**osrmtime: 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.** This article introduces the Stata program `osrmtime`, which allows the calculation of distance and travel time between two points using latitude and longitude information. The command takes advantage of the *Open Source Routing Machine* (OSRM) and of *OpenStreetMaps* in order to find the optimal route by car, bicycle, or foot. The procedure is specially built for large georeferenced data sets, as it is fast, uses the full computational capacity of the 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:** `mgetime`, `traveltime3`, OSRM, OpenStreetMap, Google Maps, Mapquest, geospatial analysis, ArcGIS, travel time, travel distance, public road network

**Note:** The most recent Stata ado-files and help-file can be found here: [http://www.uni-regensburg.de/wirtschaftswissenschaften/vwl-moeller/forschung/index.html](http://www.uni-regensburg.de/wirtschaftswissenschaften/vwl-moeller/forschung/index.html)


1 **Introduction**

The increased availability of large georeferenced data sets 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. This paper proposes the `osrmtime` command to do this in Stata. `osrmtime` uses geographic data on latitudes and longitudes to determine the travel time and the distance between two points in space. In contrast to existing Stata programs like `globaldist`, `vincenty`, `geodist`, or `sphdist`, which compute geodetic distances, `osrmtime` calculates the travel time and distances using the public road network by car, bicycle, or on foot. This platform-independent method (Windows, MacOS, 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. In an example we calculated the distance and the travel time between 826,256 pairwise combinations of German hospitals. The calculation takes about 49 minutes, which is about 280 requests per
processor capable, and takes advantage of OSRM (*Open Source Routing Machine*\(^2\)) OSRM is a high-performance open-source C++ routing engine for shortest routes on road networks that runs with open-source maps from *OpenStreetMaps*\(^3\).

The independence from the internet and commercial providers has some advantages: First, georeferenced data often contains sensitive data, and their rules of use often forbid the usage of an internet connection, either due to legally binding constraints, or a non-disclosure agreement.

Second, and probably most important, an offline procedure which uses open source software only ensures that the results can be replicated at any time, and carries no risk on the command becoming obsolete—as was the case with *traveltime* (Ozimek and Miles\(^2\)), *traveltime*\(^4\), and *mqtime* (Voorheis\(^5\)). These earlier programs calculated travel time and distances using the Application Programming Interface (API)\(^5\) from commercial providers via the internet. APIs from third-party providers, however, can change their interface or their terms of use, and hence Stata programs can become obsolete. The command *traveltime*, for example, was built to use the Google Maps API v.2. Unfortunately, this API is now obsolete, and therefore is *traveltime*. Although Stefan Bernhard adjusted *traveltime* in order to work with the up-to-date Google Maps API v.3, his program *traveltime*\(^3\) is no longer available as Google changed their restrictions on the use of their Distance Matrix API\(^6\). The most recent approach by Voorheis (2015) suffers a combination of both problems: His command *mqtime* was also built to take advantage of the API of the commercial provider Mapquest in order to calculate travel time and distances for an unlimited number of requests using *OpenStreetMaps*. Unfortunately, Mapquest restructured their API licensing, dramatically cutting the number of requests which *mqtime* can process. Hence, *mqtime* is no longer working, and John Voorheis has ceased to maintain the program.

Third, in contrast to 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 like to know the travel time and distance at a certain point in time. Furthermore, they frequently like to have results which can be replicated at any time. Neither is really possible using real-time data from online services, since the results are a function of time-specific circumstances. If you use georeferenced data from 2013, for example, 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.

Our program *osrntime* basically takes advantage of two tools: the *Open Source Routing Machine* (OSRM), and *OpenStreetMaps*. Both are provided by the Open Source

---

2. For more information, we refer to Luxen and Vetter (2011), and http://project-osrm.org/.
3. For more information, see www.openstreetmap.org.
4. The user-written code by Stefan Bernhard is no longer available. For further questions, please refer to stefanbernhard888@gmail.com.
5. An API provides source code-based facilities to develop applications for a system in a given programming language.
Community, which brings some advantages, but also a few disadvantages. One positive aspect worth mentioning is that both tools can be downloaded, used, spread, and adjusted without restrictions, which gives the Stata user full control over the software. One negative aspect is that the maps provided by OpenStreetMaps are not validated by a general authority like the maps of a commercial provider, but are recorded and maintained by users in a decentralized fashion. However, this does not necessarily take the shine out of OpenStreetMaps, since 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 which are most profitable in regard to sales, whereas the quality of geographical information from open sources is a function of the effort the users exert in a given region. That means that regions with a lively community probably have better maps than regions with only a few active users. Overall, OpenStreetMaps are used heavily in scientific research, as Arsanjani et al. (2015) show in their overview.

The following section describes how to install OSRM with all its dependencies. Section 3 explains the osrmtime command, section 4 exemplifies the direction of use, and section 5 concludes with a comparison to ArcGIS.

2 Prerequisites

osrmtime calculates travel time and distances from a point of origin to a point of destination using the high-performance routing open-source software OSRM. osrmtime automatically starts OSRM from the hard disk and performs the calculation using an extract of OpenStreetMaps, which needs to be saved on the hard-disk. In order to get osrmtime working, your system must support a 64-bit architecture (e.g. Windows 7 or later). Additionally, some files have to be installed. The next section describes this installation procedure.

2.1 Install the routing software

osrmtime uses some files from the Microsoft Visual C++ Redistributable, and the routing software OSRM. Both must be installed on your system in order for you 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
```
osrmtime: Calculate Travel Time and Distance

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 for Visual Studio 2015.

3. Install the routing software OSRM by downloading and unpacking the OSRM executables to a folder of your choice in which stata has write access, e.g. C:/osrm/.

Please note: The implementation of OSRM is different on Linux and MacOSX systems. For instructions on how to build OSRM on your Linux or MacOSX system, see: https://github.com/Project-OSRM/osrm-backend/wiki/Building%20OSRM.

2.2 Prepare Maps with osrmprepare

In order to use osrmtime, at least one map covering the region of interest must be downloaded and prepared for routing. This is necessary for several reasons. Most importantly, raw OpenStreetMap data also includes information that is 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 in an efficient way by the routing machine OSRM. We offer the osrmprepare command to execute all necessary steps automatically. Depending on the size of your map and the capacity of your system, the execution of osrmprepare takes some time. Please note that you only have to prepare your map once. The prepared map can be used as often as you like. If you wish 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, e.g. C:/mymaps/mymap.osm.pbf.

2. Prepare a map for routing. In order 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("C:/mymap/examplemap.osm.pbf") [ 
    osrmdir("C:/osrm/") diskspace(# MB) profile(speed_profile) ]
```

---

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 3.40GHz CPU.
10. Maps can be downloaded here: http://download.geofabrik.de, for example.
mapfile() declares the location of the downloaded OpenStreetMap in *.osm.pbf format. osrmdir() announces the path in which the OSRM executables is saved (the default is "C:/osrm/"). profile can be either car, bicycle, or foot to prepare a map that contains either the routes for car, bicycle, or foot. The option diskspace allows to allocate disk space for preparation, default is 5000 MB. If your system is not able to allocate 5000 MB you must adjust here, otherwise it is not working.

3 The osrmtime command

3.1 Syntax and Description

```
osrmtime latitude1 longitude1 latitude2 longitude2,
    mapfile("C:/mymap/examplemap.osrm") [osrmdir("C:/osrm/") nocleanup
    threads(#) servers(#) ports(numlist)]
```

latitude1, longitude1, latitude2, and longitude2 are numeric variables, denoted in decimal degree. They contain the starting point (latitude1 longitude1), and the destination (latitude2 longitude2) in a system of coordinates. In mapfile() you have to announce the location of the map of interest in the *.osrm-file format. This file can be extracted by using osrmprepare as explained above. The osrmdir option enables the user to determine the path in which the OSRM binary (see step 1 of preparation) is saved. Setting nocleanup keeps temporary files which are generated during the process, and prevents OSRM from being shut down. This can speed up the calculation if osrmtime is used in a consecutive fashion with the same map, because osrmtime does not need to shut down and start OSRM over and over again.

Advanced users with very large data sets can optimize the parallel computing in order to speed up calculation on their system further 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 OSRM—at least if your system permits, the default being 1; ports encounters problems with used TCP ports by manually specifying the port to use, the default being 5000.

osrmtime generates five variables:

distance distance of the shortest route in meters

duration travel time of the shortest route in seconds

jumpdist1 (spheric) distance between specified input location (origin) and matched location to road network in meters

jumpdist2 (spheric) distance between specified input location (destination) and matched location to road network in meters

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).
osrmtime: Calculate Travel Time and Distance

return_code 0 ⇒ everything is fine; 1 ⇒ no route was found by OSRM with points specified; 2 ⇒ OSRM did not respond; 3 ⇒ something else went wrong.

Please 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 since the ‘jump distance’ is not considered in the travel time calculation. That means, 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 via OSRM in various ways. It is possible, for instance, to exclude certain kind of roads, or to adjust the speed profile (e.g.: change the maximum speed allowed on highways). Moreover, the OpenStreetMap itself can be manipulated.

4 Example

The following log exemplifies how osrmt ime and osrmprepare can be used. In the example, we calculate the travel time and distance from Alexanderplatz in Berlin to 3374 restaurants also located in Berlin.

```stata
. *download the map of Berlin
  . cap mkdir mymaps
  . copy "http://download.geofabrik.de/europe/germany/berlin-latest.osm.pbf" "mymaps/berlin.osm.pbf", replace
  . *prepare the map (this takes some time ~2 minutes, depending on your system):
  . osrmprepare, mapfile("mymaps/berlin.osm.pbf") osrmdir("C:\osrm\") profile(ca > r)
  . *open open coordinates of restaurants in Berlin
  . discard
  . insheet using "http://www.uni-regensburg.de/wirtschaftswissenschaften/vwl-moe > ller/medien/osrmtime/restaurants_berlin.csv", delimiter(;) clear
    (4 vars, 3374 obs)
  . *add destination Alexanderplatz
  . gen lat_alex = 52.5219184
  . gen lon_alex = 13.4132147
  . list in 1/3

<table>
<thead>
<tr>
<th>lon</th>
<th>lat</th>
<th>osm_id</th>
<th>name</th>
<th>lat_alex</th>
<th>lon_alex</th>
</tr>
</thead>
<tbody>
<tr>
<td>13.32283</td>
<td>52.50691</td>
<td>26735749</td>
<td>Aida</td>
<td>52.52192</td>
<td>13.41321</td>
</tr>
<tr>
<td>13.31732</td>
<td>52.50624</td>
<td>26735760</td>
<td>La Forneria</td>
<td>52.52192</td>
<td>13.41321</td>
</tr>
<tr>
<td>13.32078</td>
<td>52.50734</td>
<td>26735763</td>
<td>Sakana</td>
<td>52.52192</td>
<td>13.41321</td>
</tr>
</tbody>
</table>

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

Travel time 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!

<table>
<thead>
<tr>
<th>Variable</th>
<th>Obs</th>
<th>Mean</th>
<th>Std. Dev.</th>
<th>Min</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>lon</td>
<td>3374</td>
<td>13.37962</td>
<td>.0867841</td>
<td>13.09485</td>
<td>13.75691</td>
</tr>
<tr>
<td>lat</td>
<td>3374</td>
<td>52.50509</td>
<td>.0399598</td>
<td>52.35291</td>
<td>52.66253</td>
</tr>
<tr>
<td>osm_id</td>
<td>3374</td>
<td>1.54e+09</td>
<td>1.08e+09</td>
<td>2.67e+07</td>
<td>3.80e+09</td>
</tr>
<tr>
<td>name</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>lat_alex</td>
<td>3374</td>
<td>52.52192</td>
<td>0</td>
<td>52.52192</td>
<td>52.52192</td>
</tr>
<tr>
<td>lon_alex</td>
<td>3374</td>
<td>13.41321</td>
<td>0</td>
<td>13.41321</td>
<td>13.41321</td>
</tr>
<tr>
<td>distance</td>
<td>3374</td>
<td>7793.33</td>
<td>5198.176</td>
<td>287</td>
<td>30133</td>
</tr>
<tr>
<td>duration</td>
<td>3374</td>
<td>618.0628</td>
<td>379.8858</td>
<td>28</td>
<td>2416</td>
</tr>
<tr>
<td>jumpdist1</td>
<td>3374</td>
<td>17.78008</td>
<td>16.58378</td>
<td>0</td>
<td>292</td>
</tr>
<tr>
<td>jumpdist2</td>
<td>3374</td>
<td>103</td>
<td>0</td>
<td>103</td>
<td>103</td>
</tr>
<tr>
<td>return_code</td>
<td>3374</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

. list name distance duration jumpdist1 jumpdist2 in 1/3

<table>
<thead>
<tr>
<th>name</th>
<th>distance</th>
<th>duration</th>
<th>jumpdist1</th>
<th>jumpdist2</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Aida</td>
<td>7361</td>
<td>612</td>
<td>18</td>
<td>103</td>
</tr>
<tr>
<td>2. La Forneria</td>
<td>7711</td>
<td>634</td>
<td>8</td>
<td>103</td>
</tr>
<tr>
<td>3. Sakana</td>
<td>7417</td>
<td>618</td>
<td>11</td>
<td>103</td>
</tr>
</tbody>
</table>

5 Conclusion

This article introduces a fast procedure to calculate traveling times and distances using the transport network by car, bicycle, 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 even 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 to osrmtime. First, the Network Analyst Extension required is costly. Second, the routing algorithm works less efficiently than 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, doing the same requests with one PC and osrmtime, ArcGIS is outperformed by a factor of at least 100.
6 References


About the authors

Stephan Huber and Christoph Rust are research assistants of Joachim Möller at the University of Regensburg. Stephan Huber is also 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.