Example 2 using python

Time to event from parametric hazards

David Garibay, M.P.P.* Hawre Jalal, MD, Ph.D.†
Fernando Alarid-Escudero, Ph.D.‡§

Code function

This document presents the python code corresponding to the second example presented in the "A Fast Nonparametric Sampling (NPS) Method for Time-to-Event in Individual-Level Simulation Models." manuscript.

```
# 01 Initial Setup ------------------------------------------------------

# Import required modules
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import scipy.stats as stats
import pandas as pd
```

```
# 02 Load base data -----------------------------------------------------

df_mort_data_raw = pd.read_csv(
  filepath_or_buffer = "../data/all_cause_mortality.csv")
```

*Health Research Consortium (CISIDAT), Cuernavaca, Morelos, Mexico.
†School of Epidemiology and Public Health, Faculty of Medicine, University of Ottawa, Ottawa, ON, CA.
‡Department of Health Policy, Stanford University School of Medicine, Stanford, CA, USA.
§Center for Health Policy, Freeman Spogli Institute, Stanford University, Stanford, CA, USA.

```
# 03 Data wrangling ----------------------------------------------------

# Keep data only for year 2015
df_mort_data_1 = df_mort_data_raw.query("Year == 2015").copy()

# Reset index (row ennumeration) of new data set
df_mort_data_1.reset_index(drop = True, inplace = True)

# Arrange data by sex, year and age values
df_mort_data_1.sort_values(by = ["Sex", "Year", "Age"], inplace = True)

# Group by sex
df_grouped = df_mort_data_1.groupby('Sex')

# Steps to derive the Instantaneous probability from the cumulative hazard
# H(t) - Cumulative hazard
df_mort_data_1['H_t'] = df_grouped['Rate'].cumsum()
# S(t) - Cumulative survival
df_mort_data_1['S_t'] = df_mort_data_1['H_t'].apply(lambda x: np.exp(-x))
# F(t) - Cumulative probability: 1 - S(t)
df_mort_data_1['F_t'] = 1 - df_mort_data_1['S_t']
# f(t) - Instantaneous probability
df_mort_data_1['p_t'] = (df_mort_data_1.groupby('Sex')['F_t'].
  diff().fillna(df_mort_data_1['F_t']))

# #* Check sum of probabilities
# df_mort_data_1.groupby(["Sex"])["p_t"].sum()

#* It is incomplete, so we will create a data frame to fill the probabilities
#* for each group
df_mort_append_0 = df_mort_data_1.groupby(["Sex"]).tail(1)

df_mort_append_0.loc[:, "Age"] = 101
df_mort_append_0.loc[:, "p_t"] = 1 - df_mort_append_0["F_t"]
df_mort_append_0.loc[:, "F_t"] = df_mort_append_0[["F_t", "p_t"]].sum(axis=1)
df_mort_append_0.loc[:, "S_t"] = 1 - df_mort_append_0["F_t"]
df_mort_append_0.loc[:, "H_t"] = np.nan
df_mort_append_0.loc[:, ["H_t", "Rate"]]  = np.nan


# Concatenate the original and the extra dataframes
df_mort_data_2 = pd.concat([df_mort_data_1, df_mort_append_0])
```

```
df_mort_data_2.sort_values(by = ["Sex", "Year", "Age"], inplace = True)
df_mort_data_2.reset_index(drop = True, inplace = True)

# # Now the sum of probs adds up to 1 for all groups
# df_mort_data_2.groupby(["Sex"])["p_t"].sum()

#* Convert into wide format
#* - Year will be discarded while turning data into wide format
df_mort_data_2_wide = df_mort_data_2.pivot(
  columns = "Age",
  index   = ["Year", "Sex"],
  values  = "p_t")

# Pass index into columns
df_mort_data_2_wide.reset_index(inplace = True)

# Remove row-axis name
df_mort_data_2_wide.rename_axis(None, axis = 1, inplace = True)
```

```
# 04 Sample times to events ---------------------------------------------
# We will use the "Total" Sex category to sample 100,000 individuals
a_age_values = np.arange(0, 102)
n_samples = int(1e5)

# Extract probability distribution for "Total" sex category
df_prob_values = (df_mort_data_2_wide.
  loc[df_mort_data_2_wide["Sex"] == "Total", 0:101])

# Convert into an array
a_prob_values = df_prob_values.iloc[0].to_numpy()

# # Plot probability distribution
# plt.rcParams["font.size"] = "22"
# plt.plot(a_age_values, a_prob_values, linewidth = 4)
# plt.title("Probability mass function of death by age in Total population,
#  2015")
# plt.grid(True, linestyle = "dotted", linewidth = 2)
# plt.show()
# plt.close()


# Instantiate base Data frame to fill with sampled ages of death
```

```
df_test_total = pd.DataFrame(
  data = {"Year": np.repeat(2015, n_samples), "Sex": "Total"})


# Set seed for reproducibility
np.random.seed(seed = 1234) # Set seed for reproducibility

#* Draw age of death from life tables using the `np.random.choice` function
df_test_total["age_death"] = np.random.choice(
  a       = a_age_values,
  size    = n_samples,
  replace = True,
  p       = a_prob_values)

#* Add random number between 0 and 1 to approximate continous time
df_test_total["age_death_corr"] = (df_test_total["age_death"] +
  np.random.random_sample(n_samples))

# Remove seed
np.random.seed(seed = None)
```