Padgham, Mark. 2019. “Dodgr: An R Package for Network Flow Aggregation.” *Findings*, February. https://doi.org/10.32866/6945.

TRANSPORT FINDINGS

dodgr: An R Package for Network Flow Aggregation

Mark Padgham[1]

[1] Inter-Faculty Department of Geoinformatics, University of Salzburg, Austria

Keywords: graph, street networks, transport networks, flow aggregation, flow allocation, software

https://doi.org/10.32866/6945

Findings

Allocation of flows throughout networks is a common problem in transport research, yet there are very few computational tools available, and almost no open source tools. We describe a new software package, “dodgr” (**D**istances **O**n **D**irected **GR**aphs) capable of extremely efficient flow aggregation over millions of routes within a network. This package represents a much needed advance in transport research, enabling aggregation at the finest scales of individual street segments, with results able to be direcly viewed with modern web-rendering and interactive viewing tools such as deck.gl.

RESEARCH QUESTION AND HYPOTHESES

Flow allocation is the process of aggregating flows or densities of movement throughout a network given an input set of origin and destination points, a matrix of pairwise flow densities, and an underlying network through which movement must be routed. Routes must be calculated between every pair of origin and destination points and corresponding flow densities aggregated along every segment of each route. We refer to this process as “flow aggregation.”

Flow aggregation is a special class of the betweenness centrality metric. The latter is commonly used to characterize networks through calculating all-pairs shortest paths and aggregating unit values for each path that traverses a given edge. In contrast, flow aggregation utilizes user-defined flow densities between defined sets of points, making it difficult to calculate using standard software tools such as the igraph or NetworkX packages (Csardi and Nepusz 2006; Aric, Schult, and Swart 2008).

Adapting existing software to address flow allocation problems generally requires calculation of, and aggregation along, individual paths (shortest or otherwise) between all pairs of points. A typical urban street network might contain several hundred thousand points, and even aggregating flows between a few percent of these quickly becomes computationally intractable. Accordingly, software for flow aggregation generally approaches the problem through one of two simplifications:

1. Through converting networks to raster cells and aggregating on the resultant grids (as implemented, for example, in (GRASS Development Team 2017)) and

2. Through forgetting the network entirely and focusing instead on effective visualization of interactions between a suite of origin and destination points (see (Ratti et al. 2010), for example).

We describe here a new open-source software package for the **R** language, which makes no such simplification, permitting extremely powerful and detailed analyses of network flows. The “dodgr” package (**D**istances **O**n **D**irected **GR**aphs) is capable of extremely efficient flow aggregation within a network between thousands to tens of thousands of points, and over millions of routes.

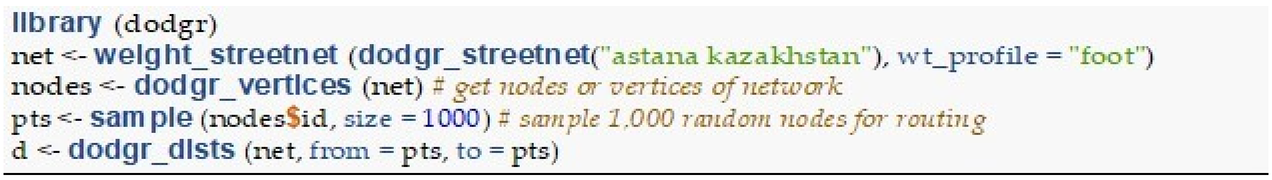
METHODS AND DATA

The dodgr package has been intentionally developed to be adaptable to any type of network, with a particular focus on flow aggregation through street networks. Networks from Open Street Map (OSM) may be readily imported into **R** with the osmdata package (Padgham et al. 2017), and dodgr also includes a helper function `(dodgr\_streetnet)` to enable simple downloading of a street network for a given city with a single line, `net \<- dodgr\_streetnet(\"my\_city\")`. The dodgr package works with graphs in almost any rectangular format, the minimum being columns indicating starts and ends of each edge or segment and corresponding distances. The function `weight\_streetnet` converts street networks in standard formats (such as Simple Features) to the equivalent dodgr format in which each network edge or segment is a single row, with distances weighted according to a specified routing preference (such as pedestrians).

Most dodgr calculations, including the classic routing problem of distances, are based upon variously modified Dijkstra shortest path algorithms, yet with internally bundled and highly optimized heap sort algorithms (also open to user specification), along with efficient parallel computation. This results in standard distance calculations generally being 5–10 times faster than the benchmark “standard” network distance calculations of igraph (Csardi and Nepusz 2006) or networkx (Aric, Schult, and Swart 2008).

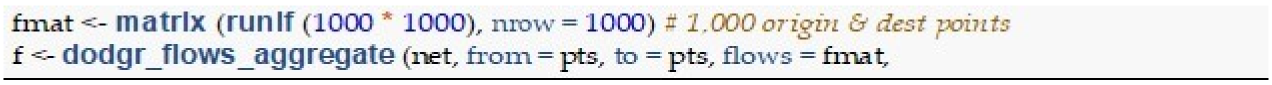
To further boost efficiency, dodgr offers the ability to contract networks by removing all vertices intermediate to street junctions, referred to as “graph simplification” in the python package osmnx (Boeing 2017). OSM networks generally contain 5–10 intermediate nodes for every junction vertex, and graph contraction reduces sizes of graphs and increases computational efficiency by similar proportions.

The unique ability of dodgr that is of particular note here is the `dodgr\_flows\_aggregate function`, which works just like distance calculation, yet with an additional argument called “flows,” specifying a matrix of flow or movement densities between given origin and destination points. Rather than returning a simple matrix as is done for distance calculations, this function simply appends an additional “flow” column to the input network, which quantifies aggregate flows along each network edge. This “flow” column will equal the betweenness centrality of a network in the degenerate case with both origins and destinations equal to all nodes in a network, and a “flow” matrix containing all ones. Most applications will diverge from this case; the



```
library (dodgr)
net <- weight_streetnet (dodgr_streetnet("astana kazakhstan"), wt_profile = "foot")
nodes <- dodgr_vertices (net) # get nodes or vertices of network
pts <- sample (nodes$id, size = 1000) # sample 1,000 random nodes for routing
d <- dodgr_dists (net, from = pts, to = pts)
```

Code Figure 1



```
fmat <- matrix (runif (1000 * 1000), nrow = 1000) # 1,000 origin & dest points
f <- dodgr_flows_aggregate (net, from = pts, to = pts, flows = fmat,
```

Code Figure 2

algorithm is optimized to efficiently aggregate flows through only aggregating over specified paths and through parallelization, where the results of each thread are dumped to disk to await final aggregation over threads on job completion.

The dodgr package also includes an additional function, `dodgr\_flows\_disperse`, to aggregate flows where only destinations are known, implementing a user-defined dispersal function. The resultant flows are directed just as the input graphs are, meaning that flows from *A* to *B* may not equal flows from *B* to *A*. While this distinction of direction is important for many analyses, it is impractical for visualization, and the package accordingly provides a function, `merge\_directed\_flows`, to convert the directed flow graph to an undirected version amenable to visualization. The result may be visualized with the `dodgr\_flowmap` function.

FINDINGS

The following code demonstrates the ease of distance calculations with dodgr:

The result is a matrix of 1,000 x 1,000 distances ranging up to 90 km. This calculation of 1 million pairwise distances takes only 1.5 seconds, while equivalent calculation with igraph (Csardi and Nepusz 2006) takes approximately 5 times longer. For comparison, the Google Distance Matrix API (Google 2018) currently costs US$0.004 per call (although generally for only up to 500,000 calls), translating to US$4,000 for 2 seconds of calculation.

Flow aggregation is implemented through the following straightforward extensions of the previously shown code, with the function taking only 4 seconds for aggregation of 1 million pairwise flows. The final line generates a plot of the aggregated flows (Figure 1).

Acknowledgments

This work was supported in part through the Austrian FWF (Federal Science Fund) Project P29135.

```
                          contract = TRUE) # contract graph prior to aggregation
dodgr_flowmap (merge_directed_flows (f), linescale = 5)
```

Code Figure 3

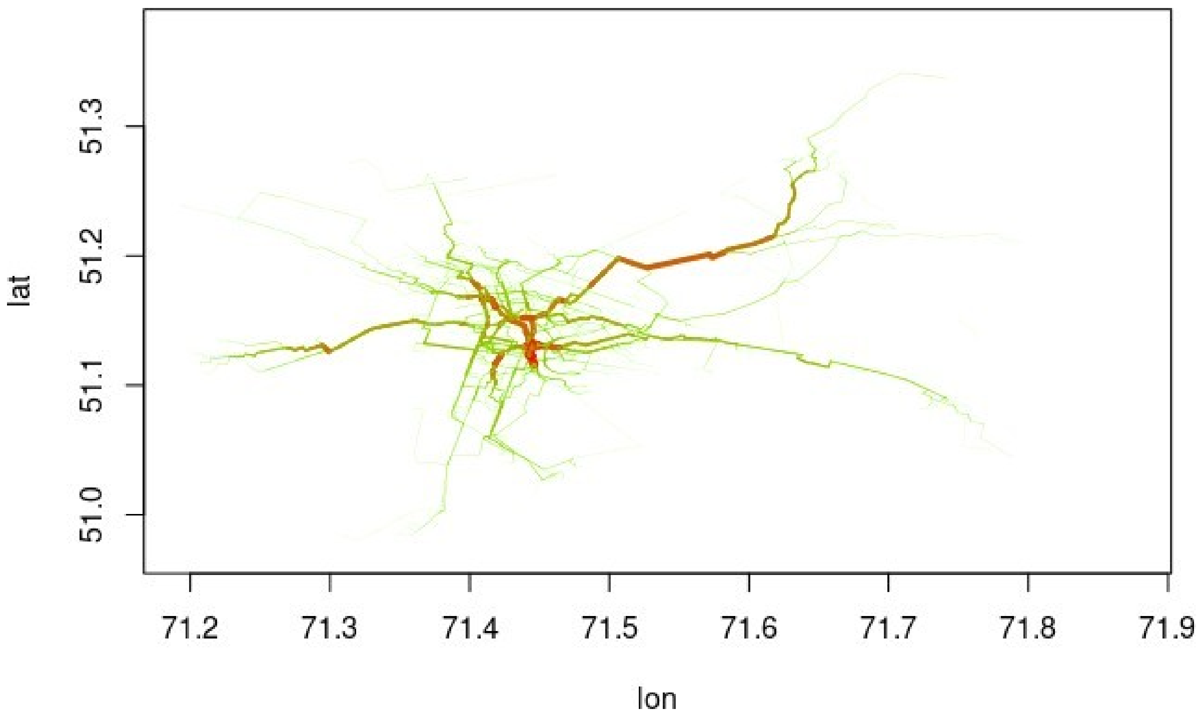


Figure 1: Flows Aggregated Over 1 Million Random Journeys Through the City of Astana, Kazakhstan

REFERENCES

Aric, A.H., D.A. Schult, and P.J. Swart. 2008. "Exploring Network Structure, Dynamics, and Function Using Networkx." In *Proceedings of the 7th Python in Science Conference (Scipy2008)*, edited by Gäel Varoquaux Travis Vaught and J. Millman, 11–15. Pasadena, CA USA.

Boeing, Geoff. 2017. "OSMnx: New Methods for Acquiring, Constructing, Analyzing, and Visualizing Complex Street Networks." *Computers, Environment and Urban Systems* 65 (September): 126–39. https://doi.org/10.1016/j.compenvurbsys.2017.05.004.

Csardi, G., and T. Nepusz. 2006. "The Igraph Software Package for Complex Network Research." *InterJournal Complex Systems*. http://igraph.org.

Google. 2018. "Distance Matrix Api." 2018. https://developers.google.com/maps/documentation/distance-matrix.

GRASS Development Team. 2017. "Geographic Resources Analysis Support System (Grass Gis) Software, Version 7.2." *Open Source Geospatial Foundation*. https://grass.osgeo.org/grass64/manuals/v.rast.stats.html.

Padgham, Mark, Robin Lovelace, Maëlle Salmon, and Bob Rudis. 2017. "Osmdata." *The Journal of Open Source Software* 2 (14): 305. https://doi.org/10.21105/joss.00305.

Ratti, Carlo, Stanislav Sobolevsky, Francesco Calabrese, Clio Andris, Jonathan Reades, Mauro Martino, Rob Claxton, and Steven H. Strogatz. 2010. "Redrawing the Map of Great Britain from a Network of Human Interactions." Edited by Olaf Sporns. *PLoS ONE* 5 (12): e14248. https://doi.org/10.1371/journal.pone.0014248.