October 30, 2018

United States Disease Outbreak Simulation (USDOS) User Manual

PREREQUISITES TO USING THIS GUIDE

This user manual assumes that the user

1) Is familiar with USDOS and has determined that it is a good fit for their question
2) Has access to a cluster (high-performance) computing environment

If these do not apply, please consult these suggested resources:

1) To become familiar with USDOS and determine whether it is a good fit for the research question, please see the following references

   Buhnerkempe et al (2014) "*The Impact of Movements and Animal Density on Continental Scale Cattle Disease Outbreaks in the United States*." Plos One 9(3): 10.

   Foundational work building a cattle shipment model for the United States
   - Buhnerkempe et al (2013) "*A national-scale picture of US cattle movements obtained from Interstate Certificate of Veterinary Inspection data*." Preventive Veterinary Medicine 112(3-4): 318-329.
   - Gorsich et al (2018) "*Model-guided suggestions for targeted surveillance based on cattle shipments in the US.*" Preventive Veterinary Medicine 150: 52-59.
   - Lindstrom et al (2013) "*A Bayesian Approach for Modeling Cattle Movements in the United States: Scaling up a Partially Observed Network.*" Plos One 8(1): 11.
   - Portacci et al (2013) "*Assessment of paper interstate certificates of veterinary inspection used to support disease tracing in cattle*." Journal of the American Veterinary Medical Association 243(4): 555-560.

2) To gain access to a cluster computing environment, check with your institution's IT department

October 30, 2018

MODEL SUMMARY

-A one page executive summary of model

- Background
- How it works
- What it can do (Control measures)
- What it accounts for (constraints)

October 30, 2018

QUICK START GUIDE

Required setup for an initial run of USDOS (Note: If USDOS is already initialized on your computing system, please proceed to the run-specific steps below)

1. Download USDOS zip from repository
2. Unzip USDOS folder into your cloud computing working directory
3. Create object directory (mkdir obj)
4. Add any additional necessary files
5. Edit makefile and associated files

Run-specific steps

1. Use createConfig_Function.R to generate config, batch, and job files
2. Load required modules
3. Compile
4. Submit job
5. Export files
6. Generate results reports with post_processing.R

October 30, 2018

TABLE OF CONTENTS

Style conventions used in this guide

xx is used to signify a two-digit number between 01 and 10

```
Text in this color and font is programming code
```

** is used to signify that additional, general text (such as a filename) is present here. This is in contrast to the single * which is used in programming to specify general text.

Initial setup of USDOS in a high-performance computing environment

Note: These steps are only required once per model version

1. Download the USDOS master folder from the appropriate repository
2. Transfer the zipped directory onto the desired working directory of your high-performance computing system
3. Unzip the folder in the working directory
4. Create an object directory: mkdir obj
5. Add any required files
    a. inputfiles
        i. Toms's old beef and dairy
        ii. Landfills
    b. FLAPS
        i. Create a folder for the FLAPS: mkdir FLAPS
    c. Templates
    d. R config script
6. Complete any necessary edits to the makefile and related files

Steps requires for each model run

Each model run requires three types of files

- config files include information about run type and the associated parameters
- BATCH files are used in a high-performance, parallel computing environment to identify computing parameters, such as the run priority, memory allowed, maximum time, etc.
- A job file is used to run all the BATCH files, which in turn run the config files

Runs can allow several different types of control measures

- IP cull
- IP cull & DC cull
- IP cull & DC vax
- IP cull & 3km ring vaccination
- IP cull & 10km ring vaccination

To run the model

1. Edit the pre-processing R script to produce the desired runs using `nano createConfig_Function.R`

2. Run createConfig_Function.R to create the required config, batch, and job files
   a. Load R: `load module R/3.5.0`
   b. Run the R script: `Rscript createConfig_Function.R`

3. Make sure the job, batch, and config files are saved in the folder containing the USDOS run program. Note: This folder should also contain the /inputfiles, /include, and /FLAPS folders

4. This should generate, for each control type
   a. 100 files names config_**_00xx_ZZ_YYYYMMDD.txt
      i. xx: 01-10 for the 10 FLAPS
      ii. ZZ: Also 01-10, signifying the 10 runs of each FLAPS
      iii. YYYYMMDD: The date (October 24 2018 is written as 20181024)
   b. 100 files called "BATCH_baseflaps12_00XX_ZZ_YYYYMMDD.sh"
   c. 1 jobfile called **.job

5. Load required modules
   a. `module load gcc/6.1.0`
   b. `module load gsl/2.1`

6. Compile the run program with `make`
   a. Note: if you have made changes since the previous run, use `make clean` to remove the previous program, then `make` to re-generate it

7. Run the program
    a. `chmod a+x jobname.job`
    b. `./jobname.job`

8. Check that the run is working
    a. Running the ./jobname.job command above should result in a list of 100 (per run) lines reading "Submitted batch job [a number]"
    b. Check your working directory for .err and .out files (MI_**_fxx_xx.err/.out)
        i. If nothing is present, check your queue—your job may not have been allocated computing time yet.
        ii. If present, check these files using `nano [file name]`
            1. .err files should be blank
            2. .out files should list steps the process has completed
                a. Note: These steps take some time to complete, so what you see in the .out file will vary depending on how far along the process is.
    c. If the .err and .out files look good, wait. This takes a while.

9. When the run is completed, extract the files
    a. Input
        i. config
        ii. BATCH
        iii. .job
        iv. Runlog.txt
    b. output
        i. .err
        ii. .out
        iii. _detail.txt
        iv. _summary.txt

10. Process results/generate reports with post-processing code

Abbreviations

DC: Dangerous Contact

FLAPS: Farm Location and Animal Population Simulator

IP: Infected Premises

SFTP: Secure File Transfer Protocol

USAMM: United States Animal Movement Model

USDOS: United States Disease Outbreak Simulation

VAX: Vaccine/vaccinated/vaccination

Glossary

Cull: To depopulate the animals on a permises

Dangerous Contact (DC): Premises that have an epidemiological link to the infected premises.

Infected Premises (IP): Premises that are infected and have been detected and reported.

Movement ban: A prohibition on animal movements, here implemented at the state or county level.

Ring vaccination: Vaccination in a solid circle centered on the IP with a radius of 3 or 10Km. These ring sizes are commonly used in modeling studies and encompass a good range of possible ring sizes. The optimum ring size for controlling the UK 2001 outbreak was predicted to be around 10Km (Tildesley, et al. 2006).

File types and extensions

File Types

Input

config_**

BATCH_**

**.job

Output

**.err

**.out

**_detail.txt

**_summary.txt

File extensions

**.sh : batch file

**.job : job file

Troubleshooting

Pre-processing

“>50 warnings of “In readLines(template.config.location) : incomplete final line found on templates/USDOS_CONFIG_TEMPLATE_NoControls_workaround.txt'”

- Incomplete final line in template: open the file, hit return after last line, save and close.

Running the model

Can’t access BATCH files

- Check in R code that allFips is correctly capitalized (is not “allfips”)
- Make sure FLAPS names are right in R code (“12_min”)

Compile fail

- edit makefile carefully
- edit the county/state/shipment manger files

Errors when running the model

The model runs, but only produces .err and .out files

- Look into these files
- If the last line refers to a file, re-download it and try again (might have an end of line error)