time for roughness goes mainly to I/O. Interpolation and triangulation take a few ms each.

The Waspvale file is 30 Kb and we generate again 301×301 points. The running time is for heights 60ms and for roughnesses 70 ms. In this case I/O consumes most time.

4.3 STL Format

Several CAD- and CFD-packages use the STL ("StereoLithography") format among them GridGen from Pointwise, Inc. The call for this program is

```
Map2Grid_STL <filename> <name.stl>
```

The output is given as the standard for STL files requires. The structure of STL files can be found elsewhere, e.g. `http://www.ennex.com/ fabbers/StL.asp`.

The STL file contains the coordinates of the rearranged triangles made by the constrained Delaunay triangulation, their normal vectors, and the height or roughness values at the vertices.

5 Discussion and conclusion

A robust and efficient method of calculating a cartesian grid of heights or roughnesses from contour line maps is developed. The method builds on Delaunay triangulation constrained to include all contour segments in the triangulation. It is furthermore refined to avoid spurious flat areas produced by the Delaunay triangulation. Robust ways to extrapolate beyond the convex hull of the map points are provided.

The algorithms are implemented in C programs, which can be run separately, and as a library integrated into WAsP Engineering. The next major release of that program will contain this library.

This work was supported by the Danish Technical Research Council through grant no. STVF 26–02–0015.

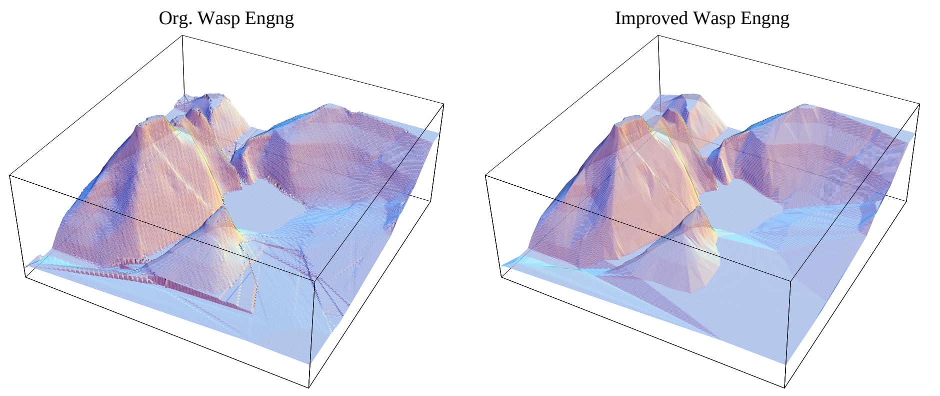

Figure 14. The original WAsP Engineering interpolation (left) and the improved gridding (right).

References

Hansen, A., Mann, J., Johansen, J. and Sørensen, N. N.: 2006, Large-eddy simulation of neutral atmospheric boundary layer, *Geophys. Res. Abstr.* **8**(EGU06-A-02672). (CD-ROM).

Mann, J., Ott, S., Jørgensen, B. H. and Frank, H. P.: 2002, WAsP Engineering 2000, *Technical Report R–1356(EN)*, Risø National Laboratoty.

O'Rourke, J.: 1998, *Computational Geometry in C, 2. ed.*, Cambridge University Press.

Shewchuk, J. R.: 1996, Triangle: Engineering a 2D quality mesh generator and Delaunay triangulator, *in* M. C. Lin and D. Manocha (eds), *Applied Computational Geometry: Towards Geometric Engineering*, Vol. 1148 of *Lecture Notes in Computer Science*, Springer-Verlag, Berlin, pp. 203–222.

Shewchuk, J. R.: 2002, Delaunay refinement algorithms for triangular mesh generation, *Computational Geometry: Theory and Applications* **22**(1–3), 21–74.

Sørensen, N. N.: 1995, General purpose flow solver applied to flow over hills, *PhD thesis Risø–R–864(EN)*, Risø National Laboratory.

Surfer: 2002, *Surfer 8*, Golden Software Inc., Golden, CO, U.S.A.

Taylor, P. A. and Teunissen, H. W.: 1983, Askervein '82: Report on the September/October 1982 experiment to study boundary-layer flow over Askervein, South Uist, *Technical Report Research Report MSRB-83-8*, Atmospheric Environment Service, Toronto, Canada.

Taylor, P. A. and Teunissen, H. W.: 1985, The Askervein hill project: Report on the Sept./Oct. 1983 main field experiment, *Technical Report Research Report MSRB-84-6*, Atmospheric Environment Service, Toronto, Canada.

Walmsley, J. L. and Taylor, P. A.: 1996, Boundary-layer flow over topography: Impacts of the Askervein study, *Boundary-Layer Meteorol.* **78**, 291–320.

Watson, D. F.: 1985, Natural neighbor sorting, *Australian Computer Journal* **17**(4), 189–193.

Risø's research is aimed at solving concrete problems in the society.

Research targets are set through continuous dialogue with business, the political system and researchers.

The effects of our research are sustainable energy supply and new technology for the health sector.

www.risoe.dk