GPS2space: An Open-source Python Library for Spatial Measure Extraction from GPS Data

Supplementary Materials

Supplemental material 1. GPS2space source code and documentation
Source code: https://github.com/shuai-zhou/gps2space

Documentation: https://gps2space.readthedocs.io/en/latest/

Supplemental material 2. Python code for iterating over multiple activity space features

```
# ----------------------------------------------------------
# Import libraries
# ----------------------------------------------------------
import pandas as pd
import geopandas as gpd
from gps2space import geodf
from gps2space import space
from gps2space import dist

print('Pandas version is:', pd.__version__)
print('Geopandas version is:', gpd.__version__)

# ----------------------------------------------------------
# Calculate buffer-based activity space
# ----------------------------------------------------------
df_twinX = pd.read_csv('./data/TwinX.csv')
gdf_twinX = geodf.df_to_gdf(df_twinX, x='longitude', y='latitude')
gdf_twinX['uid'] = gdf_twinX['user_id'].astype(str) + '_' + gdf_twinX['week'].astype(str)

buff_twinX = space.buffer_space(gdf_twinX, dist=1000, dissolve='uid', proj=2163)

# ----------------------------------------------------------
# Iterate over multiple activity space features
# ----------------------------------------------------------
shared_space_list = []

for idx, row in buff_twinX.iterrows():
    if idx<len(buff_twinX.index)-1:
        main_poly = buff_twinX.iloc[[idx]]
        other_poly = buff_twinX.iloc[(idx+1):]
        share_space = gpd.overlay(main_poly, other_poly, how='intersection')
        share_space['share_space'] = share_space.geometry.area
        shared_space_list.append(share_space)
df = pd.concat(shared_space_list)
```

Supplemental material 3. Python code for Figure 1 and Figure 2

```
# ----------------------------------------------------------
# Import libraries
# ----------------------------------------------------------
import pandas as pd
import geopandas as gpd
from gps2space import geodf
from gps2space import space
from gps2space import dist

import matplotlib.pyplot as plt
from matplotlib.lines import Line2D
from matplotlib.patches import Patch

print('Pandas version is:', pd.__version__)
print('Geopandas version is:', gpd.__version__)

# ----------------------------------------------------------
# Figure 1
# ----------------------------------------------------------
# Read data
us = gpd.read_file('./data/state2163.shp')
co = gpd.read_file('./data/co_county2163.shp')

twin_us = gpd.read_file('./data/twin_dist_US.shp')
twin_co_2016 = gpd.read_file('./data/co2016_2163.shp')
twin_co_2017 = gpd.read_file('./data/co2017_2163.shp')
twin_co_2018 = gpd.read_file('./data/co2018_2163.shp')

# Plot twin distribution
fig, ((ax1, ax2, ax3), (ax4, ax5, ax6)) = plt.subplots(2, 3, figsize=(20,10))

us.boundary.plot(ax=ax1, facecolor='grey', edgecolor='black', linewidth=0.3, zorder=1)
twin_us.loc[twin_us['year']==2016].plot(ax=ax1, color='white', markersize=0.3, zorder=2)
ax1.set_title("(a) Distribution of the twins' geolocations in the US in 2016")

us.boundary.plot(ax=ax2, facecolor='grey', edgecolor='black', linewidth=0.3, zorder=1)
twin_us.loc[twin_us['year']==2017].plot(ax=ax2, color='white', markersize=0.3, zorder=2)
ax2.set_title("(b) Distribution of the twins' geolocations in the US in 2017")

us.boundary.plot(ax=ax3, facecolor='grey', edgecolor='black', linewidth=0.3, zorder=1)
twin_us.loc[twin_us['year']==2018].plot(ax=ax3, color='white', markersize=0.3, zorder=2)
ax3.set_title("(c) Distribution of the twins' geolocations in the US in 2018")

co.boundary.plot(ax=ax4, facecolor='grey', edgecolor='black', linewidth=0.3, zorder=1)
twin_co_2016.plot(ax=ax4, color='white', markersize=0.3, zorder=2)
ax4.set_title("(d) Distribution of the twins' geolocations in CO in 2016")
```

```
co.boundary.plot(ax=ax5, facecolor='grey', edgecolor='black', linewidth=0.3, zorder=1)
twin_co_2017.plot(ax=ax5, color='white', markersize=0.3, zorder=2)
ax5.set_title("(e) Distribution of the twins' geolocations in CO in 2017")

co.boundary.plot(ax=ax6, facecolor='grey', edgecolor='black', linewidth=0.3, zorder=1)
twin_co_2018.plot(ax=ax6, color='white', markersize=0.3, zorder=2)
ax6.set_title("(f) Distribution of the twins' geolocations in CO in 2018")

for ax in fig.get_axes():
    ax.axes.xaxis.set_visible(False)
    ax.axes.yaxis.set_visible(False)

fig.subplots_adjust(hspace=0.0, wspace=0.1)
plt.savefig('./Figure_1.png', bbox_inches='tight', dpi=600)

# ---------------------------------------------------------
# Figure 2
# ---------------------------------------------------------
# Load csv and convert to spatial data
df_twinX_a = pd.read_csv('./data/Twin8a_512.csv')
df_twinX_b = pd.read_csv('./data/Twin8b_512.csv')

gdf_twinX_a = geodf.df_to_gdf(df_twinX_a, x='longitude', y='latitude')
gdf_twinX_b = geodf.df_to_gdf(df_twinX_b, x='longitude', y='latitude')

# Project spatial data
gdf_twinX_a = gdf_twinX_a.to_crs('epsg:2163')
gdf_twinX_b = gdf_twinX_b.to_crs('epsg:2163')

# Calculate buffer- and convex hull-based activity space
buff_twinX_a = space.buffer_space(gdf_twinX_a, dist=1000, dissolve='day', proj=2163)
buff_twinX_b = space.buffer_space(gdf_twinX_b, dist=1000, dissolve='day', proj=2163)

convex_twinX_a = space.convex_space(gdf_twinX_a, group='day', proj=2163)
convex_twinX_b = space.convex_space(gdf_twinX_b, group='day', proj=2163)

# Calculate shared space
buff_share = gpd.overlay(buff_twinX_a, buff_twinX_b, how='intersection')
convex_share = gpd.overlay(convex_twinX_a, convex_twinX_b, how='intersection')

buff_share['share_buffer'] = buff_share['geometry'].area
convex_share['share_convex'] = convex_share['geometry'].area

# Buffer-based activity space to SQUARE MILES
buff_twinX_a['act_buffer_mi'] = buff_twinX_a['buff_area'] * 0.0000003861
buff_twinX_b['act_buffer_mi'] = buff_twinX_b['buff_area'] * 0.0000003861
```

```
# Convex hull-based activity space to SQUARE MILES
convex_twinX_a['act_convex_mi'] = convex_twinX_a['convex_area'] * 0.0000003861
convex_twinX_b['act_convex_mi'] = convex_twinX_b['convex_area'] * 0.0000003861

# Shared space to SQUARE MILES
buff_share['share_buffer_mi'] = buff_share['share_buffer'] * 0.0000003861
convex_share['share_convex_mi'] = convex_share['share_convex'] * 0.0000003861

buff_twinX_a[['act_buffer_mi']]
buff_twinX_b[['act_buffer_mi']]

convex_twinX_a[['act_convex_mi']]
convex_twinX_b[['act_convex_mi']]

buff_share[['share_buffer_mi']]
convex_share[['share_convex_mi']]

# Plot activity space and shared space
fig, (ax1, ax2) = plt.subplots(1, 2, figsize=(25, 12))

gdf_twinX_a.plot(ax=ax1, marker='o', markersize=8, c='g')
gdf_twinX_b.plot(ax=ax1, marker='^', markersize=8, c='r')
buff_twinX_a.boundary.plot(ax=ax1, edgecolor='g', linewidth=2)
buff_twinX_b.boundary.plot(ax=ax1, edgecolor='r', linewidth=2)
buff_share.boundary.plot(ax=ax1, facecolor='grey', alpha=0.4)

ax1.axes.xaxis.set_visible(False)
ax1.axes.yaxis.set_visible(False)
ax1.set_title('(a) Buffer-based activity space\n and shared space for TwinX on May 12, 2017')

gdf_twinX_a.plot(ax=ax2, marker='o', markersize=8, c='g')
gdf_twinX_b.plot(ax=ax2, marker='^', markersize=8, c='r')
convex_twinX_a.boundary.plot(ax=ax2, edgecolor='g', linewidth=2)
convex_twinX_b.boundary.plot(ax=ax2, edgecolor='r', linewidth=2)
convex_share.boundary.plot(ax=ax2, facecolor='grey', alpha=0.6)

ax2.axes.xaxis.set_visible(False)
ax2.axes.yaxis.set_visible(False)
ax2.set_title('(b) Convex hull-based activity space\n and shared space for TwinX on May 12, 2017')

leg_ax1 = [Line2D([0],[0], marker='o', color='w', label='Geolocation_TwinXa', markerfacecolor='g', markersize=8),
                  Line2D([0],[0], marker='^', color='w', label='Geolocation_TwinXb', markerfacecolor='r',
markersize=8),
                  Line2D([0],[0], color='green', lw=2, label='Activity space_TwinXa (10.32 $mi^{2}$)'),
                  Line2D([0],[0], color='red', lw=2, label='Activity space_TwinXb (12.54 $mi^{2}$)'),
                  Patch(facecolor='grey', alpha=0.6, label='Shared space (8.08 $mi^{2}$)')]
```

```
leg_ax2 = [Line2D([0],[0], marker='o', color='w', label='Geolocation_TwinXa', markerfacecolor='g', markersize=8),
                  Line2D([0],[0], marker='^', color='w', label='Geolocation_TwinXb', markerfacecolor='r',
markersize=8),
                  Line2D([0],[0], color='green', lw=2, label='Activity space_TwinXa (8.99 $mi^{2}$)'),
                  Line2D([0],[0], color='red', lw=2, label='Activity space_TwinXb (11.08 $mi^{2}$)'),
                  Patch(facecolor='grey', alpha=0.6, label='Shared space (8.48 $mi^{2}$)')]

ax1.legend(handles=leg_ax1, loc='upper right', fontsize=10)
ax2.legend(handles=leg_ax2, loc='upper right', fontsize=10)

fig.subplots_adjust(wspace=0.0, hspace=0.0, right=0.6)
plt.savefig('./Figure_2.png', bbox_inches='tight', dpi=600)
```

Supplemental material 4. Python code for Table 1

```
# ---------------------------------------------------------
# Import libraries
# ---------------------------------------------------------
import pandas as pd
import geopandas as gpd
from gps2space import geodf
from gps2space import space
from gps2space import dist

print('Pandas version is:', pd.__version__)
print('Geopandas version is:', gpd.__version__)

# ---------------------------------------------------------
# Read twin data
# ---------------------------------------------------------
twin_df = pd.read_csv('./data/Twin8_CO.csv')
twin_gdf = geodf.df_to_gdf(twin_df, x='longitude', y='latitude')

# ---------------------------------------------------------
# Read park, playground, and supermarket data
# ---------------------------------------------------------
park_poly = gpd.read_file('./data/co_park_poly_2163.shp')
playground_poly = gpd.read_file('./data/co_playground_poly_2163.shp')
market_poly = gpd.read_file('./data/co_market_poly_2163.shp')

park_point = gpd.read_file('./data/co_park_point_2163.shp')
playground_point = gpd.read_file('./data/co_playground_point_2163.shp')
marker_point = gpd.read_file('./data/co_market_point_2163.shp')

# ---------------------------------------------------------
# Distance to point
# ---------------------------------------------------------
dist2point_park = dist.dist_to_point(twin_gdf, co_park_point, proj=2163)
dist2point_park['dist_mile'] = dist2point_park['dist2point'] * 0.000621371

dist2point_playground = dist.dist_to_point(twin_gdf, co_playground_point, proj=2163)
dist2point_playground['dist_mile'] = dist2point_playground['dist2point'] * 0.000621371

dist2point_market = dist.dist_to_point(twin_gdf, co_market_point, proj=2163)
dist2point_market['dist_mile'] = dist2point_market['dist2point'] * 0.000621371

# ---------------------------------------------------------
# Distance to polygon
# ---------------------------------------------------------
dist2poly_park = dist.dist_to_poly(twin_gdf, co_park, proj=2163)
dist2poly_park['dist_mile'] = dist2poly_park['dist2poly'] * 0.000621371
```

```
dist2poly_playground = dist.dist_to_poly(twin_gdf, co_playground, proj=2163)
dist2poly_playground['dist_mile'] = dist2poly_playground['dist2poly'] * 0.000621371

dist2poly_market = dist.dist_to_poly(twin_gdf, co_market, proj=2163)
dist2poly_market['dist_mile'] = dist2poly_market['dist2poly'] * 0.000621371

# -----------------------------------------------------------
# Description to dataframe
# -----------------------------------------------------------
df_point_park = pd.DataFrame({'park_point': dist2point_park['dist_mile'].describe()})
df_point_playground = pd.DataFrame({'playground_point': dist2point_playground['dist_mile'].describe()})
df_point_market = pd.DataFrame({'market_point': dist2point_market['dist_mile'].describe()})

df_poly_park = pd.DataFrame({'park_poly': dist2poly_park['dist_mile'].describe()})
df_poly_playground = pd.DataFrame({'playground_poly': dist2poly_playground['dist_mile'].describe()})
df_poly_market = pd.DataFrame({'market_poly': dist2poly_market['dist_mile'].describe()})

dataframe_list = [df_point_park, df_point_playground, df_point_market, df_poly_park, df_poly_playground,
df_poly_market]
df = pd.concat(dataframe_list, axis=1)
df.describe().T
```

Supplemental material 5. R code for Table 2 and Table 3

```r
# ------------------------------------------------------------
# Load brms package for Bayesian regression modeling
# ------------------------------------------------------------
library(brms)

# ------------------------------------------------------------
# Model for activity space
# ------------------------------------------------------------
a_model <- brm(AS ~ Age*Gender+BaselineAge+Weekend+Summer+Fall+Winter+
              (Age|family:user_id)+(Age*Gender+BaselineAge|family),
              data = data,
              warmup = 2000,
              iter = 5000,
              chains = 2,
              inits = "random",
              cores = 2,
              seed = 123)

# ------------------------------------------------------------
# Check estimation results
# ------------------------------------------------------------
summary(a_model)

# ------------------------------------------------------------
# Model for shared space
# ------------------------------------------------------------
s_model <- brm(PSS ~ Age*Gender+DZSS*Gender+DZOS*Gender+
               DZSS*BaselineAge+DZOS*BaselineAge+
               Weekend+Summer+Fall+Winter+
              (Age|family:user_id)+(Age*Gender+BaselineAge|family),
              data = data,
              warmup = 2000,
              iter = 5000,
              chains = 2,
              inits = "random",
              cores = 2,
              seed = 123,
              family = Beta(link="logit"))
# ------------------------------------------------------------
# Check estimation results
# ------------------------------------------------------------
summary(s_model)
```

Table S1 Parameter estimates of the growth curve model on (log-transformed) activity space based on full data from the CoTwins study, 2016-2018

Parameter	Estimate	SE	95% CI
Fixed effects			
Intercept, δ_{000}	1.78	0.03	[1.73, 1.83]
Gender, δ_{010}	-0.07	0.02	[-0.12, -0.01]
Baseline age, δ_{020}	0.13	0.02	[0.09, 0.17]
Age, δ_{100}	-0.05	0.02	[-0.09, 0.00]
Age*Gender, δ_{110}	0.01	0.02	[-0.03, 0.04]
Weekend, β_2	0.06	0.00	[0.05, 0.07]
Summer, β_3	0.06	0.00	[0.05, 0.07]
Fall, β_4	-0.13	0.01	[-0.14, -0.12]
Winter, β_5	-0.09	0.01	[-0.10, -0.08]
Level-2 random effects			
Intercept standard deviation, τ_0	0.30	0.02	[0.27, 0.34]
Age standard deviation, τ_1	0.28	0.02	[0.25, 0.32]
Intercept-Age correlation, $\tau_{01}/(\tau_0 * \tau_1)$	-0.23	0.08	[-0.38, -0.07]
Level-3 random effects			
Intercept standard deviation, φ_0	0.38	0.03	[0.33, 0.43]
Age standard deviation, φ_3	0.19	0.04	[0.10, 0.25]
Residual standard deviation, σ	0.72	0.00	[0.71, 0.72]

Note: SE = standard errors estimated by standard deviations of the posterior samples; CI = credible interval. N = 561 participants. The number of time points for each participant ranged from 3 to 569.

Table S2 Parameter estimates of the growth curve model on the proportion of shared space (PSS) based on full data from the CoTwins study, 2016-2018

Parameter	Estimate	SE	95% CI
Fixed effects			
Intercept, δ_{000}	0.72	0.08	[0.56, 0.88]
Gender, δ_{010}	0.02	0.08	[-0.13, 0.17]
Baseline age, δ_{020}	-0.30	0.06	[-0.42, -0.18]
Age, δ_{100}	-0.34	0.03	[-0.41, -0.28]
Age*Gender, δ_{110}	-0.04	0.03	[-0.10, 0.02]
DZSS, δ_{001}	-0.29	0.11	[-0.49, -0.08]
DZOS, δ_{002}	-0.51	0.12	[-0.74, -0.28]
DZSS*Gender, δ_{011}	0.04	0.10	[-0.17, 0.23]
DZOS*Gender, δ_{012}	0.09	0.09	[-0.08, 0.26]
DZSS*Baseline age, δ_{021}	-0.12	0.08	[-0.27, 0.03]
DZOS*Baseline age, δ_{022}	-0.04	0.09	[-0.22, 0.15]
Weekend, β_2	-0.12	0.01	[-0.14, -0.11]
Summer, β_3	-0.29	0.01	[-0.31, -0.28]
Fall, β_4	-0.44	0.01	[-0.46, -0.42]
Winter, β_5	-0.09	0.01	[-0.11, -0.07]
Level-2 random effects			
Intercept standard deviation, τ_0	0.37	0.02	[0.32, 0.42]
Age standard deviation, τ_1	0.36	0.03	[0.30, 0.42]
Intercept-Age correlation, $\tau_{01}/(\tau_0 * \tau_1)$	-0.13	0.09	[-0.31, 0.05]
Level-3 random effects			
Intercept standard deviation, φ_0	0.58	0.04	[0.51, 0.66]
Age standard deviation, φ_3	0.33	0.04	[0.25, 0.41]
Dispersion parameter, ϕ	1.86	0.01	[1.84, 1.87]

Note: SE = standard errors estimated by standard deviations of the posterior samples; CI = credible interval. N = 498 participants (or 249 pairs of twins). The number of time points for each participant ranged from 3 to 569.