Dppsampl Package

01 January 2021

Description:

This package provides algorithms for sampling from the Determinantal Point Processes (DPPs) for SAS.

Authors:

Ada Gassowska <ada.gassowska.stud@pw.edu.pl>
Katarzyna Solawa <katarzyna.solawa.stud@pw.edu.pl>
Kacper Kurowski <kacper.kurowski.dokt@pw.edu.pl>

Version: 0.5

Modules:

- sample_from_beta_ensemble_full Function
- sample_from_beta_ensemble_banded Function

Overview:

The `dppsampl` package provides an implementation of the algorithms for sampling from the Determinanatal Point Processes (DPPs) written natively in SAS. DPPs are stochastic point processes (so, their result is a subset of that set) which respect the diversity present in the set.

The package offers ways to sample from Finite, Continuous and the so-called Exotic DPPs. For Finite DPPs the set of possible values is finite, for continuous it is infinite and the samples should approximate some continuous distribution. The exotic DPPs however, are a special kind of DPPs which were analyzed for different reasons and one needed a change-of-perspective to deduce that they can be thought of as DPPs.

The available continuous DPPs come from a family of distributions called Beta ensembles. They were initally studied in Physics, as they can be thought of as models of a Coulomb gas. [1].

Before you use the `dppsampl` package, you must install it by using the PACKAGE INSTALL statement. For example, if the ZIP file is located in the directory C:\Packages, then the following statement installs the package:

```
proc iml;
        package install "C:\Packages\dppsampl.zip";
quit;
```

Data Sets

Currently our package does not provide any datasets.

sample_from_beta_ensemble_full Function

Syntax

```
sample_from_beta_ensemble_full(
        ensemble_version,
        M_1, M_2, /* M variables only for Laguerre (first) and Jacobi (both) */
      size=10,
```

```
beta=2,
normalize=1,
      haar_mode="Hermite", /* haar_mode only available for circular ensemble */
 heurestic_fix=1, /* heurestic_fix only available for circular ensemble */
random_state=1618);
```

Parameters

ensemble_version — Version of Beta ensemble to use. Available values are “Hermite”, “Laguerre”, “Jacobi”, “Circular”, and “Ginibre”.

M_1 — *Distribution parameter for the “Laguerre” and “Jacobi” ensemble_version s . Should be greater or equal to Size.*

M_2 — *Distribution parameter for the “Jacobi” ensemble_version . Should be greater or equal to Size.*

Size — *Size of the of the sampled subset.*

Beta — *Beta parameter. Should be 1, 2, or 4.*

Normalize — *Parameter which states whether the sample should be normalized to fit one of the more known distributions.*

Haar_Mode — *Which Haar measure mode to use. Can be “Hermite” or “QR”. Influences the result only for the Circular Ensemble. (Should be 1 or 0).*

Heurestic_Fix — *Whether to apply the heuresis to fix the oversampling problem present in the Circular and Ginibre ensembles. Should be 1 or 0.*

Random_State — *Seed for the randomness. Should be a positive integer.*

Description

The function provides the method for sampling from beta ensemble using the full matrix method. There are five versions of Beta ensembles that have been implemented. “Hermite”, “Laguerre”, “Jacobi”, “Circular”, and “Ginibre”. For the first three, the result is a one-column sample. For the next two, it is a two-column sample.

Example

```
ensemble_version = "Circular";
size=10;
beta=4;
M_1=10; M_2 = 10;
haar_mode="Hermite";
normalize=0;
heurestic_fix=0;
random_state=1618;


sample = sample_from_beta_ensemble_full(
```

```
          ensemble_version,
          M_1, M_2,
          size,
       beta,
       normalize,
       haar_mode,
      heurestic_fix,
     random_state);

run scatter(sample[,1], sample[,2]);
```

sample_from_beta_ensemble_banded Function

Syntax

```
sample_from_beta_ensemble_banded(
        ensemble_version,
       size=10,
       beta=2,
       loc=0.0,
        scale=1.0,
          shape = 1.0,
           a = 1.0,
          b = 1.0,
          normalize=1,
         heurestic_fix=1,
            random_state=1618);
```

Parameters

ensemble_version — Version of Beta ensemble to use. Available values are "Hermite", "Laguerre", "Jacobi", and "Ginibre".

Size — *Size of the of the sampled subset.*

Beta — *Beta parameter. Should be 1, 2, or 4.*

Loc — *Location parameter for the standard deviation for the Hermite Beta ensemble.*

Scale — *Scale parameter for the expected value for the Hermite and Laguerre Beta ensemble.*

Shape — *Shape parameter for the Laguerre Beta ensemble.*

a — *a Parameter for the Jacobi Beta ensemble. Related to the Beta distribution.*

b — *b Parameter for the Jacobi Beta ensemble. Related to the Beta distribution.*

Normalize — *Parameter which states whether the sample should be normalized to fit one of the more known distributions.*

Heurestic_Fix — *Whether to apply the heuresis to fix the oversampling problem present in the Circular and Ginibre ensembles. Should be 1 or 0.*

Random_State — *Seed for the randomness. Should be a positive integer.*

Description

The function provides the method for sampling from beta ensemble using the bandedmatrix method. There are five versions of Beta ensembles that have been implemented. “Hermite”, “Laguerre”, “Jacobi”, and “Ginibre”. For the first three, the result is a one-column sample. For the Ginibre ensemble, it is a two-column sample.

Example

```
ensemble_version = "Hermite";
size=1000;
beta=2;
loc=0.0;
scale=1.0;
shape = 1.0;
a = 1.0;
b = 1.0;
normalize=0;
heurestic_fix=0;
random_state=1618;
sample = sample_from_beta_ensemble_banded(
          ensemble_version,
           size,
           beta,
           loc,
          scale,
           shape,
           a,
            b,
             normalize,
             heurestic_fix,
         random_state);


run histogram(sample);
```

References:

[1] *Peter Forrester, Beta Ensembles: Universality, Integrability, and Asymptotics,* (2016)