Appendix C: Technical Appendix

August 20, 2015

This appendix explains the technical side of the interpolation programme. I first give some more background on how the programme operates. I then raise the crucial question of how large to make raster (how detailed to make the map of the boundaries). I show that whilst increasing the raster size increases running time in a non-linear fashion, the actual projections are extremely highly correlated above a certain size.

1. OUTLINE OF THE METHOD

Areal interpolation in general requires shapefiles and uses the 'zonal statistics' command in the relevant software package. My programme uses a raster (pixelated grid) and a shapefile to extract some function, e.g. mean or standard deviation, of the values of the set of raster pixels included inside each polygon in the shapefile.[1] The R command 'extract' forms the core of this programme; it provides an essential benefit by directly calculating the allocation of pixels into polygons—rather than only reporting a summary statistic. This is desirable as it means the most computationally intensive stage of the programme needs only be run once for any two sets of boundaries. By using the extracted distribution to directly calculate the transfer weights between old constituencies J and new constituencies K, one can calculate any number of statistics using (much faster) matrix multiplication.

All versions of the programme further rely on the trick of assigning a unique (prime) id to each sub-constituency unit (again generically referred to as a ward) and constituency. It then creates two raster using that unique id for both the old boundaries and the ward boundaries. It then multiplies those two rasters together, i.e. creates a new pixelated grid in which each pixel's value is the product of its (old) constituency id and its ward id. From here, the distribution of pixels into the new boundaries is extracted. Because all of the original ids were unique primes, it is possible to factorise the resulting output to calculate the (old) constituency-ward composition of each new constituency. As wards may be split across constituencies due to deliberate changes or slight losses in resolution of the raster, it is important to know $s_{j,l}$ exactly without assuming indivisible wards. Indeed, such split wards are highly likely if one compares quite distant elections as local reorganisations may have occurred.

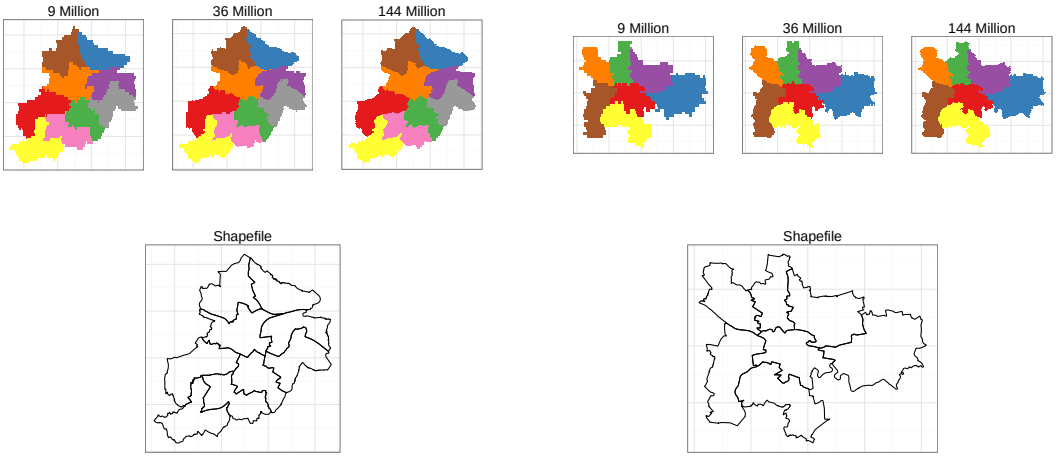
An additional feature of the programme is a 'repair' option; this is needed when some of the sub-constituency units fail to be rasterised due to their small size, e.g. very small metropolitan wards. The repair option notes that their population should still be included in the analysis; it locates the closest pixel to the un-rasterised ward and 'repairs' the rasterization by assigning the population of the un-rasterised ward to the ward of the closest pixel. The programme will note when this occurs as well as the percentage (and number) of wards that fail to be rasterised. The default setting is to use the Repair procedure when under 99% of wards are rasterised. Whilst ideally one would wish to make the number of un-rasterised wards as small as possible, this might require creating undesirably long interpolation times with little gain in quality. All interpolations reported in the main paper have this option enabled; details as to its necessity are listed in Appendix A.

[1] The relevant package in R is **raster** and the function is 'zonal'; QGIS, ArcGIS, and Python have an array of similar capabilities with commands generally denoted as 'zonal statistics'.

2. A QUESTION OF SIZE

A key issue in this method and any spatial interpolation involving rasters is how many pixels should the map be divided into. There is no firm answer, though more is likely better, but this also corresponds into a non-linear increase in computation time. The 'lower bound' on the size is likely that in which all old constituencies (and/or wards) are assigned at least one pixel, but the exact number depends on the relative sizes of the constituencies used.[2] To illustrate this point, Figure 1 shows the shapefiles for Glasgow and Birmingham (two areas with dense populations and numerous (geographically) small constituencies) using the 2010 boundaries; it also generates three rasters of British constitutions at different resolutions (9 million, 36 million, 144 million) and shows the relevant sub-region of each raster. Whilst some granularity is visible, particularly

Figure 1: Shapefiles versus Rasters (Pixelated Grids) at Varying Resolutions



(a) Birmingham (b) Glasgow

on the smallest resolution (3,000 by 3,000), the figures show that there is a close correspondence between the shapefile and the pixelated grid. As the constituencies and wards increase in size, this problem of granularity becomes increasingly unimportant as its impact will be small. For the analysis presented in the main paper, I use the following resolutions: 6,000 by 6,000 (36 million pixels) for Australia, Germany, Great Britain, and New Zealand. This results in nearly all sub-constituency units being successfully rasterised. I use a larger resolution for Canada (10,000 by 10,000) as it seems that the inclusion of geographically large providences means a higher resolution is needed to get sufficiently high coverage. In the US, I rasterise each state separately and thus a resolution of 6,000 by 6,000 is appropriate. These resolutions are likely 'overcautious' meaning that smaller ones would generate similar results, see Appendix B.

The subsections below examine the effect of raster size on the results and confirm they are highly robust. It focuses on the Canadian case (projecting the 2000 boundaries onto the 2004 ones) insofar as this is likely to show the biggest sensitivity to increasing the raster size.

2.1. *Running Time*

Figure 2 compares the running time of the programme at different sizes of the raster (in 1000s of pixels). It splits the time taken into four parts: the Rasterisation stage (creating the rasters of the old boundaries and the population sub-units), the Extract stage (extracting the distribution

[2]Because of the rectangular nature of the raster, a proportion of these pixels lie outside the shapefiles and are not used.

of pixels in the multiplied raster on the new boundaries), the Repair stage (assigning un-rasterised wards to the nearest pixel) and the Overall time. The Overall time also includes the time taken to transform the data from the Extract stage into a convenient form as well as multiplying the old data by the transformation weights.

I compare two specifications in R. The R specifications differ by whether they require the GDAL package (**rgdal**) or if they use default package for raster analysis (**raster**) in R. The reason for permitting two different packages is twofold; first, **rgdal** requires the installation of auxiliary GDAL programmes that may not be possible for all researchers, e.g. those using a networked computer. Second, the GDAL method requires writing the raster to a file on the computer (which the programme itself is told to remove once it is no longer needed); as some of the rasters may be quite large (e.g. a 12,000 by 12,000 raster is $\approx$ 1 GB in size), this may not be ideal for all researchers. By contrast, using the default raster package does not require writing to a file but stores its rasters 'in memory'.[3] Figure 2 shows that GDAL is clearly preferable as it is much faster than the default raster package, especially when considering very large rasters. Interestingly, as the raster size increases, the main time 'cost' to the **rgdal**-version is the Extract stage, whilst the Rasterisation becomes increasingly costly using **raster**. The slight decline in processing time for the largest resolution (14,000) pixels is somewhat inexplicable and is likely a by-product of how the timing tests were implemented in the CONDOR processing system.

2.2. *Robustness of Projections*

This section suggests, however, that using reasonably detailed projections is important. Figures 3 and 4 shows a heat map of the correlations between projections (using population-weighted interpolation) at different raster resolutions for the pooled party vote variables. The 'repair' algorithm is enabled for all interpolations and the **rgdal** method is used.

Unsurprisingly, the results are extremely similar. The vast majority of correlations are extremely high, e.g. above 0.90, and there appears to be only minimal change above around approximately 5,000 by 5,000. The results do suggest, however, using a reasonably high resolution for the dasymetric method insofar as there are quite pronounced differences in the interpolation results for small resolutions, especially 1,000 or 2,000. Such resolutions also generate a warning in the programme that a number of constituencies are not rasterised and thus the results are wholly unreliable.

The general results here confirm that researchers should use the **rgdal** method if possible and select raster sizes of a sufficient detail to ensure that a high proportion of wards are rasterised. Given those desiderata, however, it seems that further detail simply increases the running time of the programme without materially altering the results.

[3] Differences between the two methods regarding the interpolation weights are minute. The minimum correlation between the **rgdal** and **raster** method is 0.999414 for any variable or resolution used in Figures 4 and 3.

Figure 2: Running Time by Programme Method

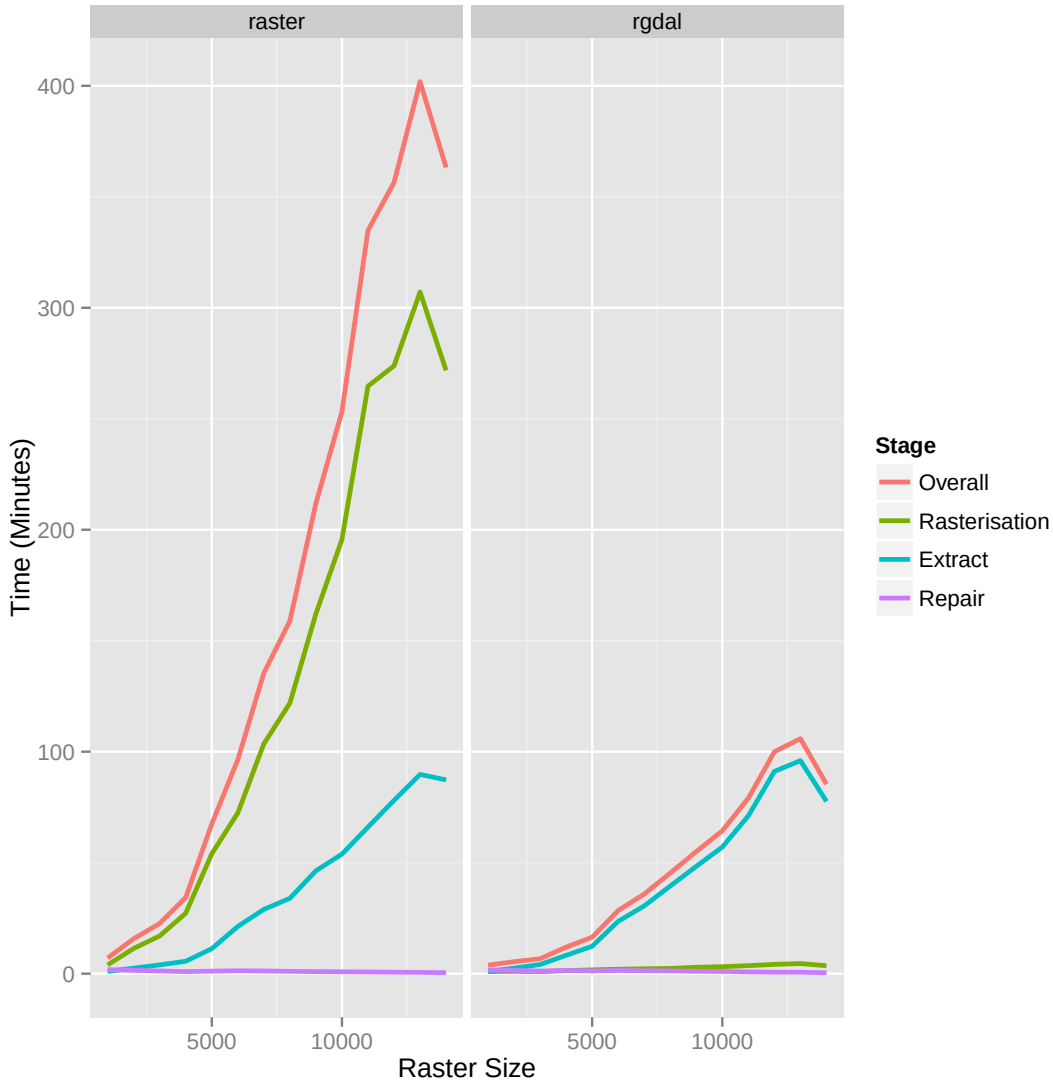


Figure 3: Correlations Between Projections of Electoral Variables: Areal Weighting

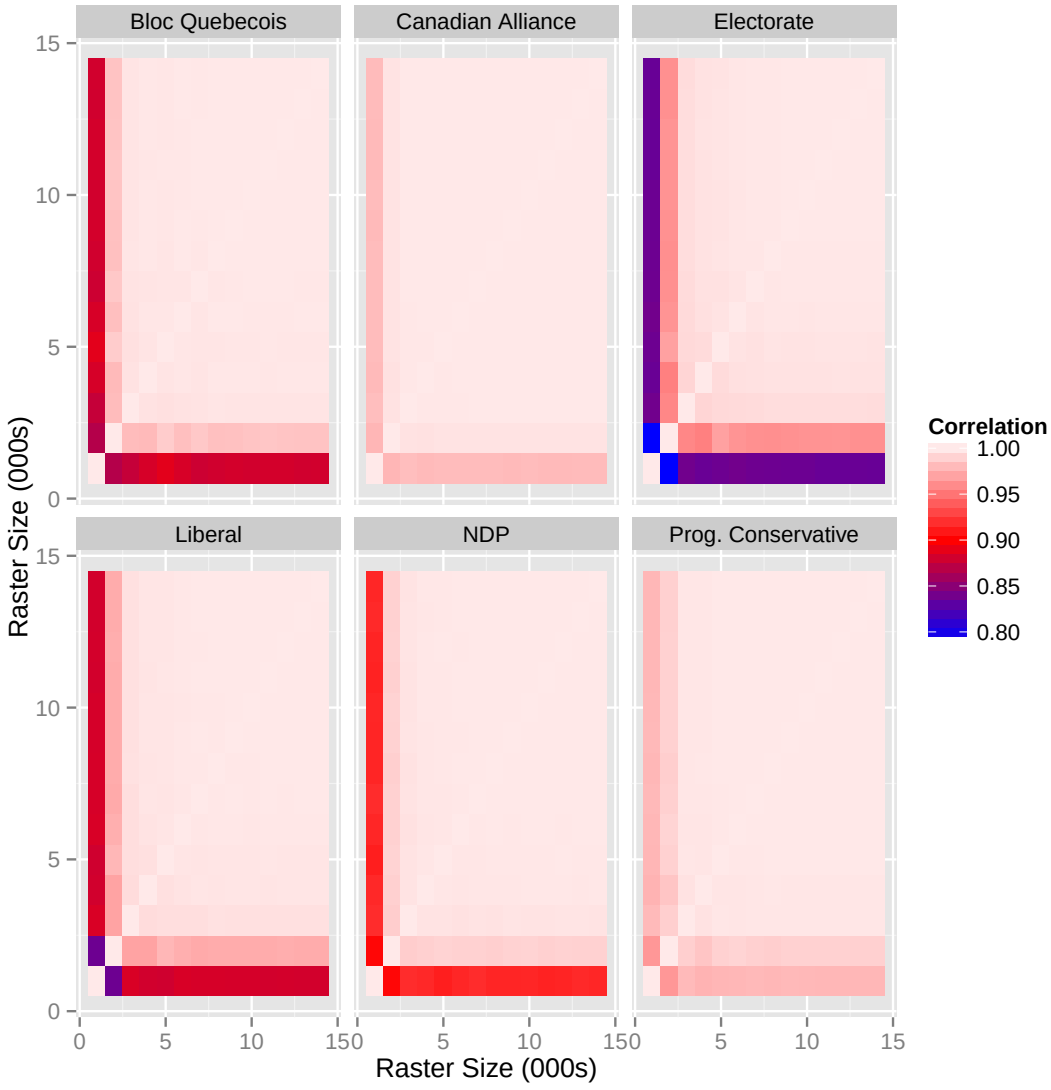


Figure 4: Correlations Between Projections of Electoral Variables: Dasymetric Interpolation