Appendix B: Simulation Appendix

August 20, 2015

This appendix provides further information on the Monte Carlo simulations. It describes each step of the procedure and provides more information on the results of the simulations. R code included in the Supplementary Materials contains the exact code used. This code relies on parallelisation in R to speed up implementation.[1]

1. SPATIALLY AUTOCORRELATED DATA

I generate spatially autocorrelated data in R using the **gstat** package.[2] It allows one to simulate data using a variety of methods; I rely on 'unconditional Gaussian simulation' for a given theoretical variogram (autocorrelation structure).[3] Four parameters are set for each use of **gstat**: β, p, m, r. β defines the expected value of the simulated data (in the absence of autocorrelation); I set it to be 100 for the population data and 1 for the proportion of government supporters. In the latter case, I rescale the generated data to range from 0 to 1. p denotes the sill, i.e. the variance of simulated data in the absence of autocorrelation. I set $p = 1$ for generating the propensity data and $p = 1000$ for the population data. To prevent the emergence of 'negative' population, I rescale the data by ensuring the smallest population value is equal to zero, i.e. subtract the minimum value of the raster from every point in the raster.

m denotes the model or type of variogram used; I use the common 'exponential' model in all simulations.[4] This model has the key property that spatial autocorrelation asymptotically approaches zero as the distance between any two points increases. Finally, r is the range of the variogram and this is varied to create differences in the extent of spatial autocorrelation. The 'theoretical' range is the parameter value set in **gstat**, however, due to the asymptotic nature of an exponential variogram, the more relevant parameter is the 'effective range', i.e. the distance at which 95% of the autocorrelation has disappeared or equivalently the value of the variogram is 95% of the sill. In the exponential variogram model, this occurs at approximately three times the effective range noting that $1 - e^3 \approx 0.95$.

One point to note when analysing this model, however, is that the distribution of the *vote* is a product of two (independent) distributions each with spatial autocorrelation. Thus, the relevant parameter is some combination of the two ranges. I simplify this by assuming a constant range for the population in all models (0.25). This is a large effective range and thus assumes a reasonably high degree of spatial autocorrelation in the population data. I vary the range of the proportion of government supporters ($r_{\text{prop}} \in \{0.01, 0.05, 0.10, 0.20, 0.25\}$) to create distributions, shown in the main text, that have quite different degrees of spatial autocorrelation.

2. CONSTITUENCIES

As noted in the main text, one set of constituencies is generated by first taking N independent draws from a truncated normal distribution $TN(\mu, \sigma)$ with a lower limit of 0 and an upper limit

1. **doParallel** is used. Revolution Analytics and Weston 2014.

2. Pebesma 2004.

3. The exact code is `gstat(formula=z~1, locations=~x+y, dummy=T, beta=`β`, model=vgm(...))` where the content of `vgm` varies and β varies depending on whether I am simulating population or vote share.

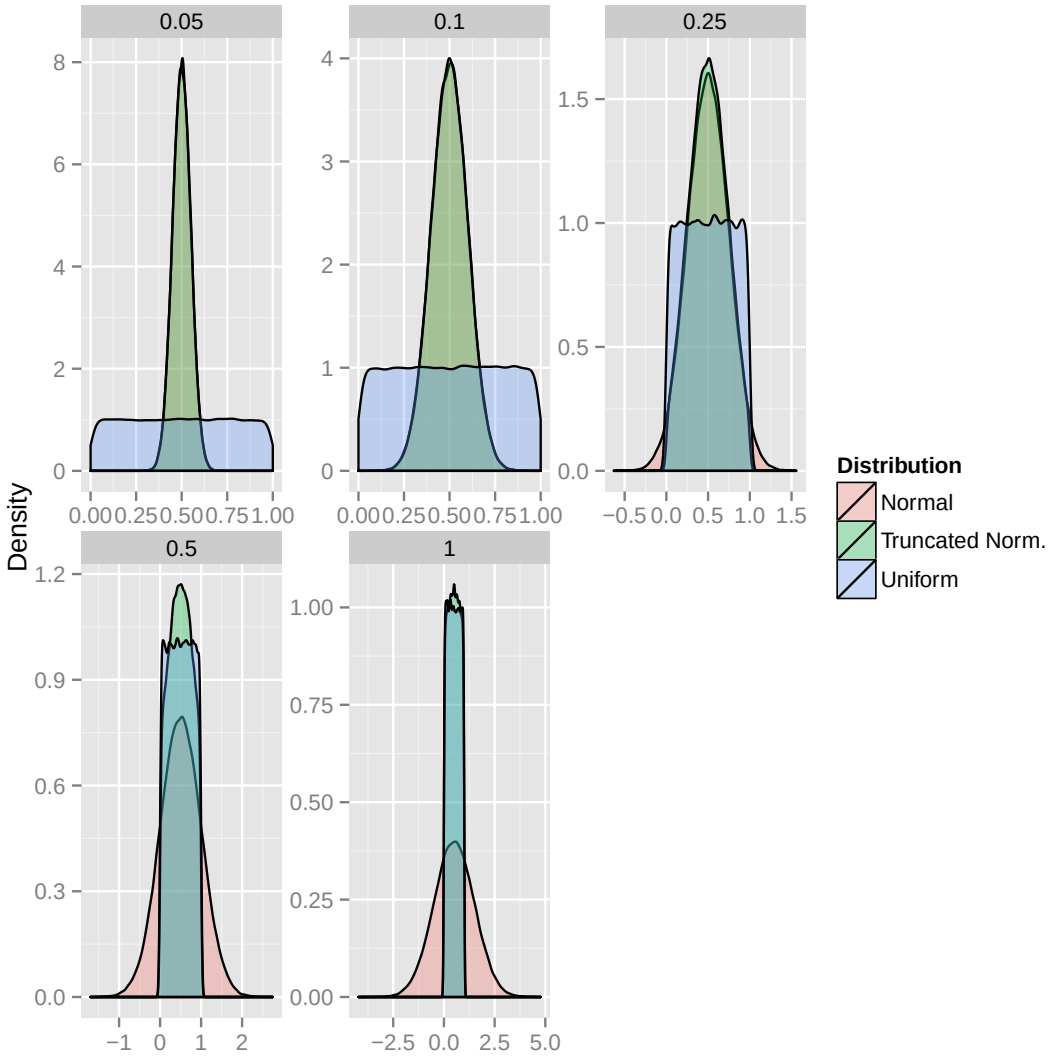
4. Formally, and assuming the nugget to be zero, $\gamma(h) = (p)(1 - e^{-h/r})$ where h represents the distance between any two points.

of 1. The PDF of TN is zero for x outside of the limits and the PDF for $x \in [0, 1]$ is shown below, where ϕ and Φ denote the PDF and CDF of the (untruncated) normal distribution with mean μ and standard deviation σ. I set $\mu = 0.50$ for the remainder of the analysis.

$$f(x, \mu, \sigma) = \frac{\phi(x, \mu, \sigma)}{\Phi(1, \mu, \sigma) - \Phi(0, \mu, \sigma)} \tag{1}$$

It can be shown that as $\sigma \to \infty$, then TN approximates a uniform distribution on the unit interval. As $\sigma \to 0$, any x is less likely to cross the truncation boundaries and $(\Phi(1, 0.50, \sigma) - \Phi(0, 0.50, \sigma)) \to 1$. Thus TN should approximate the untruncated normal distribution with $\mu = 0.50$ and standard deviation σ. Values are drawn using the **msm** package in R.[5] Graphically, Figure 1 provides an illustration by plotting 100,000 draws from each distribution for a given standard deviation.

Figure 1: Truncated Normal Distribution with Varying σ



Creating the constituencies themselves begins by taking a sets of N i.i.d. points from $TN(0.50, \sigma)$ to get the x- and y-coordinates that lie inside a unit square. I then append the coordinates $(0, 0)$

5. Jackson 2011.

and $(1, 1)$ to the list to create the set of points from which Voronoi polygons, also known as Dirichlet polygons, are constructed using the **deldir** package.[6] It creates N polygons such that for any polygon i corresponding to point n, all points inside i are closer to point n compared to any other point $n' \in N$. As the figure in the main paper showed, more spread out points (smaller σ) correspond to more equally sized polygons whilst tightly clustered points (given fixed boundaries of the unit square) create more heterogeneity in polygon size.

3. GERRYMANDERING

As noted in the main text, gerrymandering is an extremely complex and varied phenomenon. The method implemented here is merely an attempt to capture a simple version of it. It works in the following way: first, generate two sets of boundaries ('old' and 'new') using the process above. Second, create a new set of polygons (shapefile) formed by the 'intersection' of the two sets of boundaries.[7] Then, select some proportion (g) of those small polygons and apply the gerrymandering algorithm. Call these the 'candidate' polygons. This algorithm will attempt to gerrymander the new constituencies.

1. Note the neighbours for each polygon and the share of government support in each 'new' constituency (as currently formed).[8]

2. For the relevant candidate, check whether any of its neighbours meet the following criteria.

   - They are not a member of the same new constituency.
   - The share of government support is higher in the new constituency to which the neighbour belongs than in the candidate polygon. If multiple neighbours fit this criteria, select the neighbour for which the 'new' constituency's share is the highest.

3. If that provides a unique neighbour, then assign the candidate polygon to that new constituency. If it provides multiple neighbours, these will almost certainly be in the same 'new' constituency (unless two neighbouring 'new' constituencies had exactly the same government support). Randomly select one and assign the candidate polygon to that constituency. However, if the suggested assignment would mean that one new constituency ceased to exist (i.e. the candidate is the only remaining small polygon in a new constituency), then do not assign the candidate.

4. Recalculate the share of government support in each new constituency as modified by the above procedure.

This leads to polygons that have highly irregular shapes—corresponding to the classic images of gerrymandering in the United States.[9] Figure 2 shows an example for varying g.

Further inspection of the data confirms that the gerrymandering procedure does create the expected differences between the un-gerrymandered and modified boundaries. For each simulation and set of boundaries used, the standard deviation of the area of all polygons was recorded.[10] Further, I also noted the average vote share for the government (i.e. averaging the twenty-two constituencies shares together) as well as the number of government 'wins' (i.e. the number of seats where the government share is more than 50%). Figure 3 uses this data to compare the unmodified (new) constituencies against their gerrymandered version. It confirms that increased gerrymandering works in the expected way: it increases the heterogeneity of the constituency sizes as well as creating a higher average government share, on average, as well as more government 'wins'.

6. Turner 2014.

7. Somewhat confusingly, the relevant R command is 'union' from the **raster** package. Hijmans 2014.

8. The relevant R package here is **spdep** and the function is 'poly2nb'. Bivand 2014.

9. The procedure used will not guarantee contiguity of the gerrymandered polygons but the reliance on only moving 'neighbours' means it is generally preserved.

10. The average remains unchanged because the total area and number of constituencies is unchanged.

Figure 2: Gerrymandered Constituencies with Varying g

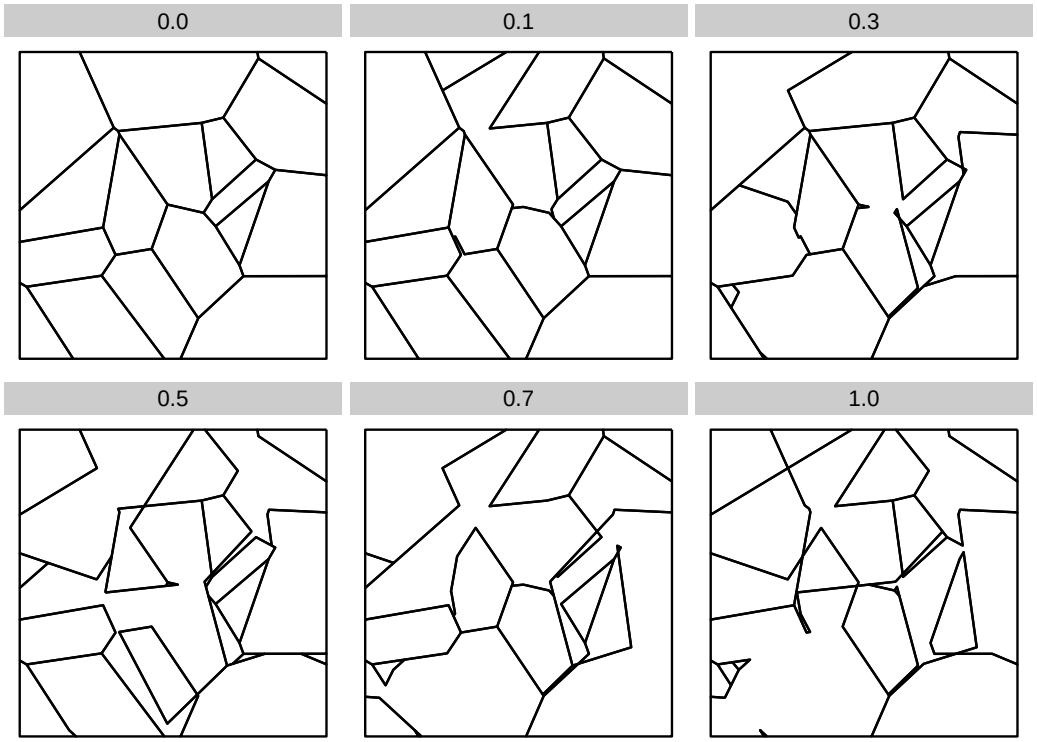
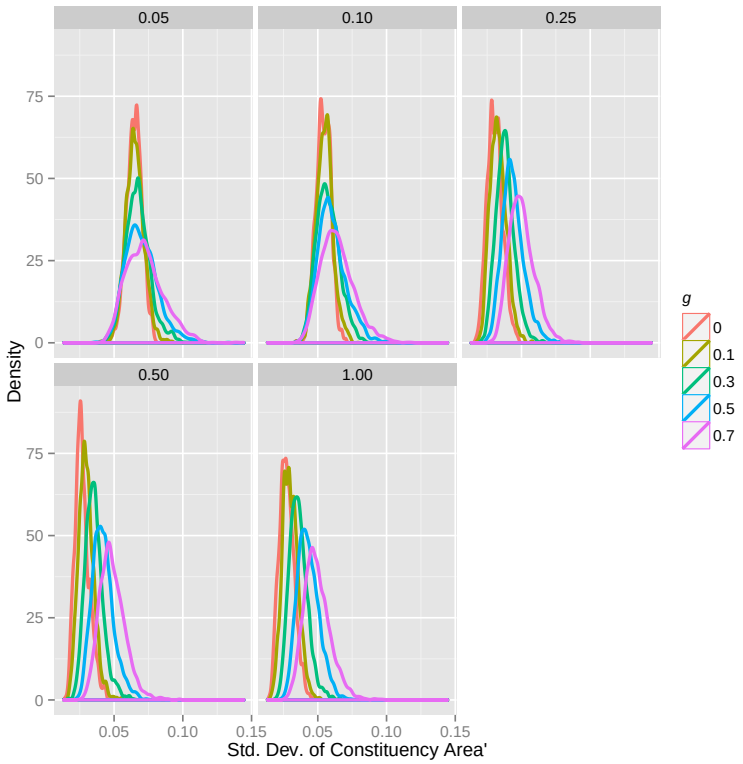
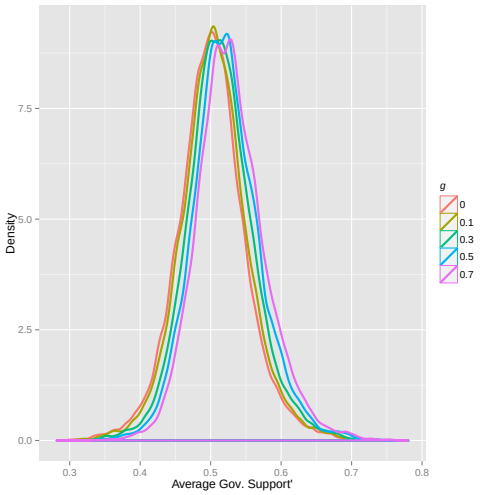


Figure 3: Effects of Gerrymandering

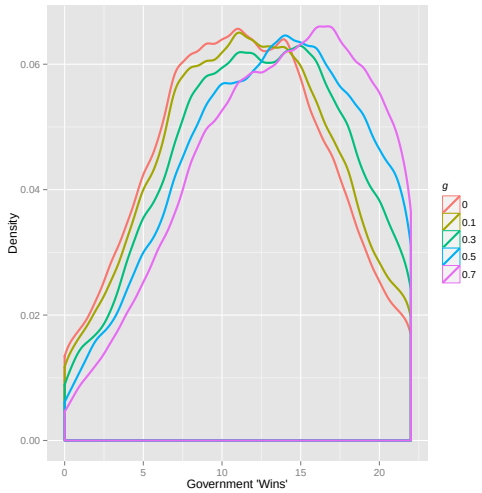
(a) Standard Deviation of Constituency Sizes, Facetted by σ



(b) Average Government Vote Share



(c) Number of Government 'Wins'



4. RESULTS OF THE SIMULATIONS

The main paper presented a heat map of the simulation results. The replication R code contains the data and models to allow a reader to verify the statistical significance of the simulation parameters. Some other error statistics, e.g. the inter-quartile range of the absolute error, are also included. One point to note is that whilst 18750 iterations were run, 13 failed—presumably due to some polygons being too small to rasterise successfully given the raster size specified (1,000 by 1,000 pixels). Thus, there are 187370 observations in each regression.

References

Bivand, Roger. 2014. *spdep: Spatial Dependence — Weighting Schemes, Statistics and Models.* R package version 0.5-77. `http://CRAN.R-project.org/package=spdep`.

Hijmans, Robert J. 2014. *raster: Geographic Data Analysis and Modeling.* R package version 2.3-12. `http://CRAN.R-project.org/package=raster`.

Jackson, Christopher H. 2011. Multi-State Models for Panel Data: The msm Package for R. *Journal of Statistical Software* 38 (8): 1–29.

Pebesma, Edzer J. 2004. Multivariable geostatistics in s: the gstat package. *Computers & Geosciences* 30:683–691.

Revolution Analytics and Steve Weston. 2014. *doParallel: Foreach Parallel Adaptor for the parallel Package.* R package version 1.0.8. `http://CRAN.R-project.org/package=doParallel`.

Turner, Rolf. 2014. *deldir: Delaunay Triangulation and Dirichlet (Voronoi) Tessellation.* R package version 0.1-7. `http://CRAN.R-project.org/package=deldir`.