```
# From Bayesian Models for Astrophysical Data
# by Hilbe, de Souza & Ishida, 2017, Cambridge Univ. Press

# Code 7.2 - Bayesian zero-inflated Poisson in Python using Stan
# 1 response (y) and 1 explanatory variable (x1)
#

import numpy as np
import pystan
import statsmodels.api as sm

from rpy2.robjects import r, FloatVector
from scipy.stats import uniform, norm


def zipoisson(N, lambda_par, psi):
    """Zero inflated Poisson sampler."""

     # load R package
    r('library(VGAM)')

    # get R functions
    zipoissonR = r['rzipois']

    res = zipoissonR(N, FloatVector(lambda_par),
                     pstr0=FloatVector(psi))

    return np.array([int(item) for item in res])


# Data
np.random.seed(141)                         # set seed to replicate example
nobs= 5000                                  # number of obs in model

x1 = uniform.rvs(size=nobs)

xb = 1 + 2.0 * x1                           # linear predictor, xb
xc = 2 - 5.0 * x1

exb = np.exp(xb)
exc = 1.0 / (1.0 + np.exp(-xc))

zipy = zipoisson(nobs, exb, exc)            # create y as adjusted

X = np.transpose(x1)
X = sm.add_constant(X)

mydata = {}                                 # build data dictionary
mydata['N'] = nobs                          # sample size
mydata['Xb'] = X                            # predictors
mydata['Xc'] = X
mydata['Y'] = zipy                          # response variable
mydata['Kb'] = X.shape[1]                   # number of coefficients
mydata['Kc'] = X.shape[1]

# Fit
stan_code = """
data{
    int N;
    int Kb;
    int Kc;
    matrix[N, Kb] Xb;
    matrix[N, Kc] Xc;
    int Y[N];
}
parameters{
    vector[Kc] beta;
    vector[Kb] gamma;

}
transformed parameters{
    vector[N] mu;
```

```
    vector[N] Pi;

    mu = exp(Xc * beta);
    for (i in 1:N) Pi[i] = inv_logit(Xb[i] * gamma);
}
model{
    real LL[N];

    for (i in 1:N) {
        if (Y[i] == 0) {
            LL[i] = log_sum_exp(bernoulli_lpmf(1|Pi[i]),
                                bernoulli_lpmf(0|Pi[i]) +
                                poisson_lpmf(Y[i]|mu[i]));
        } else {
            LL[i] = bernoulli_lpmf(0|Pi[i]) +
                                poisson_lpmf(Y[i]|mu[i]);
        }
    }

    target += LL;
}
"""

# Run mcmc
fit = pystan.stan(model_code=stan_code, data=mydata, iter=5000, chains=3,
                  warmup=4000, n_jobs=3)

# Output
nlines = 9                                    # number of lines in screen output

output = str(fit).split('\n')
for item in output[:nlines]:
    print(item)
```