Country meshing with Julia and Triangle

(c) J. Fuhrmann

This notebook downloads a shape file dataset, and meshes a selected country.

Activate temporary Julia environment and add packages. This will take a while if called the first time, as they are downloaded.

```
using Shapefile ,DataFrames ,GeoInterface ,Triangulate ,PlutoVista ,CSV
    ,Printf ,PlutoUI
```

Table of Contents

Dataset loading

This is the name of the dataset to be downloaded:

```
dataset="TM_WORLD_BORDERS-0.3";
```

```
function download_if_needed(fname)
data_url="https://github.com/petewarden/openheatmap/raw/master/mapfileprocess/test_data/TM_WORLD_BORDERS-0.3/"
    if !isfile(fname)
        Base.download(data_url*fname,fname)
    end
    if isfile(fname)
        fname
    else
        "error"
    end
end;
```

```
"TM_WORLD_BORDERS-0.3.shp"
```

```
download_if_needed(dataset*".shp")
```

```
"TM_WORLD_BORDERS-0.3.dbf"
```

```
download_if_needed(dataset*".dbf")
```

Extract meta data table:

```
table = Shapefile.Table(dataset*".shp");
```

Create a data frame from the table:

df =

	geometry	FIPS	ISO2	ISO3	UN	NAME	AREA
1	Polygon(48 Points)	"AC"	"AG"	"ATG"	28	"Antigua and Barbuda"	44
2	Polygon(1241 Points)	"AG"	"DZ"	"DZA"	12	"Algeria"	23817
3	Polygon(871 Points)	"AJ"	"AZ"	"AZE"	31	"Azerbaijan"	8260
4	Polygon(337 Points)	"AL"	"AL"	"ALB"	8	"Albania"	2740
5	Polygon(418 Points)	"AM"	"AM"	"ARM"	51	"Armenia"	2820
6	Polygon(1113 Points)	"AO"	"AO"	"AGO"	24	"Angola"	12467
7	Polygon(72 Points)	"AQ"	"AS"	"ASM"	16	"American Samoa"	20
8	Polygon(3781 Points)	"AR"	"AR"	"ARG"	32	"Argentina"	27366
9	Polygon(8340 Points)	"AS"	"AU"	"AUS"	36	"Australia"	76823
10	Polygon(111 Points)	"BA"	"BH"	"BHR"	48	"Bahrain"	71
	more						
246	Polygon(363 Points)	"TW"	"TW"	"TWN"	158	"Taiwan"	0

```
df=DataFrame(table)
```

Extract information

Extract shape information from table:

```
geoms = Shapefile.shapes(table);
```

This is the way we figure out which is the index of the country in the data. We use the two character ISO2 label:

```
find_country_row(ISO2::String)=findall(df.ISO2 .== ISO2)[1];
```

Contry data seem to consist of several paths for the different connected components like islands etc. We extract the largest one assuming that this usually defines the country main shape. This is not perfect, for the US, we get e.g. only Alaska...

```
find_country_paths(ISO2::String)=geoms[find_country_row(ISO2)];
```

```
find_largest_path(ISO2::String)=find_largest_path(find_country_paths(ISO2));
```

```
function find_largest_path(paths::Shapefile.Polygon)
    coord=GeoInterface.coordinates(paths)
    npaths=length(coord)
    pathsize=[]
    for i=1:npaths
        push!(pathsize,length(coord[i][1]))
    end
    largest_path=findmax(pathsize)[2]
    x,y=clean_path_coordinates(coord[largest_path])
end;
```

We must clean the data a bit as there are some badly positioned close points which make Triangulate crash:

```
function clean_path_coordinates(path;tol=1.0e-3)
    x=Float64[]
    y=Float64[]
    for coord in path[1]
        dx=0
        dy=0
        discard=false
        l= length(x)
        if l>0
            for i=1:l
                dx=x[i]-coord[1]
                dy=y[i]-coord[2]
                dist=sqrt(dx^2+dy^2)
                if dist <tol
                    discard=true
                    continue
                end
            end
        end
        if !discard || l==0
          push!(x,coord[1])
          push!(y,coord[2])
        end
    end
    x,y
end;
```

Create a triangulation for a country given by ISO2 code using Triangle by J.R.Shewchuk. Assume that the point list describes a closed path.

```
function countrymesh(country;maxarea=0.1)
    x,y=find_largest_path(country)

    npoints=length(x)
    border_segments=Array{Cint,2}(undef,2,npoints)
    for i=1:npoints-1
      border_segments[:,i].=[i,i+1]
    end
    border_segments[:,npoints].=[npoints,1]

    tin=TriangulateIO(pointlist=hcat(x,y)',segmentlist=border_segments)

    flags=@sprintf("pVqea%f",maxarea)

    out,vout=triangulate(flags,tin)
    out
end;
```

```
triout =
TriangulateIO(
pointlist=[10.979443000000003 10.95555500000006 … 11.369015312173962 11.501533806151953; 5
pointmarkerlist=Int32[1, 1, 1, 1, 1, 1, 1, 1, 1, 1  …  0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
trianglelist=Int32[924 2450 … 4410 4395; 1622 1478 … 4370 4410; 925 2782 … 4406 4409],
segmentlist=Int32[1 2 … 2956 3251; 1602 3 … 1572 677],
segmentmarkerlist=Int32[1, 1, 1, 1, 1, 1, 1, 1, 1, 1  …  1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
edgelist=Int32[924 1622 … 4410 4409; 1622 925 … 4395 4410],
edgemarkerlist=Int32[0, 0, 1, 0, 0, 0, 0, 1, 0, 0  …  0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
)
```

```
triout=countrymesh(country,maxarea=maxarea)
```

Number of triangles: 6951

Plot

```
trimesh(triout.pointlist,triout.trianglelist,resolution=(500,500))
```

Adjust country and maximum triangle area:

```
country = "DE"
```

```
country="DE"
```

```
maxarea = 0.025
```

```
maxarea=0.025
```

These plots appear to be distorted, as the input data are given in longitudes and latitudes instead of distances.