

rojud

a Type-1 Font for the 42 counties of Romania

Vlad Bazon*

November 2020

Abstract

The idea of a font with the “images” of a given set of geographic regions is seeded in the `CountriesOfEurope` font (see [1]), which—as other about 3700 Type-1 fonts in TexLive distribution—is stamped as “% Generated by FontForge”. But how one can *construct* a such font? We describe and apply a general procedure for that.

1 The counties as glyphs

From GADM ([2]) we obtain the administrative map of the country, as a *shapefile* file; this file contain “polygons” for each county, which are closed *paths* (speaking as in Postscript) expressed in geographic coordinates. With a **R** program ([3]), using the **sp** package, we extract the counties contours and project them to cartezian coordinates by UTM¹ (having into account that not all counties maps to the same UTM-zone). The respective paths usually contains very many points (and we will have to keep in mind the Postscript *limitcheck* barrier); so we use the javascript **mapshaper** library ([4]) for to simplify the contours (in the Romania case is sufficient 18% of the initial vertices).

The contours so obtained could be traced or painted in a Ghostscript session, using `moveto` and `lineto`; but in the Type-1 font-file we need to use the relative alternative of these, `rmoveto` and `rlineto`. So by a **R** program we transform the coordinates matrix of each contour, replacing each line with its difference from the previous one; therewith by this **R** program, we formulate a “.raw” file, in which we write these matrixes completed on every line by a “rlineto” word (excepting the first line, where we have “rmoveto”, and the last line which is replaced by word “closepath”).

More precisely, in this “.raw” file we obtain, for each county, the needed glyph definition in the terms of the Type-1 format, for example:

*vlad.bazon@gmail.com

¹https://en.wikipedia.org/wiki/Universal_Transverse_Mercator_coordinate_system

```

/jAB { % județul Alba (Alba county)
0 1204 hsbw % "horizontal sidebearing and width"
74881 100 div 4378 100 div rmoveto
-634 100 div -958 100 div rlineto
-492 100 div -105 100 div rlineto
% etc.
861 100 div -86 100 div rlineto
1067 100 div 293 100 div rlineto
closepath
endchar
} ND %% total: 979 rows

```

Then we insert in this ".raw" file the /CharStrings dictionary definition, the /Encoding definition, etc. – having at beginning (see [6]):

```

7 dict begin
/FontType 1 def
/FontMatrix [0.001 0 0 0.001 0 0] readonly def
/FontName /rojud def
/FontBBox {0 0 1776 1375} readonly def
/PaintType 0 def

```

and ending the ".raw" file (about 77700 lines) by:

```

end end
readonly put
put
dup/FontName get exch definefont pop
mark currentfile closefile
cleartomark

```

From this ".raw" file we obtain the desired .pfb file, using the **t1asm** program ([5]).

Using **pf2afm.ps** from [7] (or an "on-line converter") we obtain the corresponding metric file ".afm", from which using **afm2pl** and **pltof**, we obtain the needed for TeX metric file ".tfm". For the package definition of the new font what else is needed is to add the ".fd" and ".sty" files; of course, these files (.pfb, .tfm, .map, .fd and .sty) must be incorporated in the appropriate places of the TeXLive directory structure.

Actually we add two minimal ".fd" files, "TUrojud.fd" for **xelatex** and "OT1rojud.fd" for **pdflatex**; in the ".sty" file we separate by these cases the symbol declarations and we add a single new command, "\paintt", which use the \special TeX primitive for to paint the glyph in the PDF "FillAnd-Stroke" rendering mode (but fixing the colours, at a perhaps decent level).

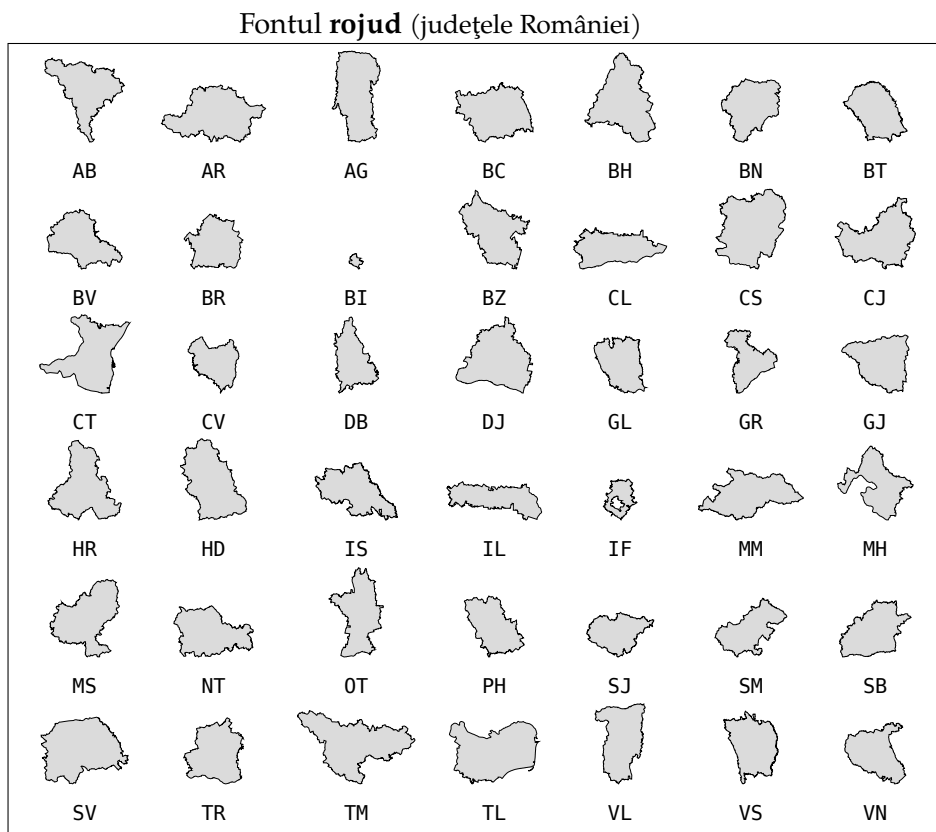
2 Tests (simple examples)

We don't see a practical use for such a font, whose glyphs shape some geographical regions... the interesting issue is their very definition in the font file, described above.

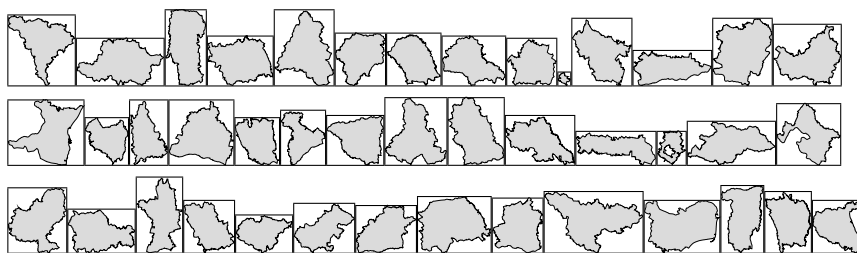
But for simple tests or some examples, consider in the preamble of a LaTeX file `usepackage{rojud}` and let's say, these simplifying commands:

```
\newcommand{\setFont}[1]{\fontfamily{#1}\selectfont}
\newcommand\Sj[1]{\setFont{rojud}\Huge\paintt{#1}}
\newcommand\Sn[1]{\footnotesize\texttt{#1}}
```






It is easy to write a little Python script to generate a TeX file "rows.tex", giving the rows and columns (with cells `\Sj` and `\Sn`, applied to glyphs) of a `\tabular` environment; then `\include rows.tex` in this environment of the initial LaTeX file (see the TeX source of this document), to present the glyphs as follows:












Repeating `{\fbox{\paintt{\symbol{\value{N}}}}}`, with `N` from 167 to 208 (the codes from the /Encoding table of the font) we could produce the glyphs engraved in his bounding boxes (here with `\huge`):








Lastly, we produce a little paragraph of text—in romanian language, at `\small`, `\normalsize` and `\large`—integrating a few glyphs:

„Județul Iași  se învecinează cu județele Botoșani  spre nord, Suceava  spre nord-vest, Neamț  spre vest și cu Vaslui  spre sud.”

„Județul Iași  se învecinează cu județele Botoșani  spre nord, Suceava  spre nord-vest, Neamț  spre vest și cu Vaslui  spre sud.”

„Județul Iași  se învecinează cu județele Botoșani  spre nord, Suceava  spre nord-vest, Neamț  spre vest ...”

„The neighboring counties with  are: , ,  and .

We must note here that in the bounding boxes of the county glyphs we did not leave the usual extra horizontal space (to separate two neighboring glyphs); usually these irregular glyphs will appear in a text interspersed with characters of another font (an not as neighbors).

Also note that (implicitly by the above construction) the county glyphs respects (as possible) the cartographic proportions of the counties.

References

- [1] <https://ctan.org/pkg/countriesofeurope>
A font with the images of the countries of Europe
- [2] <https://gadm.org> *the Database of Global Administrative Areas*
- [3] <https://www.r-project.org/>
a free software environment for statistical computing and graphics
- [4] <https://github.com/mbloch/mapshaper>
A tool for topologically aware shape simplification
- [5] <https://ctan.org/pkg/t1utils>
Simple Type 1 font manipulation programs
- [6] <https://www.adobe.com/jp/print/postscript/pdfs/PLRM.pdf>
- [7] <https://www.ghostscript.com/>