1 Instructions to run the R codes

1. `BMSE.utility.functions.R`: It has user defined functions such as log-likelihood function (viz. `logLfn.piscr(), logLfn.rpiscr()`) , Euclidean distance between two points (viz. `e2dist()`), function which provides a good choice for initial value of 'L' vector (viz. `Lvec.fn()`) etc. The required R packages are - 'mvtnorm', 'MCMCpack', 'scrbook', 'abind', 'ggplot2', 'Brobdingnag', 'parallel', 'snowfall', 'rlecuyer', 'stargazer'. It also contains a function `sim.partial.data()` which simulates partially identified capture history data under Dey et al. (in press). The user needs to specify the input arguments to simulate. Details on the input arguments are given in Table 1.

Table 1: *Input arguments in* `sim.partial.data()`.

Argument	**Definition**
`N`	Population size or abundance.
`N.Male`	Number of male individuals in the population.
`phi`	Probability that an individual i is detected by a detector on some occasion k given that it is present at that trap.
`omega0`	Baseline trap entry probability, i.e., probability that an individual passes through a trap station assuming its centre of activity is also located at that trap station.
`sigma`	`sigma` measures the spatial extent of movement around individual activity centre. `sigma = (sigmam, sigmaf)` where `sigmam` corresponds to male individuals and `sigmaf` corresponds to female individuals. It is a vector of size 2×1.
`Msexsigma`	A binary variable. If `Msexsigma` $= 1$, sigma is modelled as a function of indiviudal sex covariate where `sigmam` measures the spatial extent of movement for the male individuals and `sigmaf` measures the spatial extent of movement for female individuals. If `Msexsigma` $= 0$, spatial extent of all the individuals will be measured by `sigmam`. So in this case, value of `sigmaf` will not effect the analysis (and any value can be specified for `sigmaf`).
`nrow_trap, ncol_trap`	The number of trap stations in each row and column in a rectangular trapping array, respectively.
`nocc`	Number of sampling occasions.
`xlim, ylim`	Start and end points of the horizontal and vertical axes of a rectangular state space.
`buffer`	Distance from the each side of the trapping array to the boundary of the state space.

2. `BMSE.PISCRfn.R`: It contains a function `PISCRfn()` which analyses SCR data under the model developed in Dey et al. (in press). If `Msexsigma` $= 1$, σ is modelled as a function of the sex category of the individual; i.e., $\sigma = \sigma_m$ if the individual is male, $\sigma = \sigma_f$ if the individual is female. If `Msexsigma` $= 0$, σ is a scalar. MCMC samples from the posterior distributions of the parameters are generated and saved in local files, together with summary statistics. Please refer to Table 2 for details on the input arguments.

3. `BMSE.Royle.PISCRfn.R`: It contains a function `RPISCRfn()` which analyses SCR data under the model developed in Royle (2015). Following `BMSE.PISCRfn.R`, if `Msexsigma` $= 1$, σ is modelled as a function of the sex category of the individual; i.e., σ_m for males, σ_f for females. If

`Msexsigma` $= 0$, σ is a scalar. MCMC samples from the posterior distributions of the parameters are generated and saved in local files, together with summary statistics. Please refer to Table 2 for details on the input arguments.

Table 2: *Input arguments in* ***PISCRfn()*** *and* ***RPISCRfn()****.*

Argument	**Definition**
`ndraws`	Number of MCMC iterations
`burnin`	Burn-in period of MCMC iterations
`M`	Maximum number of individuals within the state space.
`scale`	Scaling factor for the UTM coordinates.
`nloopL`	Number of iterations in optimizing the initial value of L vector.
`n.update`	Number of MCMC updates for $\mathbf{L}$ vector.
`batchsize`	The size of a batch to update proposal variance.
`mindelta`	Reducing factor of proposal variance.
`sigmaOfProposal.logitphi`	Proposal variance of logit(ϕ).
`sigmaOfProposal.logitomega0`	Proposal variance of logit(ω_0).
`sigmaOfProposal.logitp0`	Proposal variance of logit(p_0).
`sigmaOfProposal.logsigmam`	Proposal variance of log(σ_m).
`sigmaOfProposal.logsigmaf`	Proposal variance of log(σ_m).
`sigmaOfProposal.L`	Proposal variance of required to update L vector.
`sigmaOfProposal.s`	Proposal variance of $\mathbf{S}$. It is a vector of length M.
`dd`	Maximum permissible value of movement range for each individual during the study.
`piscrobj`	A list object. It contains the output from the function `sim.partial.data()` with following variables: edf1 (encounter data file from detector 1), edf2 (encounter data file from detector 2), sex.data (data on individual sexes from both the two detectors), tdf (trap deployment data), xlim, ylim, buffer, ntrap, nocc. Details on `edf1`, `edf2`, `sex.data` and `tdf` are given in Section 1.1.
`SimstudyIndicator`	A binary variable. If `SimstudyIndicator = 1`, it implies that the user is using the output of `sim.partial.data()` function as the input `piscrobj`, where the true values of the parameters are known from the specifications. If `SimstudyIndicator = 0`, it implies that the true values of the parameters are not known and the user have manually stored the observed data as the input of `PISCRfn()` and `RPISCRfn()` function.

4. `BMSE.intlik.u0zs.waic.R`: This R code computes (i) the integrated likelihood after integrating the likelihood with respect to the prior distribution of the parameters of $\mathbf{u_0}$, $\mathbf{z}$ and $\mathbf{S}$, (ii) computes the individual specific likelihood for each MCMC iteration. It should be run after `GDBF.IL.R`, `BMSE.DIC.R` and `BMSE.WAIC.R` as values of the likelihood and the integrated likelihood will be needed to compute Bayes factor using IL method, DIC and WAIC.

5. `GDBF.MAP.R`: This R code computes the Gelfand-Dey estimator of marginal likelihood of data with the MAP approximation approach. It also computes the harmonic mean estimator.

6. `GDBF.IL.R`: This R code computes the Gelfand-Dey estimator of marginal likelihood of data with the integrated likelihood approximation approach.

7. `BMSE.WAIC.R`: This R code computes the WAIC criterion.

8. `BMSE.DIC.R`: This R code computes the DIC criterion. It should be run after `GDBF.MAP.R` as `BMSE.DIC.R` uses the MAP indices from the output of the R code `GDBF.MAP.R`.

9. `BMSE.PPL.R`: This R code computes the posterior predictive loss criterion.

10. `BMSE.runPISCR.R`: This is an example code showing how to use the function `PISCRfn()` and `RPISCRfn()` for analysing partially identified detection history using models from Dey et al. (in press) and Royle (2015) respectively.

11. `MAPfn.R`: It includes a generic function named `MAPfn()` to compute *maximum a posteriori* (MAP) estimate for a set of parameters using their MCMC iterations by following the method described in Dey et al. (in press). Details of its input arguments are given in Table 1. Please ensure par1.chain and par2.chain are either a vector or a matrix or a data frame

Table 3: *Input arguments in* `MAPfn()`.

Argument	Definition
`loglikfn`	It is a function to compute loglikelihood. The arguments of loglikfn are data, par1, par2 (in that order), e.g., loglikfn(data, par1, par2)
`logpriorfn`	It is a function to compute prior density. The arguments of logpriorfn are par1, par2 (in that order), e.g., logpriorfn(par1, par2)
`loglik.chain`	it is a vector of loglikelihod values for each MCMC iterations
`logprior.chain`	it is vector of logprior density values for each MCMC iterations
`par1.chain`	a matrix of MCMC chain for par1 with chain of different parameters in different columns
`par2.chain`	a matrix of MCMC chain for par2 with chain of different parameters in different columns
`data`	It should be a single R object (e.g., list, matrix, data.frame etc.) which takes all the non-paramter elements (i.e., capture history, mask data, trap deployment data). and to be supplied as argument of likelihood function
`burnin`	burnin value of the MCMC chain. In case the chains are already truncated, please provide the value of 'burnin' as 0 (zero).

1.1 Description of Data

`PISCRfn()` and `RPISCRfn()` require the following data inputs:

1. Animal capture data coming from detector 1 (Session, animal ID, capture occasion, trap ID, status)

2. Animal capture data coming from detector 2 (Session, animal ID, capture occasion, trap ID, status)

3. Data on sexes of the captured individuals (Animal ID from the two detectors, their sexes)

4. Trap deployment data (Trap ID, location and deployment record)

INPUT DATA 1 and 2: Animal capture data from detector 1 and 2

Both the detector-1 and detector-2 capture data should 5 columns: session, the animal identity number, the sampling occasion number, Location Number and the status, in that order. Each of the two matrices take only numeric integer entries. Each captured individual captured should be given a unique identification number, ranging from 1 to n, where n is the total number of individuals included

in this file. n can be different between detector-1 file to detector-2 file. Note that, first IDfixed number of row entries in each of the two matrices should be same where 'IDfixed' is the number of captured individuals who are completely identified by the two detectors. The individuals who are completely identified are highlighted by the status '12' in the fifth column of each of the encounter data files. In the example below, the records have been sorted by 'Individual' to facilitate checking that no numbers are missing. Location numbers must correspond to the numbers in the 'Trap deployment file'. If only $n = 7$ and $n = 6$ animals were captured by detector 1 and 2, of which only the first 5 are fully identified (across the detectors), the detector-1 data and detector-2 data should look like the following: Note that, the status is given as '1' and '2' for the sixth individual in the detector 1 data

Table 4: Capture-recapture data

Detector 1 capture data

Session	Individual	Occasion	Trap	Status
1	1	14	11	12
1	2	12	15	12
1	3	20	9	12
1	4	17	1	12
1	5	13	5	12
1	6	17	7	1
1	7	9	6	1

Detector 2 capture data

Session	Individual	Occasion	Trap	Status
1	1	14	11	12
1	2	12	15	12
1	3	20	9	12
1	4	17	1	12
1	5	13	5	12
1	6	10	14	2

and detector 2 data respectively; this means that these two animals might not be the same and are partially identified. The first row of in each data tells us that individual-1 was captured at Location ID 11 on the 14th sampling occasion. The column headings of the two data files must be exactly the same as shown above, or it will not be recognized by `PISCRfn()` and `RPISCRfn()`.

INPUT DATA 3: Data on sexes of the captured individuals in detector 1 and 2

This file has four columns: 'Individual1', 'Sex1', 'Individual2' and 'Sex2'. The order of the individuals should be the same as given in the capture matrices for detector 1 and 2 respectively. The sexes should be specified as 'Male' and 'Female'. An example of such a file is given below :

Table 5: Data on sex category

Individual1	Sex1	Individual2	Sex2
1	Male	1	Male
2	Female	2	Female
3	Female	3	Female
4	Male	4	Male
5	Female	5	Female
6	Male	6	Female
7	Female	NA	NA

INPUT DATA 4: Trap deployment data

The first column of this file has the location number. The next two columns have the X (Easting / Longitude) and Y (Northing / Latitude) coordinates of the trap location (see the example below). After these first three columns, the next columns correspond to each trapping occasion. Here, '1' means that a trap was operating at that location on that occasion, '0' means that no trap was operating. In particular, trap malfunction, theft, vandalism, etc. can be accounted for. An example

trap deployment data is shown below. This file has 12 locations, but there were only 4 traps, which were moved between locations.

Table 6: Trap deployment data

TrapID	Easting	Northing	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	619303	1325966	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
2	624151	1325013	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
3	624722	1323864	1	1	1	0	1	0	0	0	0	0	0	0	0	0	0
4	621806	1322453	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
5	622451	1320137	0	0	0	0	0	1	1	1	1	1	0	0	0	0	0
6	622599	1317937	0	0	0	0	0	1	1	1	1	1	0	0	0	0	0
7	623179	1315941	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	625156	1315587	0	0	0	0	0	1	1	1	1	1	0	0	0	0	0
9	626022	1314224	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1
10	627568	1315494	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1
11	619604	1324739	0	0	0	0	0	0	0	0	0	0	1	1	0	1	1
12	621478	1324515	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1

The table shows that the first four locations were operational during sampling occasions 1-5, except that the trap at location 3 was not working on occasion 4. No traps were operational at these sites on occasions 6 to 15. Again, the first row must have column headings as shown in the example above.

2 Example

```
rm(list = ls())

source('BMSE.utility.functions.R')
source('BMSE.sim.R')
source('BMSE.PISCRfn.R')
source('BMSE.Royle.PISCRfn.R')

piscrobj = sim.partial.data(N = 100, N.Male = 40, phi = 0.5, omega0 = 0.005,
            sigma = c(0.3, 0.15), Msexsigma = 1, nrow_trap = 16, ncol_trap = 10,
            nocc = 50, xlim = c(0,5), ylim = c(0,5), buffer = 1)

PISCRfn(piscrobj, Msexsigma = 1, ndraws = 2000, burnin = 1000, thin  = 1,  M = 400,
          bignum = 10^200, folderpath = NA, nloopL = 50, n.update = 20, dd = 5,
          batchsize = 1000, mindelta = 0.01, sigmaOfProposal.logitphi = 0.05,
          sigmaOfProposal.logitomega0 = 0.08, sigmaOfProposal.logsigmam = 0.02,
          sigmaOfProposal.logsigmaf = 0.02, sigmaOfProposal.L = 4,
          sigmaOfProposal.s = rep(3, 400), \ttt{SimstudyIndicator} = 1)

RPISCRfn(piscrobj, Msexsigma = 1, ndraws = 2000, burnin = 1000, thin  = 1, M = 400,
          bignum = 10^200, folderpath = NA, nloopL = 50, n.update = 20, dd = 5,
          batchsize = 1000, mindelta = 0.01, sigmaOfProposal.logitp0 = 0.08,
          sigmaOfProposal.logsigmam = 0.02, sigmaOfProposal.logsigmaf = 0.02,
          sigmaOfProposal.L = 4, sigmaOfProposal.s = rep(3, 400))

source('BMSE.intlik.u0zs.waic.R')
source('GDBF.IL.R')
source('GDBF.MAP.R')
```

```
source('BMSE.DIC.R')
source('BMSE.WAIC.R')
source('BMSE.PPL.R')
```

2.1 Manual data input in `PISCRfn()` and `RPISCRfn()`

If the user is not using `sim.partial.data()` which simulates partially identified capture history data, he also can manually store the observed data set in a list object (viz. `piscrobj`) of R software (R Core Team, 2018) as the following.

```
piscrobj = list(edf1, edf2, sex.data, tdf, xlim, ylim, buffer)
```

Here `edf1` and `edf2` are encounter data files as explained in the Table 4, `sex.data` is observation of the sex categories of the captured individuals (a matrix exactly in the form of Table 5), `tdf` is trap deployment file which can be prepared by following Table 6. `xlim` and `ylim` are the boundary limit of the state space along x-axis and y-axis (e.g., `xlim = c(0, 5), ylim = c(0,7)`) `buffer` denotes the distance between boundary the trap array and boundary of the state space. It is to be noted that when the user is manually preparing the list object (`piscrpj`) for the observed capture-recapture data, `SimstudyIndicator` should be specified as 0 while running the function `PISCRfn()` and `RPISCRfn()`.

References

Dey, S., Delampady, M., Karanth, K. U., and Gopalaswamy, A. M. (in press). A spatially explicit capture recapture model for partially identified individuals when trap detection rate is less than one. *Calcutta Statistical Association Bulletin.*

R Core Team (2018). *R: A Language and Environment for Statistical Computing.* R Foundation for Statistical Computing, Vienna, Austria.

Royle, J. A. (2015). Spatial capture-recapture with partial identity. *arXiv preprint arXiv:1503.06873.*