The Stata Journal

The *Stata Journal* publishes reviewed papers together with shorter notes or comments, regular columns, book reviews, and other material of interest to Stata users. Examples of the types of papers include 1) expository papers that link the use of Stata commands or programs to associated principles, such as those that will serve as tutorials for users first encountering a new field of statistics or a major new technique; 2) papers that go "beyond the Stata manual" in explaining key features or uses of Stata that are of interest to intermediate or advanced users of Stata; 3) papers that discuss new commands or Stata programs of interest either to a wide spectrum of users (e.g., in data management or graphics) or to some large segment of Stata users (e.g., in survey statistics, survival analysis, panel analysis, or limited dependent variable modeling); 4) papers analyzing the statistical properties of new or existing estimators and tests in Stata; 5) papers that could be of interest or usefulness to researchers, especially in fields that are of practical importance but are not often included in texts or other journals, such as the use of Stata in managing datasets, especially large datasets, with advice from hard-won experience; and 6) papers of interest to those who teach, including Stata with topics such as extended examples of techniques and interpretation of results, simulations of statistical concepts, and overviews of subject areas.

The *Stata Journal* is indexed and abstracted by *CompuMath Citation Index*, *Current Contents/Social and Behavioral Sciences*, *RePEc: Research Papers in Economics*, *Science Citation Index Expanded* (also known as *SciSearch*), *Scopus*, and *Social Sciences Citation Index*.

For more information on the *Stata Journal*, including information for authors, see the webpage

http://www.stata-journal.com

Subscriptions are available from StataCorp, 4905 Lakeway Drive, College Station, Texas 77845, telephone 979-696-4600 or 800-782-8272, fax 979-696-4601, or online at

http://www.stata.com/bookstore/sj.html

Subscription rates listed below include both a printed and an electronic copy unless otherwise mentioned.

U.S. and Canada		Elsewhere	
Printed & electronic		**Printed & electronic**	
1-year subscription	$115	1-year subscription	$145
2-year subscription	$210	2-year subscription	$270
3-year subscription	$285	3-year subscription	$375
1-year student subscription	$ 85	1-year student subscription	$115
1-year institutional subscription	$345	1-year institutional subscription	$375
2-year institutional subscription	$625	2-year institutional subscription	$685
3-year institutional subscription	$875	3-year institutional subscription	$965
Electronic only		**Electronic only**	
1-year subscription	$ 85	1-year subscription	$ 85
2-year subscription	$155	2-year subscription	$155
3-year subscription	$215	3-year subscription	$215
1-year student subscription	$ 55	1-year student subscription	$ 55

Back issues of the *Stata Journal* may be ordered online at

http://www.stata.com/bookstore/sjj.html

Individual articles three or more years old may be accessed online without charge. More recent articles may be ordered online.

http://www.stata-journal.com/archives.html

The *Stata Journal* is published quarterly by the Stata Press, College Station, Texas, USA.

Address changes should be sent to the *Stata Journal*, StataCorp, 4905 Lakeway Drive, College Station, TX 77845, USA, or emailed to sj@stata.com.

The Stata Journal (2016)
16, Number 2, pp. 416–423

Calculate travel time and distance with OpenStreetMap data using the Open Source Routing Machine (OSRM)

Stephan Huber
University of Regensburg
Regensburg, Germany
stephan.huber@wiwi.uni-regensburg.de

Christoph Rust
University of Regensburg
Regensburg, Germany
christoph.rust@stud.uni-regensburg.de

Abstract. In this article, we introduce the `osrmtime` command, which calculates the distance and travel time between two points using latitude and longitude information. The command uses the Open Source Routing Machine (OSRM) and OpenStreetMap to find the optimal route by car, by bicycle, or on foot. The procedure is specially built for large georeferenced datasets. Because it is fast, the command uses the full computational capacity of a PC, allows the user to make unlimited requests, and is independent of the Internet and commercial online providers. Hence, there is no risk of the command becoming obsolete. Moreover, the results can be replicated at any time.

Keywords: dm0088, osrmtime, osrmprepare, mqtime, traveltime3, OSRM, OpenStreetMap, Google Maps, MapQuest, geospatial analysis, ArcGIS, travel time, travel distance, public road network

1 Introduction

The increased availability of large georeferenced datasets for scientific purposes calls for an efficient method to calculate the distances between subjects and the time it takes to travel from A to B. In this article, we introduce the `osrmtime` command, which uses geographic data on latitudes and longitudes to determine the travel time and the distance between two points. In contrast to existing commands like `globaldist`, `vincenty`, `geodist`, or `sphdist`, which compute geodetic distances, `osrmtime` calculates the travel time and distance to find the optimal route over public roads by car, by bicycle, or on foot. This platform-independent method (Windows, Mac, Linux) is innovative because it allows the user to calculate an unlimited number of requests, and it works offline, which ensures that the results can be replicated. Moreover, `osrmtime` works efficiently. It can calculate thousands of requests within seconds,[1] because it is multiprocessor capable and uses the Open Source Routing Machine (OSRM).[2] The OSRM

1. In an example, we calculated the distance and the travel time between 826,256 pairwise combinations of German hospitals. The calculation took about 49 minutes, which is about 280 requests per second on a system with 16 GB RAM and an Intel i7-2600 3.40 GHz CPU.
2. For more information, see Luxen and Vetter (2011) and http://project-osrm.org/.

is a high-performance open-source C++ routing engine that indicates the shortest routes on public roads and runs with open-source maps from OpenStreetMap.[3]

The program's independence from the Internet and commercial providers has some advantages. First, georeferenced data often contain sensitive data, and their rules of use often forbid using an Internet connection because of either legally binding constraints or a nondisclosure agreement.

Second, and probably most important, an offline procedure that uses only open-source software ensures that the results can be replicated at any time and carries no risk of the command becoming obsolete—as was the case with `traveltime` (Ozimek and Miles 2011), `traveltime3`,[4] and `mqtime` (Voorheis 2015). These earlier programs calculated travel time and distance using the application programming interface (API)[5] from commercial providers via the Internet. Third-party providers, however, can change their APIs or their terms of use; thus user-written commands can become obsolete. The `traveltime` command, for example, was created to use the Google Maps API v.2. Unfortunately, this API is now obsolete; therefore, so is `traveltime`. Although Stefan Bernhard adjusted `traveltime` to work with the up-to-date Google Maps API v.3, his program `traveltime3` is no longer available because Google changed its restrictions on using the Distance Matrix API.[6] The most recent approach by Voorheis (2015) suffers a combination of both problems. His command, `mqtime`, was created to use the API of the commercial provider MapQuest to calculate travel time and distance for an unlimited number of requests by using OpenStreetMap. Unfortunately, MapQuest restructured its API licensing, dramatically cutting the number of requests that `mqtime` can process. Hence, `mqtime` no longer works, and John Voorheis has ceased to maintain the command.

Third, unlike approaches that use online-mapping services, our approach is not based on real-time data. Although a real-time calculation is sometimes desired, researchers often want to know the travel time and distance at a certain point in time. Furthermore, they frequently want results that can be replicated at any time. Neither is really possible when using real-time data from online services, because the results are a function of time-specific circumstances. For example, if you use georeferenced data from 2013, you will probably not want to calculate the travel time and the distance on a Monday morning in late 2015 during rush hour. In turn, it would probably be misleading to use the resulting travel-time data to explain economic behavior in 2013.

`osrmtime` implements two tools: the OSRM and OpenStreetMap. Both are provided by the open-source community, which offers some advantages but also a few disadvantages. One advantage is that both tools can be downloaded, used, spread, and adjusted without restrictions, which gives the user full control over the software. One disadvantage is that the maps provided by OpenStreetMap are not validated by a general authority like the maps of a commercial provider but are recorded and maintained

3. For more information, see http://www.openstreetmap.org.
4. The user-written code by Stefan Bernhard is no longer available. For further information, please email stefanbernhard88@gmail.com.
5. An API provides source code-based facilities to develop applications for a system in a given programming language.
6. See https://developers.google.com/maps/documentation/distancematrix/.

by users in a decentralized fashion. However, this does not necessarily devalue OpenStreetMap, because the quality of both ways of recording and updating geographical data is subject to criticism. Commercial providers record and maintain geographical information more intensively for regions that are most profitable in sales, whereas the quality of geographical information from open sources is a function of the effort of users in a given region. Therefore, regions with a lively community probably have better maps than regions with only a few active users. Overall, OpenStreetMap is used heavily in scientific research, as Arsanjani et al. (2015) show in their overview.

In the following section, we describe how to install the OSRM with all its dependencies. In section 3, we explain the `osrmtime` command. In section 4, we illustrate its use. In section 5, we conclude by comparing it with ArcGIS.

2 Prerequisites

`osrmtime` calculates the travel time and distance from a point of origin to a point of destination using the high-performance routing open-source software, OSRM. `osrmtime` automatically starts the OSRM from the hard disk and performs the calculation using an extract from OpenStreetMap, which needs to be saved on the hard disk. To use `osrmtime`, your system must support a 64-bit architecture (for example, Windows 7 or later). Some files from the Microsoft Visual C++ Redistributable package must also be installed. In the next section, we describe this installation procedure.

2.1 Install the files

`osrmtime` uses the OSRM and some files from the Microsoft Visual C++ Redistributable package. Both must be installed on your system to run `osrmtime`. The installation can be done manually or automatically.

Automatic

```
. net install osrmtime, from("http://www.uni-regensburg.de/
> wirtschaftswissenschaften/vwl-moeller/medien/osrmtime")
. net get osrmtime, from("http://www.uni-regensburg.de/
> wirtschaftswissenschaften/vwl-moeller/medien/osrmtime")
. shell osrminstall.cmd
```

Manual

1. Copy the ado-files `osrmtime.ado`, `osrmprepare.ado`, and `osrminterface.ado` into your `PERSONAL` ado-folder.

2. Install the recent Microsoft Visual C++ Redistributable package for Visual Studio 2015.[7]

7. See https://www.microsoft.com/en-us/download/details.aspx?id=48145.

3. Install the OSRM by downloading[8] and unpacking the OSRM executables to a folder of your choice in which Stata has write access, for example, `C:/osrm/`.

Note that implementing the OSRM is different on Linux and Mac OS X systems. For instructions on how to build the OSRM on these systems, see https://github.com/Project-OSRM/osrm-backend/wiki/Building%20OSRM.

2.2 Prepare maps with osrmprepare

To use `osrmtime`, you must download at least one map covering the region of interest and prepare it for routing. This is necessary for several reasons. Most importantly, raw OpenStreetMap data also include information that are not relevant for routing, such as public toilets or memorials. The preparation ensures that only relevant information is extracted and that this information can be used efficiently by the OSRM. We offer the `osrmprepare` command to execute all necessary steps automatically. The execution speed for `osrmprepare` depends on the size of your map and the capacity of your system.[9] Note that you have to prepare your map only once. The prepared map can be used as often as you like. To update your map, however, you have to download a more recent map and prepare it again.

The following steps explain how to proceed:

1. Download an OpenStreetMap data file in the `osm.pbf` format to a folder of your choice, for example, `C:/mymaps/mymap.osm.pbf`.[10]

2. Prepare a map for routing. To make this step easier for the user, we wrote the `osrmprepare` command. Install the command and use it as explained below.

Syntax of osrmprepare

`osrmprepare, mapfile(`*pbf_path*`)` [`osrmdir(`*path*`) diskspace(`# `MB)`
`profile(`*speed_profile*`)`]

Options of osrmprepare

`mapfile(`*pbf_path*`)` specifies the location of the downloaded map file from OpenStreetMap in *`.osm.pbf` format, for example,
`mapfile("C:/mymap/examplemap.osm.pbf")`. `mapfile()` is required.

8. See http://www.uni-regensburg.de/wirtschaftswissenschaften/vwl-moeller/medien/osrmtime/osrm.zip.
9. For instance, it takes about 27 minutes to extract a map for Germany (about 2.6 GB) on a system with 16 GB RAM with an Intel i7-2600 3.40 GHz CPU.
10. Maps can be downloaded, for example, from http://download.geofabrik.de.

`osrmdir(`*path*`)` specifies the path in which the OSRM executables are saved. The default is `osrmdir("C:/osrm/")` for Windows and `osrmdir("/usr/local/osrm/")` for Linux.

`diskspace(`*#* `MB)` specifies the allocation of disk space for preparation. The default is `diskspace(5000 MB)`. If your system cannot allocate 5,000 MB, you must adjust this number here; otherwise, the command will not work.

`profile(`*speed_profile*`)` specifies to prepare a map that contains the routes for traveling by car, by bicycle, or on foot. *speed_profile* can be `car`, `bicycle`, or `foot`.

3 The osrmtime command

3.1 Syntax

`osrmtime` *latitude1 longitude1 latitude2 longitude2* [`, mapfile(`*osrm_path*`)` `osrmdir(`*path*`)` `nocleanup` `threads(`*#*`)` `servers(`*#*`)` `ports(`*numlist*`)`]

latitude1, *longitude1*, *latitude2*, and *longitude2* are numeric variables, denoted in decimal degrees.[11] They contain the starting point (*latitude1 longitude1*) and the destination (*latitude2 longitude2*) in a system of coordinates.

3.2 Options

`mapfile(`*osrm_path*`)` specifies the location of the *`*.osrm`* file format map, for example, `mapfile("C:/mymap/examplemap.osrm")`. This file can be extracted by using the `osrmprepare` command as explained above.

`osrmdir(`*path*`)` specifies the path in which the OSRM binary (see step 1 of preparation) is saved. The default is `osrmdir("C:/osrm/")` for Windows and `osrmdir("/usr/local/osrm/")` for Linux.

`nocleanup` indicates to keep temporary files that are generated during the process and prevents the OSRM from being shut down. This can speed up the calculation if `osrmtime` is used consecutively with the same map, because `osrmtime` does not need to shut down and start the OSRM over and over again.

Advanced users with large datasets can optimize the parallel computing to speed up calculation on their system by using the following options: `threads(`*#*`)` specifies the number of parallel Stata threads per running OSRM instance, the default value being `4`; `servers(`*#*`)` starts several instances of the OSRM—at least if your system permits, the default being `1`; `ports(`*numlist*`)` resolves problems with used TCP ports by manually specifying the port to use, the default being `5000`.

11. We use the standard coordinate system in its latest revision, World Geodetic System (WGS 84). It also works as the reference coordinate system of the Global Positioning System (GPS).

3.3 Description

osrmtime provides an interface to the free high-performance OSRM. This enables the calculation of travel time and distance from a point of origin to a point of destination in Stata. Provided that the OSRM is already installed on your system and you already have prepared your map of interest, osrmtime automatically starts the OSRM and performs the calculation. osrmtime already implements parallel computation, so the time for calculating shortest distances can be reduced significantly depending on your system.

osrmtime generates the following five variables:

- distance: the distance of the shortest route in meters
- duration: the travel time of the shortest route in seconds
- jumpdist1: the (spheric) distance between the specified input location (origin) and a matched location to the road network in meters
- jumpdist2: the (spheric) distance between the specified input location (destination) and a matched location to the road network in meters
- return_code: 0 ⇒ everything is fine; 1 ⇒ no route was found by the OSRM with points specified; 2 ⇒ the OSRM did not respond; and 3 ⇒ something else went wrong.

Note that large values for jumpdist1 or jumpdist2 can be a signal that the map is incomplete, meaning that an existing street is not listed in the map. Hence, we recommend to check the length of both jump distances, especially because the jump distance is not considered in the travel-time calculation, which means that large jump distances can yield an underestimated travel time. One way to solve this problem, for example, is to assign a certain number of seconds per meter that it takes to travel the jump distances and add this time to the travel time.

Advanced users can manipulate the routing using the OSRM in various ways. It is possible, for instance, to exclude certain kinds of roads or to adjust the speed profile (for example, change the maximum speed allowed on highways). Moreover, the map file from OpenStreetMap itself can be manipulated.

4 Example

The following results exemplify how osrmtime and osrmprepare can be used. In the example, we calculate the travel time and distance from Alexanderplatz in Berlin to 3,374 restaurants also located in Berlin.

```
. *download the map of Berlin
. capture mkdir mymaps
. copy "http://download.geofabrik.de/europe/germany/berlin-latest.osm.pbf"
> "mymaps/berlin.osm.pbf", replace
(note: file mymaps/berlin.osm.pbf not found)
```

```
. *prepare the map (this takes some time ~2 minutes, depending on your system):
. osrmprepare, mapfile("mymaps/berlin.osm.pbf") osrmdir("C:\osrm\") profile(car)

. *open coordinates of restaurants in Berlin
. discard

. import delimited "http://www.uni-regensburg.de/wirtschaftswissenschaften/
> vwl-moeller/medien/osrmtime/restaurants_berlin.csv", delimiter(";") clear
(4 vars, 3,374 obs)

. *add destination Alexanderplatz
. generate lat_alex = 52.5219184

. generate lon_alex = 13.4132147

. list in 1/3

     +-------------------------------------------------------------------------+
     |      lon        lat     osm_id           name   lat_alex   lon_alex |
     |-------------------------------------------------------------------------|
  1. | 13.32283   52.50691   26735749           Aida   52.52192   13.41321 |
  2. | 13.31732   52.50624   26735760   La Forneria   52.52192   13.41321 |
  3. | 13.32078   52.50734   26735763         Sakana   52.52192   13.41321 |
     +-------------------------------------------------------------------------+

. * calculate travel time and distances:
. osrmtime lat lon lat_alex lon_alex, mapfile("mymaps/berlin.osrm")
> osrmdir("C:\osrm\")
------------------------------------------------------------

Traveltime and Distance with OSRM

Check for running OSRM: not running!
Starting OSRM           now running!
Writing do-files:       done!
Partitioning datasets:  done!
Calculating:
0%---10%---20%---30%--40%---50%---60%--70%---80%---90%--100%

finished calculation!
------------------------------------------------------------

. summarize

    Variable |        Obs        Mean    Std. Dev.       Min        Max
-------------+---------------------------------------------------------
         lon |      3,374    13.37962    .0867841   13.09485   13.75691
         lat |      3,374    52.50509    .0399598   52.35291   52.66253
      osm_id |      3,374    1.54e+09    1.08e+09   2.67e+07   3.80e+09
        name |          0
    lat_alex |      3,374    52.52192           0   52.52192   52.52192
-------------+---------------------------------------------------------
    lon_alex |      3,374    13.41321           0   13.41321   13.41321
    distance |      3,374    7797.189    5203.881        287      31190
    duration |      3,374    618.3402    380.2277         28       2469
   jumpdist1 |      3,374    17.75756    16.54522          0        292
   jumpdist2 |      3,374         103           0        103        103
-------------+---------------------------------------------------------
 return_code |      3,374           0           0          0          0
```

```
. list name distance duration jumpdist1 jumpdist2 in 1/3
```

	name	distance	duration	jumpdi~1	jumpdi~2
1.	Aida	7360	612	18	103
2.	La Forneria	7710	634	8	103
3.	Sakana	7416	618	11	103

5 Conclusion

In this article, we introduced a fast procedure to calculate travel time and distance using public roads by car, by bicycle, and on foot. This kind of geographic information is fundamental to regional sciences and can be applied to empirical research in various subjects, including economics, sociology, and epidemiology. `osrmtime` has advantages over other offline routing software. The high-end mapping software ArcGIS, for example, also allows the user to calculate the travel time and distance but has some drawbacks compared with `osrmtime`. First, the Network Analyst Extension required is costly. Second, the routing algorithm works less efficiently than the OSRM. Third, ArcGIS does not have a tool that easily allows the user to calculate hundreds of requests. Thus the processing of many requests requires experience with Python. In a previous project, we succeeded in calculating thousands of routing requests using ArcGIS on a cluster of eight PCs. However, when calculating the same requests with one PC and `osrmtime`, we find that ArcGIS is outperformed by a factor of at least 100.

6 References

Arsanjani, J. J., A. Zipf, P. Mooney, and M. Helbich, eds. 2015. *OpenStreetMap in GIScience: Experiences, Research, and Applications*. Cham, Switzerland: Springer.

Luxen, D., and C. Vetter. 2011. Real-time routing with OpenStreetMap data. In *Proceedings of the 19th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, 513–516. New York: Association for Computing Machinery.

Ozimek, A., and D. Miles. 2011. Stata utilities for geocoding and generating travel time and travel distance information. *Stata Journal* 11: 106–119.

Voorheis, J. 2015. mqtime: A Stata tool for calculating travel time and distance using MapQuest web services. *Stata Journal* 15: 845–853.

About the authors

Stephan Huber and Christoph Rust are research assistants of Joachim Möller at the University of Regensburg. Huber is also a doctoral candidate at the University of Trier. His thesis is about disaggregated international bilateral trade flows and the impact of FDI and international trade on economic development.