BIOROUTE C++ documentation

Introduction

This documentation explains the usage of bioroute v1.0, a implementation of Metropolis-Hastings path sampling (MHPS) algorithm for route choice modeling using GPS data.

Features:

1. MHPS algorithm.
2. Random walk algorithm.
3. Deals with probabilistic map matching results.
4. Generation of biogeme inputs (model file and data file).

Installation

Usage

The basic usage is following:

```
bioroute -c COMMAND -f PARAM_FILE -r -n NBR_OBSERVATIONS -s
```

`-c COMMAND`	the function to be executed
`-f PARAM_FILE`	is the path to the parameter file which is explained in Section Parameters.
`-r`	specified if the observations are real, not synthetic.
`-s`	specified if the network is synthetic, not real.
`-n NBR_OBSERVATIONS`	number of observations.

COMMAND

- *Sample*: sampling alternatives for route choice observations.
- *SampleWithOd*: sampling alternatives with the given OD.
- *SampleEqualProbability*: sampling alternatives with equal probability
- *MHEnumeratePaths*: enumerate paths
- *WriteBiogeme*: write biogeme model and data files
- *SimulateObservation*: simulate route choice observations with specified route choice model.
- *ExportNetwork*:
- *TestNetwork*:
- *Verify*: verify sampling results.
- *SimulateObservationsError*: simulation errors in observations.
- *WriteNetworkToDB*: write network to postgresql database.
- *AnalyzeChoiceSet*:
- *WriteChoiceSetSHP*:
- *KML2SHP*:

- *OBS2SHP*:
- *AnalyzeOBS*:

Sampling alternatives for route choice observations

The most important feature of *bioroute* is to sample alternatives for GPS map matching results generated by *pmm*

```
bioroute -c Sample -f PARAM_FILE  -r
```

where *-r* indicates that the map matching results are from real data, not synthetic route choice results.

Sampling alternatives with the given OD

Sampling alternatives with equal probability

Simulate route choice observations

Parameters

```
<module name="pathSampling">
<param name="pathSampleAlgorithm" value="MH" type="string"/>
<!--
        Sampling algorithm:
                - MH: Metropolis-Hastings path sampling.
                - RW: random walk.
-->

<!-- total number of samples to be generated-->
        <param name="SAMPLE_COUNT" value="100" type="int"/>

        <!-- For biogeme data, different number of alternatives can be specified -->
        <!-- It is used to test the effect of different number of alterantives -->
        <!-- Syntax is `NUMBER,`, for example, 30,50,100, -->
<param name="choiceSetInBiogemeData" value="30,50,100," type="string"/>

<!-- The behavior of sampling alternative or not if the result already exists-->
<!-- 1  sample again and overwrite the sample file-->
<!-- 0 does not sample in order to save time -->
<param name="overwriteSampleFile" value="1" type="int"/>
</module>

<module name="RandomWalk">
        <!-- Parameters for random walk algorithm. -->
        <param name="kumaA" value="30.0" type="float"/>
        <param name="kumaB" value="1.0" type="float"/>
</module>


<module name="MHPathSampling">

<!-- Folder for experiment and its subfolders. -->
<!--    experimentDirectory: the parent folder. -->
<!--    observationDirectory: the subfolder that contains the route choice observations (mapmatching results) -->
<!--    choiceSetFolder: the subfolder that the sampling results writes to -->
<param name="experimentDirectory" value="EXPERIMENT_HOME" type="string"/>
<param name="observationDirectory" value="path_observations/" type="string"/>
<param name="choiceSetFolder" value="1.026_1.0/" type="string"/>

        <!-- warm up iteration and sampling interval -->
<param name="WARMUP_ITERATIONS" value="500000" type="int"/>
        <param name="SAMPLEINTERVAL_ELEMENT" value="100000" type="int"/>

        <!-- zeta -->
<param name="mh_link_scale"  value="1.026" type="float"/>
```

```
<!-- coefficients for length, speed bump and path size -->
<param name="mh_length_coef"  value="-0.01" type="float" />
<param name="mh_sb_coef"  value="0.0" type="float"/>
<param name="mh_ps_coef"  value="0.0" type="float"/>

<!-- omega 2 parameter -->
<param name="mh_obs_scale"  value="1.0" type="float"/>


<!-- Export reduced network. 1 export, 0 no.-->
<!-- Reduced network is in MHPS paper.-->
<!-- Cut off probability for reduced network. TODO check the function.-->
<!--
```

```
                double expansion = 1.0 - log(m_cutOffProbability) / (linkCostSP);
                if (patNBParameters::the()->mh_link_scale_relative == 1) {
                        expansion = 1.0
                                        - log(m_cutOffProbability)
                                                        * (patNBParameters::the()->mh_link_scale - 1.0)
                                                        / (log(2));
                }
-->
<param name="exportReducedNetwork" value="1" type="int"/>
<param name="CUTOFFPROBABILITY_ELEMENT" value="0.00000001" type="float"/>

<!-- Export compressed network network.
                Compressed network merges links which do not have intersections in between-->
<param name="compressNetwork"  value="1" type="int"/>

        <!-- Origin and destination node id in OSM network. -->
        <param name="OriginId" value="296253984" type="int"/>
        <param name="DestinationId" value="973818127" type="int"/>




        <!-- random seed for sampling -->
        <param name="RANDOMSEED_ELEMENT" value="4213332" type="int"/>
        <!-- print message every X iterations -->
        <param name="MSGINTERVAL_ELEMENT" value="10000" type="int"/>



        <param name="RELATIVECOSTSCALE_ELEMENT" value="1.0200" type="float"/>

        <!--- link cost scale for router -->
        <!--- Don't chage it! -->
<param name="router_cost_link_scale"  value="-1.0" type="float"/>
<param name="mh_function_type"  value="2" type="int"/>
<param name="mh_trigger_path"  value="" type="string" />


<!-- Do not change it -->
<!-- patLinkAndPathCost -->
<!-- MHPathGenerator -->
<!-- MHSamplingWeightFunction -->
<param name="mh_link_scale_relative"  value="1" type="int"/>



</module>


<module name="SimulateObservation">

        <param name="samplingWithObs" value="1" type="int"/><!-- used in synwritebiogeme.cc -->
<param name="ObsErrorDistanceScale" value="0.1" type="float"/>
<param name="ObsErrorDistance" value="200.0" type="float"/>
<param name="nbrOfSimulatedErrorPaths"  value="5" type="int"/>
<param name="errorInSimulatedObservations"  value="0.2" type="float"/>
</module>


<module name="UselessAfterMultiModal">
        <param name="TOTALITERATIONS_ELEMENT" value="100000" type="int"/>
        <param name="writeBiogemeModelFile" value="1" type="int"/>
        <param name="syntheticObservations" value="1" type="int"/>

        <param name="pathSamplingAlgo" value="1" type="int"/>
                <param name="maxNumberOfGeneratedPaths" value="200" type="int"/>
        <param name="maxTrialsForRandomWalk" value="50" type="int"/>
</module>
```

A use case

This case study implements the real data experiment described in Jingmin Chen's thesis. The map matching results of 19 trips are given. Decompress the content to a folder, and we note the path to this directory as **EXPERIMENT_HOME** The configuration file *config_gps_route_choice.xml* is in the param folder.

```
<param name="experimentDirectory" value="EXPERIMENT_HOME/" type=string"/>
```

Note: always put a / after a directory!

Input

The unimodal map matching is applied to generate path observations from GPS data. The gps data is in *gps* folder. * Map matching results with *kml* format are in *path_observations* folder.

```
<param name="observationDirectory" value="path_observations/"string"/>
```

We would like to use MH path sampling to generate biogeme inputs. Each set of sampled alternatives is composed of 100 draws. In order to test the effect of the sampling size, different sets of alterantives are produced: 30, 50, 100. There will be three inputs generated. One contains 30 draws in the sampled alterantives; one contains 50, and one contains 100.

```
<param name="SAMPLE_COUNT" value="100" type="int"/>
<param name="choiceSetInBiogemeData" value="30,50,100," type="string"/>
```

Note `,` after *100*. It is a required syntax.

- Set the *burn-in* and *sampling interval* for MH path sampling

```
<param name="WARMUP_ITERATIONS" value="500000" type="int"/>
<param name="SAMPLEINTERVAL_ELEMENT" value="100000" type="int"/>
```

- Set $\zeta$and ω_2 parameters for the sampling

```
<param name="mh_link_scale"  value="1.026" type="float"/>
<param name="mh_obs_scale"  value="1.0" type="float"/>
```

- Set the sampling results folder. The choice set for each observation, along with the biogeme dat file and mod file will be produced in this folder.

```
<param name="choiceSetFolder" value="1.026_1.0/" type="string"/>
```

- Set the network file

```
<param name="OsmNetworkFileName" value="EXEPRIMENT_HOME/lausanne.osm" type="string"/>
```

Run MH sampling

```
EXPERIMENT_HOME$ /data/jingmin/bin/bioroute -c Sample -f params/config_gps_route_choice.xml  -r
```

Output

- For each candidate path, a set of alterantives are generated. For an observation with file name *3 -6623*, there are 7 candidate paths, noted from *3 -6623_1* to *3 -6623_7*. For path 1 *3 -6623_1_sample.kmlreduced.kml.kml* and *3 -6623_1_sample.kmlreduced.kml.shp* are the reduced network (described in MH paper) in *kml* and *shp* format. *3 -6623_1_sample.kml* is the sampling result.
- Three biogeme data files are produced, with 30, 50, 100 draws respectively. The files are *observations_30.dat*, *observations_50.dat* and *observations_100.dat*