Implementation

Fitting g across size classes

Inputs
- SE trial data, observation columns, predictor columns
- CP trial data, observation columns, predictor columns
- Search schedules
- Size class factor
- Number of iterations

Run SEmodsacrosssizes
Run ThetaSEcreateacrosssizes
Pass output to SE table function (AICtabcreateSEmods)
Pass output to SE graphics functions
Allow user to select from the model fits for each size class
- Store the choice of model for each size class

Run CPmodsacrosssizes
Run ThetaCPcreateacrosssizes
Pass output to CP table function (AICtabcreateCPmods)
Pass output to CP graphics functions
Allow user to select from the model fits for each size class
- Store the choice of model for each size class

Run gcreateacrosssizes
- Produces an array of g values with dimensionality (Niter, 1, Nss, Ncellcombos, Nsizeclasses)

Pass output to g table function
Pass output to g graphics functions

Estimating M across splits

Inputs
- Array from gcreateacrosssizes
- Proportion weighted area searched (PWAS) for each size class x turbine x search schedule combination
- Carcass observations, split column

Run Mhatgenerator
- Produces an array of $\widehat{M}$ values with dimensionality (Niter, Nss, Nunits, Nsplitcategories, Nsizeclasses)

Condense $\widehat{M}$ array to split category-level estimates using Mhatcondense
Pass output to $\widehat{M}$ table function (Mhattable), which allows for whole-facility expansion
Pass output to $\widehat{M}$ graphics function

Functions

SEmodsacrosssizes

Inputs: SE trial data, SE observation columns, SE predictors, size class column, if k should be fixed and value
Actions: Fit all possible models for each size class
Output: list (length Nsizeclasses) of lists (each length NmodelsSE) of model fits
To Do: modularize the internal model fitting to its own function

ThetaSEcreateacrosssizes

Inputs: SE trial data, SE predictors, size class column, list from SEmodsacrosssizes, Niterations, if k should be fixed and value
Actions: draw Niter samples from the models for each cell for each size class
Output: multidimensional array [Niterations, 2, NcellSE, NmodelsSE, Nsizeclasses]

AICtabcreateSEmods

Inputs: list from SEmodsacrosssizes, selection of what to sort on (NULL, "AIC", or "AICc")
Actions: create a model output table array (length = Nsizeclasses), each table has NmodelsSE rows, sort based on input
Output: list of sorted model tables for SE; also saves out as separate csv for each size class

SEgraphscreate

Inputs: SE model list, SE data, SE predictors, array of theta SEs, Niterations, observation columns, size class column
Actions: create parameter plots and SE decay curves for each size class, model, and cell combination, and compare to cell means
Outputs: plots of parameters and SE decay curves for each size class, each model, and cell combo, compared to cell means; saved out
To Do: modularize, clean up code

CPmodsacrosssizes

Inputs: CP trial data, CP predictors, size class column, unit of time to use
Actions: Fit all possible models for each size class
Output: list (length Nsizeclasses) of lists (each length NmodelsCP) of model fits
To Do: modularize the internal model fitting to its own function

ThetaCPcreateacrosssizes

Inputs: CP trial data, CP predictors, size class column, list from CPmodsacrosssizes, Niterations
Actions: draw Niter samples from each model for each cell for each size class
Output: multidimensional array [Niterations, 2, NcellCP, NmodelsCP, Nsizeclasses]

AICtabcreateCPmods

Inputs: list from CPmodsacrosssizes, selection of what to sort on (NULL, "AIC", or "AICc")
Actions: create a model output table array (length = Nsizeclasses), each table has NmodelsCP rows, sort based on input
Output: list of sorted model tables for CP; also saves out as separate csv for each size class

CPgraphscreate

Inputs: CP model list, CP data, CP predictors, array of theta CPs, Niterations, time unit choice, size class column
Actions: create a K-M survival curve and the model-fitted curve for each cell combo within each model within each size class
Outputs: survival curves for each size class, each model, and cell combo; saved out
To Do: modularize, clean up code

gcreateacrosssizes:

Inputs: CP trial data, SE trial data, Search Schedule data, CP predictors, SE predictors, array of CP thetas, array of SE thetas, list from CPmodsacrosssizes, CP models to use for each size, SE models to use for each size

Actions: calculate g (using gvec) for each search schedule x cell combination x size class

Output: multidimensional array of g values [Niter, 1, Nss, Ncellcombos, Nclasses]

gtablecreate

Inputs: g array, confidence interval width

Action: summarize the g iterations according to each size class, search schedule, and cell combo

Output: table of mean (with CI) gs, also saved out

ggraphscreate

Inputs: garray

Actions: create a distribution of simulated g values for each size class, search schedule, and cell combo

Output: distributions of gs for each size class, search schedule, and cell combo; saved out

Mhatgenerator

Inputs: carcass observations, PWAS for each size class x turbine x search schedule combo, size class column, split column, unit column, search schedule column, seed for random number generator, CP predictors, SE predictors, CP trial data, SE trial data, g array from gcreateacrosssizes

Action: calculate $\widehat{M}$ values

-For each size class r (*NOTE: r replaced q due to q() being a function in* R)

-For each split category l

-For each turbine k

-For each Search Schedule j

-For each cell i

-subset the inputs to X_{ijklr} (length 1), a_{jkr} (length 1), g_{ijr} (length Niter)

-Draw $\tilde{X}_{ijklr} \sim Bin\left(\frac{X_{ijklr}}{g_{ijr}}, g_{ijr}\right)$

-Calculate $\widetilde{M}_{ijklr} = round(\frac{\tilde{X}_{ijklr}}{g_{ijr}})$

-Sum $\widetilde{M}$ across all cells within the search schedule

-Calculate $\widehat{M}_{jklr} = \frac{\sum_{i=1}^{Ncells} \widetilde{M}_{ijklr}}{a_{jkr}}$

Output: multidimensional array of $\widehat{M}$ [Niter, Nss, Nunits, Nsplitcategories, Nsizeclasses]

Mhatcondense

Inputs: Mhataray

Action: condense across search schedules, units, and size classes

Output: matrix of Mhat values: rows = Niterations, columns = Nsplitcategories

Mhattable

Inputs: condensed (to split categories) Mhat, fraction of facility searched, confidence interval width

Action: summarize the Mhat iterations according to the split confidence intervals, for searched and whole

Output: table of mean (with CI) mortalities for the searched area and the whole facility, also saved out

Mhatgraph

Inputs: condensed (to split categories) Mhat, fraction of facility searched

Action: create a distribution of simulated searched-area and whole-facility Mhats for each split category

Output: distributions of Mhat for each split category, both area searched and whole-facility; saved out

factorcombinations

Inputs: predictor variables, dataset
Action: creates a factor combination table for a CP or SE analysis
Output: factor combination table
to-do: generalize (base on make_egDat function)

crossmodelcells

Inputs: CP predictors, SE predictors, CP trial data, SE trial data
Action: creates a factor combination table across the CP and SE analyses
Output: factor table applicable to both analyses
to-do: generalize (base on make_egDat function)

pkfunction

Inputs: number of trials with no detection for each carcass, trial on which each carcass was found, parameters, number of parameters associated with p, groups, maximum number of misses for a carcass, combined p-k model matrix, if k should be fixed and value
Action: function is optimized over using optim()
Output: negative log likelihood of the observations given the parameters

gvec

Inputs: specific Search schedule, specific CP theta [Niterations, 2], specific CP distribution, specific SE theta [Niterations, 2]
Action: calculate g for a specific set of CP and SE parameters and a specific search schedule
Output: simulation of g values [Niterations, 1]

ppersist

Inputs: specific CP distribution, CP parameters, arrival times, search intervals
Action: calculates the probability that a carcass that arrives in the interval between t_arrive0 and t_arrive1 persists until t_search, using exact integrals
Output: probability of persistence to detection for each interval

logit

Inputs: single numeric value
Action: computes the logit
Output: logit of the single numeric value

alogit

Inputs: single numeric value
Action: computes anti-logit
Output: anti-logit of the single numeric value