



The World's Largest Open Access Agricultural & Applied Economics Digital Library

This document is discoverable and free to researchers across the globe due to the work of AgEcon Search.

Help ensure our sustainability.

Give to AgEcon Search

AgEcon Search

<http://ageconsearch.umn.edu>

aesearch@umn.edu

*Papers downloaded from **AgEcon Search** may be used for non-commercial purposes and personal study only. No other use, including posting to another Internet site, is permitted without permission from the copyright owner (not AgEcon Search), or as allowed under the provisions of Fair Use, U.S. Copyright Act, Title 17 U.S.C.*

No endorsement of AgEcon Search or its fundraising activities by the author(s) of the following work or their employer(s) is intended or implied.

cntraveltime: Travel distance and travel time in China

Xueren Zhang
School of Economics and Management
Wuhan University
Wuhan, China
snowmanzhang@whu.edu.cn

Yuan Xue
School of Management
Huazhong University of Science and Technology
Wuhan, China
xueyuan19920310@163.com

Chuntao Li
School of Economics
Henan University
Kaifeng, China
chtl@henu.edu.cn

Abstract. In this article, we introduce the command `cntraveltime`, which can calculate both the travel distance and the travel time between two locations in China with respect to different modes of transportation (driving, public transport, and cycling). Existing commands such as `georoute`, `traveltime`, and `mqtime` have difficulties in parsing Chinese locations. `cntraveltime` solves this outstanding challenge via a feature that enables it to call route-planning services from the Baidu Maps Open Platform. The results of rigorous testing on the features of the command show that, relative to similar existing commands, `cntraveltime` has the highest capacity in terms of functionality and precision. This suggests that it can be regarded as a useful complement to other existing commands, especially when calculating travel distance and time for locations within China.

Keywords: dm0111, `cntraveltime`, Baidu Maps, travel time, travel distance, China geocoding

1 Introduction

Geographic distance is frequently used as an explanatory variable in economics research (O’Brien and Tan 2015). For example, distance significantly deters foreign investments in general (Daude and Fratzscher 2008; Javorcik and Wei 2009) and foreign equity investments in particular (Papaioannou 2009). According to Picard (2010), distance is based on the great circle formula called “the geodetic distance”. The greater the geodetic distance between two locations, the higher the cost of communication. However, geodetic distance fails to represent geographical distance accurately because the travel time between two locations is normally longer owing to landscapes such as mountains and rivers within the locations. Geographic conditions such as road conditions and climate may also make it hard for geodetic distance to efficiently measure the travel costs. Evidently, existing literature in this field has suggested further research on a more realistic distance measure.

Measuring the distance between two locations could be done better by calculating the travel time and distance between them. The extant literature has already introduced other commands such as `traveltime` (Ozimek and Miles 2011), `mqtime` (Voorheis 2015), `osrmtime` (Huber and Rust 2016), `georoute` (Weber and Péclat 2017; Weber, Péclat, and Warren 2022), and more community-contributed commands disclosed by the Statistical Software Components Archive (Anderson 2013; Ansari 2015; Heß 2015; Picard 2010; Zeigermann 2016). Some of these commands are still working, but others are already obsolete. Notably, the above commands do not perform efficiently in China; for example, `traveltime` does not work because it relies on Google Maps, which cannot be accessed in mainland China. Although other commands like `mqtime` and `georoute` are accessible in China, they cannot parse Chinese locations in Chinese or English languages.

China has a huge number of Stata users. Occasionally, they encounter challenges regarding Chinese addresses and the corresponding travel or distance between locations. Xue and Li (2020) describe how to transform an address into geographical coordinates defined by longitude and latitude with `cnocode` and the opposite with `cnaddress` in Stata. Based on this work, in this article we introduce a new command, `cntraveltime`. This command can calculate both the travel distance and the travel time between two locations defined by their geographical coordinates. The command submits a query to the Baidu Maps Open Platform Application Programming Interface (API), which enables it to solve the route-planning problems.¹ According to the results from a recent survey reported in the article titled “High-Resolution Map Solutions Market Share” by International Data Corporation, the Baidu Maps platform is the most used navigation service in China.² This shows that the calculated results are realistic and highly reliable.

To the best of our knowledge, `cntraveltime` could be the only command that can accurately calculate and analyze the travel distance and time with Chinese addresses so far. Furthermore, there are three features that distinguish our command from others. First, `cntraveltime` allows three different travel modes: driving, public transport, and cycling. In this regard, the command is more efficient than `georoute`, which provides only a driving mode. With driving mode, `cntraveltime` allows users to plan their route with specific options, including highway preference, highway avoidance, congested-section avoidance, toll-station avoidance, and so on. On public transport mode, `cntraveltime` can find a route with less transit, less walking distance, no subway, or even a combination of the above options. The command `osrmtime` also permits users to plan their routes with the above three transport modes, but it is quite complicated to use. Indeed, before running `osrmtime`, users must follow a series of prerequisites (Weber and Péclat 2017). All in all, `cntraveltime` offers a wide range of travel options

-
1. As an application provider, Baidu Maps Open Platform provides an API in the form of a web address to help developers obtain application services through network requests. The developers add necessary parameters to the request, submit it, and obtain a response.
 2. International Data Corporation is the global provider of market intelligence, advisory services, and events for the information technology, telecommunications, and consumer technology markets. For more details of this report (Doc # CHC47551021), visit the website: <https://www.idc.com/getdoc.jsp?containerId=CHC47551021&pageType=PRINTFRIENDLY>.

because it enables users to obtain both travel distances and travel times more accurately in specific locations and examine the differences among different travel modes.

The second feature of **cntraveltime** is its accuracy. To parse the addresses, users need to first convert the Chinese address into longitude and latitude using the command **cngcode** and then input them into the **cntraveltime** command for calculations. Although commands like **mqtime** and **georoute** can use the longitude and latitude of the location as the input option, for information security purposes, **cngcode** can output only the longitude and latitude of the location in the BD09II geodetic system, which is obtained by encryption transformation of the WGS84 geodetic system.³ If one uses **georoute** to process the latitude and longitude using **cngcode**, the results are inaccurate. On the contrary, **cntraveltime** is efficient enough to run this option because the geodetic system of this command is already BD09II by default.

The third feature is its large capacity. Commands like **georoute** and **mqtime** can retrieve travel distances between locations, but they have limitations on the number of requests for route calculations. Weber and Péclat (2017) and Weber, Péclat, and Warren (2022) state that in **georoute**, “we directly rely only on the HERE API”.⁴ The HERE API, however, provides developers with no more than 250,000 requests per month, including geocoding and routing.⁵ Voorheis (2015) reports that **mqtime** attempts to request the MapQuest OpenStreetMap API for each origin and destination pair, and as a backup it requests the HERE Maps API if the OpenStreetMap API fails to return a valid route. Again, MapQuest API allows only 15,000 access calls per month.⁶ On the other hand, **cntraveltime** has 30,000 daily route-planning quotas and 300,000 conversions between the coordinates and the real address per day, based on the verification of a registration key.⁷ Although some commands, such as **osrtime**, can use offline maps for unlimited calculations, they are not user friendly; they require users to perform many complex and time-consuming tasks, such as downloading huge map files.

With the above discussed features, we believe that **cntraveltime** should be regarded as a perfect complement to similar commands in terms of calculating locations within China because it has higher capacity, faster processing speed, and higher accuracy. We certainly believe that the command is the best option for users who need to calculate travel distance and time within China.

3. This fact comes from the relevant documents of Baidu Maps Open Platform; see <https://lbsyun.baidu.com/index.php?title=coordinate>.

4. The HERE API, like Baidu Maps Open Platform, is also an application service provider that provides route-planning services. For details, visit their website: <https://developer.here.com>.

5. These data come from <https://developer.here.com/change-plan>.

6. See <https://business.mapquest.com/pricing-plans>.

7. After July 4, 2022, when the developer completes the personal authentication on the Baidu Maps Open Platform, the developer has only 5,000 geocoding services and 5,000 routing services per day. But developers can continue to apply to increase the daily service quota to 300,000 geocoding services and 30,000 routing services per day, which is still free: <https://lbsyun.baidu.com/cashier/quota>.

2 The `cntraveltime` command and related tools

2.1 Preparation: Acquire a Baidu Maps API Key

Before using `cntraveltime`, users must get a Baidu Maps API Key, which is registered on the Baidu Maps Open Platform (<http://lbsyun.baidu.com>) just like `cngcode` and `cnaddress`. After registration, users are required to create a new application in the console, select the server for the application type, and enable the location retrieval and route-planning service. Users also must fill in the whitelist with 0.0.0.0/0 to ensure that calls from any IP address are allowed. An AK CODE is received because it is necessary for `cntraveltime`, `cngcode`, and `cnaddress`.⁸ The Baidu Maps Open Platform then allows users to get 300,000 geocoding services per day that can convert the address into the corresponding coordinate location and vice versa; users also get 30,000 route-planning services that calculate both travel time and travel distance between the two locations.

The command `cntraveltime`, however, accepts locations only with longitude and latitude as input information. So Chinese addresses must first be converted into latitude and longitude using the `cngcode` command. We also remind the reader that both `cngcode` and `cntraveltime` require the Baidu Maps Open Platform API to initiate the operation, and the same AK CODE that is generated after registration can be used for both commands. Therefore, it may not be necessary to apply for another AK CODE for `cngcode`.

2.2 The `cntraveltime` command

2.2.1 Syntax

The syntax of `cntraveltime` is as follows:

```
cntraveltime, baidukey(string) start_lat(varname) start_long(varname)
end_lat(varname) end_long(varname) [detail mode(string)
route_option(#) geodeticsystem("BD09"|"WGS84") intercitytype(#)
intercity_route_option(#)]
```

2.2.2 Options

`baidukey(string)` indicates the user's API key. A typical Baidu Maps API key is an alphanumeric string, and users must explicitly specify it. If the user already has a Baidu Maps API key, for example, CH8eak16UT1Eb10akeWYvofh, then the option must be specified as `baidukey(CH8eak16UT1Eb10akeWYvofh)`.⁹ `baidukey()` is required.

8. AK CODE acquisition requires a registered Baidu account, which is generally immediately available.

9. The API key used in this article is for illustration only and not available for use.

`start_lat(varname)` and `start_long(varname)` specify the latitude and longitude of the origin location. `start_lat()` and `start_long()` are required.

`end_lat(varname)` and `end_long(varname)` specify the latitude and longitude of the destination location. `end_lat()` and `end_long()` are required.

`detail` provides a text description about the route chosen by `cntraveltime`. By default, route descriptions are not saved.

`mode(string)` specifies the travel mode. The default is `mode("public")`. The available routing types are

- `public`
- `car`
- `bike`

Baidu Maps calculates both the travel distance and the travel time between two specified locations based on the selected transportation mode. There will be an error message if users specify a transit mode other than the above, for example, subway.

`route_option(#)` specifies the detailed preference when the given locations are in the same city. Note that, for different transport modes, the effective range and meaning of the `route_option()` are also different. This option accepts a number as the parameter, and table 1 shows that the preferences are represented by different numbers as follows:

Table 1. Valid parameters of `route_option()`

<code>mode()</code>	valid number in <code>route_option()</code>	meaning
<code>public</code>	0	default, recommendation
<code>public</code>	1	less transit
<code>public</code>	2	less walk
<code>public</code>	3	no subway
<code>public</code>	4	as quickly as possible
<code>public</code>	5	subway
<code>car</code>	0	default
<code>car</code>	3	avoid high speed
<code>car</code>	4	high speed priority
<code>car</code>	5	avoid congested sections
<code>car</code>	6	avoiding toll stations
<code>car</code>	7	both 4 and 5
<code>car</code>	8	both 3 and 4
<code>car</code>	9	both 4 and 6
<code>car</code>	10	both 6 and 8
<code>car</code>	11	both 3 and 6
<code>bike</code>	0	default, common
<code>bike</code>	1	electric bicycle

`geodeticsystem("BD09"|"WGS84")` specifies which geodetic system the latitude and longitude belong to. The default is `geodeticsystem("BD09")`. If the coordinate data input comes from the `cngcode` command, the default option will make the calculation result accurate because the coordinates produced by `cngcode` also belong to the BD09II geodetic system. On the other hand, if the coordinate data come from WGS84, the user should specify `geodeticsystem("WGS84")` for the sake of accuracy.

`intercitytype(#)` specifies the detailed preference when the given locations are in different cities. Note that this option is valid only when the `mode("public")` option is selected and the two locations are in the same city. This option accepts 0, 1, or 2 as input. These numbers correspond to the following preferences:

- 0: default; train
- 1: plane
- 2: bus

`intercity_route_option(#)` specifies the priority requirement for cross-city public transportation. This option accepts 0, 1, or 2 as input. These numbers correspond to the following preferences:

- 0: default; as quickly as possible
- 1: start as early as possible
- 2: as cheap as possible

3 Example

We start from a small dataset to illustrate how to use `cntraveltime`. This dataset has two variables: `startaddress` and `endaddress`. Both addresses are fully detailed. Thus, the province name, city name, district name, and address (on hand) are embedded in one string. For example, the first start address translated into English is as follows: “No. 85 Minglun Street, the University of Henan, Kaifeng, Henan Province”. We use the `cngcode` command to convert it into latitude and longitude as the first step. Before proceeding, we remind the reader that the geodetic systems of all latitudes and longitudes in this article are BD09II. Because both `cngcode` and `cntraveltime` use this geodetic system by default, deviations in the calculation of results are not expected.

```
. * Set baidukey()
. global BaiduKey CH8eakl6UTlEb10akeWYvofh
. * Input address
. use "https://cntraveltime.oss-cn-hangzhou.aliyuncs.com/data1.dta"
. * Obtain coordinates using cngcode
. cngcode, baidukey($BaiduKey) fulladdress(startaddress) lat(start_lat)
> long(start_lon)
. cngcode, baidukey($BaiduKey) fulladdress(endaddress) lat(end_lat)
> long(end_lon)

. * Display
. list start_lon - end_lat
```

	start_lon	start_lat	end_lon	end_lat
1.	114.37568	34.817677	114.41983	30.518659
2.	114.37102	30.544857	114.30759	30.588503
3.	114.37102	30.544857	114.41983	30.518659
4.	114.37102	30.544857	114.36371	30.48178
5.	116.62193	40.058743	116.40396	39.915119

Now we can compute distances and durations using `cntraveltime`. So we also use `geodist` to get the geodetic distance for comparison.

```
. * Compute Geodetic distance using geodist
. geodist start_lat start_lon end_lat end_lon, gen(geo_dist)

. * Compute distances and duration using cntraveltime
. cntraveltime, baidukey($BaiduKey) start_lat(start_lat) start_long(start_lon)
> end_lat(end_lat) end_long(end_lon) mode(car) detail
Please note: The longitude and latitude used in this calculation belong to
> BD09II

. * Display
. list geo_dist distance duration
```

	geo_dist	distance	duration
1.	476.77381	537.245	374.5667
2.	7.7747829	10.609	19.95
3.	5.5109943	9.739	22.41667
4.	7.0279759	11.651	23.36667
5.	24.513308	29.8	36.81667

The start address for our first four observations is the working place of the authors, and the end addresses are some universities and the Wuhan Yangtze River Bridge, which is a famous place located near the authors' workplace. Because `cntraveltime` accepts only latitude and longitude coordinates as the input of two locations, we need to convert the addresses into corresponding coordinates via `cngcode`. At the same time, we also need to calculate the geodetic distances for given data by `geodist`, which is labeled as the `geo_dist` variable. We have purposely selected these addresses because the geodetic distance between the two locations generates different results relative to the traveling distance. As we can see from the observations' results above, the actual trips between Wuhan University and the Wuhan Yangtze River Bridge, obtained through the `cntraveltime` command, were 10.61 km based on the second observation; and the trips between Wuhan University and Huazhong University of Science and Technology were 9.74 km based on the third observation; the difference is not much. On the other hand, we found that the geodetic distance of the second observation is 7.77 km and the geodetic distance of the third observation is 5.51 km, which is only 70% of the former. This could be because the road between the two universities is in the bustling city block, and so the driving route is more tortuous.

To illustrate the difference between `geo_dist` and travel distance, we pick the fourth observation and illustrate it graphically for clear understanding. As seen in figure 1, the starting point in the map below is the Beijing Capital International Airport, one of the largest international airports in China. The endpoint is Tiananmen Square, which is recognized as the geographical center of Beijing. The distance represented by the straight line is `geo_dist`, and the distance represented by the multisegment polyline is the travel distance. From the map, it is clear that the route that people travel from the starting point to the destination is tortuous and more realistic. This could be due to urban planning.

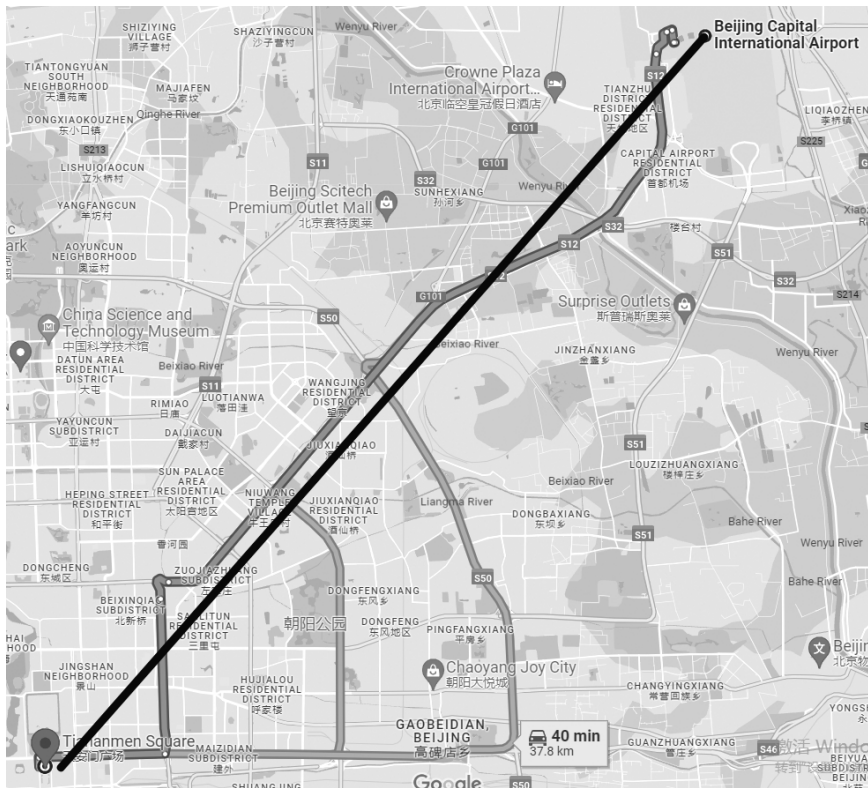


Figure 1. Source: Google Maps

Based on the above findings, we are confidently concluding that in real life, especially in big cities, the spherical distance does not efficiently measure the true distance between the two locations. Nevertheless, the travel distance and time consider the local traffic environment, making the results more meaningful.

The command `cntraveltime` not only provides a more precise travel distance than geodetic distance but also offers a wide range of travel options such as driving, public transport, and cycling. Below, we show a new dataset. We use `cngcode` to obtain coordinates of the start and end addresses.

```

. * Input address
. use "https://cntraveltime.oss-cn-hangzhou.aliyuncs.com/data2.dta", clear
. * Obtain coordinates of start address and end address using cngcode
. cngcode, baidukey($BaiduKey) fulladdress(startaddress) lat(start_lat)
> long(start_lon)
. cngcode, baidukey($BaiduKey) fulladdress(endaddress) lat(end_lat)
> long(end_lon)
. * Display
. list start_lon - end_lat

```

	start_lon	start_lat	end_lon	end_lat
1.	114.35046	30.506334	114.36371	30.48178
2.	116.62193	40.058743	116.40396	39.915119
3.	101.78445	36.623385	100.50297	36.584302
4.	120.31309	31.598118	121.35526	31.196776

We can get three different travel options to observe the results:

```

. * Compute distances and duration using cntraveltime by car
. cntraveltime, baidukey($BaiduKey) start_lat(start_lat) start_long(start_lon)
> end_lat(end_lat) end_long(end_lon) mode(car)
Please note: The longitude and latitude used in this calculation belong to
> BD09II
. rename (distance duration) (dist_car dura_car)
. * Compute distances and duration using cntraveltime by public default
. cntraveltime, baidukey($BaiduKey) start_lat(start_lat) start_long(start_lon)
> end_lat(end_lat) end_long(end_lon) detail
Please note: The longitude and latitude used in this calculation belong to
> BD09II
. rename (distance duration detail) (dist_pub dura_pub detail_pub)
. * Compute distances and duration using cntraveltime by bike
. cntraveltime, baidukey($BaiduKey) start_lat(start_lat) start_long(start_lon)
> end_lat(end_lat) end_long(end_lon) mode(bike)
Please note: The longitude and latitude used in this calculation belong to
> BD09II
. rename (distance duration) (dist_bike dura_bike)
. * Display
. list dist* dura*

```

	dist_car	dist_pub	dist_b_e	dura_car	dura_pub	dura_b_e
1.	5.798	3.137	4.654	11.68333	18.1	26.35
2.	29.8	37.319	29.338	36.78333	74.21667	166.4833
3.	160.112	.	150.136	187.2167	.	836.5833
4.	128.589	7.607	126.839	109.3333	123.0167	711.65

From the results table, we see the travel distances for all three modes of transportation are not widely dispersed. However, the time spent by driving is much less than the other two modes. These findings suggest that travel distance and time should be analyzed separately. The third observation has no data on public transport; perhaps there is no feasible route for public transport between two locations. Note that the fourth

observation, that is, the travel distance in public transport mode, is much smaller than the driving distance, but the corresponding travel time is almost the same. This is because Baidu Maps does not calculate the distance traveled by using trains and buses, or public transport mode, when calculating the travel distance. Thus, the distance in public transport mode is inaccurate.¹⁰

When calculating the travel distance in the public mode, the command also uses the `detail` option, which displays the detailed traveling plan between two locations. For the public mode, it displays the name of the platform and the distance of each trip. For example, the first observation of the dataset is translated into English as follows: “walk 1.0 km, then take 567 Road (South Lake Avenue Huazhong Nongda East), 570 Road or 590 Road, after 3 stops to Nanhu Avenue Lanhua Yuan Station, then walk 891 meters.” The display content is very detailed, so users can accurately find their specific route on the map.

Once the traveling mode is chosen, `cntraveltime` could help in setting traveling preferences that simulate a more specific environment. For example, when public transport mode is chosen, some people can spend less time on routes, while others can walk less on the trip. In any case, our outstanding problem is that we are not aware whether the travel distance and the times of different traveling preferences are significantly different. To investigate these situations, we construct the following dataset:

```
. * Input address
. use "https://cntraveltime.oss-cn-hangzhou.aliyuncs.com/data3.dta", clear
. * Obtain coordinates of start address and end address using cngcode
. cngcode, baidukey($BaiduKey) fulladdress(startaddress) lat(start_lat)
> long(start_lon)
. cngcode, baidukey($BaiduKey) fulladdress(endaddress) lat(end_lat)
> long(end_lon)
. * Display
. list start_lon - end_lat
```

	start_lon	start_lat	end_lon	end_lat
1.	114.26934	30.62357	114.31158	30.598467
2.	114.27301	30.571067	114.3246	30.542747
3.	114.30774	30.544218	114.29314	30.534964
4.	113.63142	34.753439	116.41338	39.910925
5.	114.31158	30.598467	113.96346	30.924526
6.	121.48054	31.235929	108.94647	34.347269

We use `route_option()` to specify different preferences. Here we specify 1, 2, and 4, which respectively represent less transit, less walking, and arriving as quickly as possible.

10. The travel distance calculations will be inaccurate only when the two locations are in different cities and the train and bus modes are selected in the `intercitytype()` option. If one chooses to travel by plane, the travel distances' and times' calculations are correct.

```

. * Compute distances and duration using cntraveltime
. cntraveltime, baidukey($BaiduKey) start_lat(start_lat) start_long(start_lon)
> end_lat(end_lat) end_long(end_lon) mode(public) route_option(1)
Please note: The longitude and latitude used in this calculation belong to
> BD09II
. rename (distance duration) (dist_b_1 dura_b_1)
. cntraveltime, baidukey($BaiduKey) start_lat(start_lat) start_long(start_lon)
> end_lat(end_lat) end_long(end_lon) mode(public) route_option(2)
Please note: The longitude and latitude used in this calculation belong to
> BD09II
. rename (distance duration) (dist_b_2 dura_b_2)
. cntraveltime, baidukey($BaiduKey) start_lat(start_lat) start_long(start_lon)
> end_lat(end_lat) end_long(end_lon) mode(public) route_option(4)
Please note: The longitude and latitude used in this calculation belong to
> BD09II
. rename (distance duration) (dist_b_4 dura_b_4)

```

At the same time, we can observe the situation of *car*. The preferences of avoiding high speed and prioritizing high speed are shown below.

```

. cntraveltime, baidukey($BaiduKey) start_lat(start_lat) start_long(start_lon)
> end_lat(end_lat) end_long(end_lon) mode(car) route_option(3)
Please note: The longitude and latitude used in this calculation belong to
> BD09II
. rename (distance duration) (dist_c_3 dura_c_3)
. cntraveltime, baidukey($BaiduKey) start_lat(start_lat) start_long(start_lon)
> end_lat(end_lat) end_long(end_lon) mode(car) route_option(4)
Please note: The longitude and latitude used in this calculation belong to
> BD09II
. rename (distance duration) (dist_c_4 dura_c_4)

```

Here is the result.

```

. * Display
. list dist_b_1 - dura_b_4 in 1/3

```

	dist_b_1	dura_b_1	dist_b_2	dura_b_2	dist_b_4	dura_b_4
1.	6.64	51.85	7.096	55.6	6.448	50.08333
2.	8.327	31.81667	10.776	54.9	8.327	31.81667
3.	6.668	55.56667	10.023	79.33334	6.668	55.56667

```

. list dist_c_3 - dura_c_4 in 4/6

```

	dist_c_3	dura_c_3	dist_c_4	dura_c_4
4.	750.445	1063.517	692.866	494.9167
5.	63.36	113.7167	68.933	70.33334
6.	1487.078	2009.733	1384.907	906.6667

The first three observations from the above dataset used well-known and prosperous locations in Wuhan, such as the South China Seafood Market, Hanzheng Street, and the Yellow Crane Tower. So when people travel using public transport mode, different

traveling preferences may produce large deviations in the results. From the calculations, we find that when the traveling preference is for less walking, the corresponding travel distance, `dist_b_2`, is longer than in other cases. On the other hand, when the preference is to arrive as quickly as possible, the corresponding travel time, `dura_b_4`, is no higher than in other cases. The findings entail that when the user prefers less travel time, the travel distance is not literally less, and vice versa. Of course, at a large scale, travel distance and time can be considered as effective measures of commuting costs between the two locations.

The last three observations from the above dataset used other cities as locations to capture the differences in traveling, down to the different preferences of car driving in China. Regardless of the preferred option, the results show that their travel distances are basically the same. When the user prefers to prioritize high speed, the travel time, `dura_c_4`, is much less than `dura_c_3`, which represents avoiding high speed. Evidently, based on the calculations above, in the current dataset, the former is 51% of the latter. These results show that there is a possibility that a highway could be reducing the travel time between the two locations significantly.

The above discussed cases are all based on the route within one city. If two locations are placed in two different cities, say, thousands of miles apart, the public transport mode option at a city level will not work. The command `cntraveltime`, on the other hand, provides many options for cross-city public transportation. So it allows users to accurately calculate travel distance and time across cities. Below, we illustrate the process. We are still using `data3.dta`.

```
. * Input address
. use "https://cntraveltime.oss-cn-hangzhou.aliyuncs.com/data3.dta", clear
. keep in 4/6
(3 observations deleted)

. * Obtain coordinates of start address and end address using cngcode
. cngcode, baidukey($BaiduKey) fulladdress(startaddress) lat(start_lat)
> long(start_lon)
. cngcode, baidukey($BaiduKey) fulladdress(endaddress) lat(end_lat)
> long(end_lon)

. * Display
. list start_lon - end_lat
```

	start_lon	start_lat	end_lon	end_lat
1.	113.63142	34.753439	116.41338	39.910925
2.	114.31158	30.598467	113.96346	30.924526
3.	121.48054	31.235929	108.94647	34.347269

Then we use `intercity_route_option()` to calculate three situations:

```
. * 1. As quickly as possible
. cntraveltime, baidukey($BaiduKey) start_lat(start_lat) start_long(start_lon)
> end_lat(end_lat) end_long(end_lon) mode(public) route_option(1)
> intercitytype(0) intercity_route_option(0)
Please note: The longitude and latitude used in this calculation belong to
> BD09II

. rename (distance duration) (dist_1 dura_1)

. * 2. Start as early as possible
. cntraveltime, baidukey($BaiduKey) start_lat(start_lat) start_long(start_lon)
> end_lat(end_lat) end_long(end_lon) mode(public) route_option(1)
> intercitytype(0) intercity_route_option(2)
Please note: The longitude and latitude used in this calculation belong to
> BD09II

. rename (distance duration) (dist_2 dura_2)

. * 3. As cheap as possible
. cntraveltime, baidukey($BaiduKey) start_lat(start_lat) start_long(start_lon)
> end_lat(end_lat) end_long(end_lon) mode(public) route_option(1)
> intercitytype(1) intercity_route_option(2)
Please note: The longitude and latitude used in this calculation belong to
> BD09II

. rename (distance duration) (dist_3 dura_3)

. * Display
. list dist_1 - dura_3
```

	dist_1	dura_1	dist_2	dura_2	dist_3	dura_3
1.	14.565	247.6333	14.565	458.6333	.	.
2.	67.185	206.5	67.185	206.5	67.185	206.5167
3.	24.024	538.3834	24.024	538.3834	1346.494	389.0167

We have used the same last three observations of `data3.dta` as the detection data because it is easy for us to observe the impacts of different public transport options (`intercitytype()` and `intercity_route_option()`) on the final results because these locations are across different cities in China. The result of the first calculation (`dist_1` and `dura_1`) represents the route option whereby people take the train and arrive as soon as possible, so the value of `dura_1` for all observations is not higher than the value of `dura_2`. Notably, because of the calculation characteristics of the Baidu Maps Open Platform, calculating traveling distance by train (`dist_1` and `dist_2`) produces incorrect results, while the calculation for travel distance by plane (`dist_3`) is correct.

The results of the third calculation (`dist_3` and `dura_3`) represent the option whereby people take a plane and spend less money on traveling. Notably, `dura_1` represents the shortest time when traveling by train. However, the results for its third observation indicate that more time could be spent than on `dura_3`, suggesting that the fastest train is not as fast as the plane. The first observation of the route plane option is blank because there is no flight between Beijing and Zhengzhou on that day.

Although `cntraveltime` is calculated requesting the Baidu Maps Open Platform first, the result from the command calculation is based on real-time information because

it factors in the current road conditions too. The command `cntraveltime` remains reliable in the calculations of both travel distance and time. This is because Chinese infrastructure is carefully planned and makes the navigation exercise easy and reliable. Thus, calculation of the best route and travel time from one location to another is sustainably reliable. This is an essential requirement for program reproduction.

4 Conclusion

In this article, we introduced the command `cntraveltime`. The command can calculate the travel distance and time between two locations within China. Its capacity is large because it can accommodate many features or options. Its processing speed is fast, and it has a high accuracy level. It could be useful to users who need to calculate travel distances and times within China. Therefore, it can be regarded as the best complement to some commands that calculate locations within China. As an API-based command, `cntraveltime` does not take much disk space on a computer and has various extended functions. In addition, we appreciate the service provider's own stability, so the command is not expected to be changed in the foreseeable future. In the next stage, we plan to optimize the program further to improve the execution speed of the command and identify the reasons for the failure of the calculation. We hope our ongoing work continues to benefit the Stata user community in the future.

5 Programs and supplemental materials

To install a snapshot of the corresponding software files as they existed at the time of publication of this article, type

```
. net sj 23-3
. net install dm0111      (to install program files, if available)
. net get dm0111         (to install ancillary files, if available)
```

6 References

- Anderson, M. L. 2013. `geocodeopen`: Stata module to geocode addresses using MapQuest Open Geocoding Services and Open Street Maps. Statistical Software Components S457733, Department of Economics, Boston College. <https://ideas.repec.org/c/boc/bocode/s457733.html>.
- Ansari, M. R. 2015. `gcode`: Stata module to download Google geocode data. Statistical Software Components S457969, Department of Economics, Boston College. <https://ideas.repec.org/c/boc/bocode/s457969.html>.
- Daude, C., and M. Fratzscher. 2008. The pecking order of cross-border investment. *Journal of International Economics* 74: 94–119. <https://doi.org/10.1016/j.jinteco.2007.05.010>.

- Heß, S. 2015. geocodehere: Stata module to provide geocoding relying on Nokia's Here Maps API. Statistical Software Components S458048, Department of Economics, Boston College. <https://ideas.repec.org/c/boc/bocode/s458048.html>.
- Huber, S., and C. Rust. 2016. Calculate travel time and distance with OpenStreetMap data using the Open Source Routing Machine (OSRM). *Stata Journal* 16: 416–423. <https://doi.org/10.1177/1536867X1601600209>.
- Javorcik, B. S., and S.-J. Wei. 2009. Corruption and cross-border investment in emerging markets: Firm-level evidence. *Journal of International Money and Finance* 28: 605–624. <https://doi.org/10.1016/j.jimonfin.2009.01.003>.
- O'Brien, P. C., and H. Tan. 2015. Geographic proximity and analyst coverage decisions: Evidence from IPOs. *Journal of Accounting and Economics* 59: 41–59. <https://doi.org/10.1016/j.jacceco.2014.11.002>.
- Ozimek, A., and D. Miles. 2011. Stata utilities for geocoding and generating travel time and travel distance information. *Stata Journal* 11: 106–119. <https://doi.org/10.1177/1536867X1101100107>.
- Papaioannou, E. 2009. What drives international financial flows? Politics, institutions and other determinants. *Journal of Development Economics* 9: 43–64. <https://doi.org/10.1016/j.jdeveco.2008.04.001>.
- Picard, R. 2010. geodist: Stata module to compute geodetic distances. Statistical Software Components S457147, Department of Economics, Boston College. <https://ideas.repec.org/c/boc/bocode/s457147.html>.
- Voorheis, J. 2015. mqttime: A Stata tool for calculating travel time and distance using MapQuest web services. *Stata Journal* 15: 845–853. <https://doi.org/10.1177/1536867X1501500316>.
- Weber, S., and M. Péclat. 2017. A simple command to calculate travel distance and travel time. *Stata Journal* 17: 962–971. <https://doi.org/10.1177/1536867X1801700411>.
- Weber, S., M. Péclat, and A. Warren. 2022. Travel distance and travel time using Stata: New features and major improvements in georoute. *Stata Journal* 22: 89–102. <https://doi.org/10.1177/1536867X221083857>.
- Xue, Y., and C. Li. 2020. Extracting Chinese geographic data from Baidu Map API. *Stata Journal* 20: 805–811. <https://doi.org/10.1177/1536867X20976313>.
- Zeigermann, L. 2016. opencagegeo: Stata module for forward and reverse geocoding using the OpenCage Geocoder API. Statistical Software Components S458155, Department of Economics, Boston College. <https://ideas.repec.org/c/boc/bocode/s458155.html>.

About the authors

Xueren Zhang is a PhD student in finance at the Wuhan University in Wuhan, China.

Yuan Xue is a PhD student in accounting at the Huazhong University of Science and Technology in Wuhan, China.

Chuntao Li is a professor of finance at the Henan University in Kaifeng and at the Zhongnan University of Economics and Law in Wuhan, China.

A Appendix: The process of applying for a Baidu Maps API Key

This appendix aims at providing detailed guidance on how to apply for the Baidu Maps API Key, which is required to run *cntraveltime*.

A.1 Sign up for and log into a Baidu account

To begin, users must sign up for a Baidu account. This is a prerequisite made by Baidu for all users to use Baidu Maps Open Platform resources. Below are further steps:

Users need to visit Baidu's main page (<https://www.baidu.com>), click on the [Login] (登录) button in the upper-right corner, and fill in the account name and password in the provided blank fields (see figure 2).



Figure 2

If users do not have a Baidu account, they can click on the [Register Now] (立即注册) button in the lower-right corner of the pop-up window to enter the registration page for a Baidu account (see figure 3).



The image shows a Baidu login and registration interface. At the top, there are two tabs: '帐号登录' (Account Login) and '短信登录' (SMS Login). The '帐号登录' tab is selected and underlined. To the right of the tabs is a close button 'X'. Below the tabs are two input fields: the first is labeled '手机号/用户名/邮箱' (Mobile Number/Username/Email) and the second is labeled '密码' (Password). To the right of the password field is a link '忘记密码?' (Forgot Password?). Below the input fields is a large dark button labeled '登录' (Login). Below the login button is a link '阅读并接受 百度用户协议 和 隐私政策' (Read and accept Baidu User Agreement and Privacy Policy). At the bottom right, there is a button labeled '立即注册' (Register Now).

Figure 3

On the registration page for a Baidu account, users need to follow the instructions on the page and input the requested information: username (用户名), mobile phone number (手机号), password (密码), and mobile phone verification code (验证码) (see figure 4).

The image shows the Baidu registration page with the title "欢迎注册" (Welcome to Register) and a link "已有帐号? 登录" (Already have an account? Log in). The registration form includes the following fields and buttons:

- 用户名 (Username):** A text input field with the placeholder "请设置用户名" (Please set a username).
- 手机号 (Mobile Number):** A text input field with the placeholder "可用于登录和找回密码" (Can be used for login and password recovery).
- 密码 (Password):** A text input field with the placeholder "请设置登录密码" (Please set a login password) and a password strength indicator icon.
- 验证码 (Verification Code):** A text input field with the placeholder "请输入验证码" (Please enter verification code) and a button labeled "获取验证码" (Get verification code).

Below the form is a large "注册" (Register) button. At the bottom, there is a checkbox for "阅读并接受《百度用户协议》、《儿童个人信息保护声明》及《百度隐私权保护声明》" (Read and accept the Baidu User Agreement, Children's Personal Information Protection Statement, and Baidu Privacy Policy).

Figure 4

After the registration is completed successfully, users can visit the official website (<https://lbsyun.baidu.com/index.php>). The login information is automatically displayed in the upper-right corner, indicating that the Baidu account has been successfully registered and is ready for any preliminary work (see figure 5).



Figure 5

A.2 Developer certification

To get the API quota, users must obtain a developer certificate in addition to a Baidu account. As figure 5 shows above, users can acquire the certification by first clicking on [Console] (控制台) in the upper-right corner to enter the function list of the platform. Then, as shown in figure 6 below, click on [Personal Center] (个人中心) - [Developer Certification] (开发者认证) - [Individual Developer] (个人开发者). If users own enterprises in China, they can click on [Enterprise Developer] (企业开发者) instead of [Individual Developer] to get more API quotas.



Figure 6

If users register as individual developers, relevant information such as name (姓名), mobile phone number (手机), personal email address (邮箱), ID number (身份证号), industry type (行业类型), work address (工作地址), and reason (使用场景) for use must be submitted (see figure 7). After this process is completed, certification results are generated immediately.

1.个人信息填写

2.人脸识别认证

3.认证结果

* 姓名:

张学人

修改姓名

证件类型:

身份证

* 身份证号:

请输入身份证号

* 行业类别:

请选择行业类别

* 工作地址:

省

地级市

市、县级市

* 账号用途:

☐ 个人使用

☐ 公司/单位项目开发

* 使用场景:

请详细描述: 您的产品名称、业务场景, 使用开放平台哪些接口服务, 填写内容必须在(50)字以上。

0 / 1000

Figure 7

A.3 Applying for a Baidu Maps API key

After the developer's certification process is completed, a new Baidu Maps API key for `cntraveltime` use can be requested by clicking on [Application Management] (应用管理) - [My Application] (我的应用) - [Create Application] (创建应用) on the screen (see figure 8).



Figure 8

The application form for an API key should be carefully filled in by making sure [Server] (服务端) on [Application Type] (应用类型) is selected and the computer IP address in the IP whitelist (IP 白名单) is filled in. This is to ensure that only the applicant can use this key. However, if one prefers that computers with any IP address be enabled to use this key after authorization, then 0.0.0.0/0 can be completed in the whitelist. The page should look like the one in figure 9. Once the application for the key is completed, the new API key is shown on the corresponding page of the console, and the key content can be copied to any place where it is needed to be used.



Figure 9

When the developer certificate is obtained, the maximum limit of the number of daily calls of various applications can be checked on the [Quota Management] (额度管理) - [My Quota] (我的额度) part of the console. After July 4, 2022, users have only the 5,000 geocoding services and 5,000 routing services per day as seen in figure 10, but users can click on the icon in figure 10 and fill in the reason to apply to increase the daily quota to 300,000 geocoding services and 30,000 routing services per day, which is still free.



Figure 10