Meier et al. 2022 - Supporting Information File 2: Example linking NEON TOS Coarse Downed Wood volume and Small Mammal abundance data at plot and site scales

Courtney L. Meier, Katherine M. Thibault, and David T. Barnett

March 17, 2022

Contents

Introduction

The NEON Terrestrial Observation System (TOS) data products are frequently spatially integrated at multiple scales, from the site level to the level of relatively small 'sampling cells' that occur within plots (see Supporting Information File 1, Table 3). In addition, the frequency with which TOS data products are generated at each site varies widely, from multiple bouts per year to once every 5 years. This worked example is a companion piece to our manuscript, *"Spatial and temporal sampling strategy connecting NEON Terrestrial Observation System protocols,"* and the purpose is to illustrate how a user of NEON data can draw scientific insight by working with two TOS data products collected at different spatial scales and at different temporal frequencies.

Here, we focus on the 'Coarse downed wood log survey' data product (DP1.10010.001; generated every 5-years) and the 'Small mammal box trapping' data product (DP1.10072.001; generated multiple times per year). Using R software (https://www.r-project.org/) to download data from the NEON API via the `neonUtilities` package (https://cran.r-project.org/web/packages/neonUtilities/), we show how to link data at both the plot and site spatial scales via the `plotID`. We also develop one method for comparing TOS data across products generated with very different temporal frequencies. For this example, we chose to investigate correlations between 'Coarse downed wood log survey' data and 'Small mammal box trapping' data based on the hypothesis that coarse downed wood (CDW) provides habitat for small mammals in forested systems, and increasing CDW volume may therefore be correlated with increasing small mammal abundance, both within and among sites.

Load data from the NEON Data Portal

Note: When retrieving data from the NEON API using the `neonUtilities` functions (see code chunks below), use of the `token` argument allows authentication with a user-specific API token. Use of a token is

not required, but tokens provide faster download speeds and help NEON to measure data use for reporting to the NSF.

Load Coarse Downed Wood Tally Data

At sites with sufficient logs >= 2 cm diameter, the 'Coarse downed wood log survey' data product, hereafter referred to as 'CDW Tally', is generated from both Distributed and Tower plots on a 5 year interval. However, sampling of Distributed and Tower plots is staggered through time such that each plot type is sampled every 2-3 years within a site. Distributed plots are allocated across each site in proportion to the area of dominant National Land Cover Database (NLCD) Vegetation Classes, and these plots allow creating statistically robust site-level estimates of both CDW volume (derived from tallies) and small mammal density. The Small Mammal (MAM) data product is generated annually from most TOS sites, and MAM data are collected from grids that are typically colocated with Distributed base plots, but not Tower plots, which also support CDW sampling. Thus, the first tasks for this example are to identify which NEON sites have produced CDW Tally data, identify several forested sites with recent Distributed plot data for the example, and then identify which Distributed plotIDs are colocated with MAM sampling.

The NEON 'Coarse downed wood log survey' data product citation is:

- NEON (National Ecological Observatory Network). Coarse downed wood log survey, RELEASE-2022 (DP1.10010.001). https://doi.org/10.48443/54dd-9407. Dataset accessed from https://data.neonscience.org on March 17, 2022

```
# ### Retrieve CDW Tally data for last 6 years from NEON Data Portal for all sites and dates
#
# cdwDP <- neonUtilities::loadByProduct(
#   dpID = "DP1.10010.001",
#   site = "all",
#   startdate = "2016-01",
#   enddate = "2021-12",
#   package = "basic",
#   release = "RELEASE-2022",
#   tabl = "all",
#   check.size = FALSE,
#   token = Sys.getenv('NEON_PAT')
# )
#
# #   Extract the field tally data from the download list, and save for quicker read-in
# cdw <- cdwDP$cdw_fieldtally
#
# # saveRDS(cdw, file = "cdw_fieldTally_2016-2021.RDS")
# #
# # #   Read in data saved via neonUtilities code chunk above
# # cdw <- readRDS(file = "cdw_fieldTally_2016-2021.RDS")
#
# #   Summarise data to identify 5-6 sites with recent CDW Tally data from Distributed plots that
# #   can be paired with MAM data; for demonstration purposes, goal is to choose a small number of
# #   forested sites that span a range of habitats.
# summaryDist <- cdw %>%
#   dplyr::filter(plotType=="distributed") %>%
#   dplyr::group_by(domainID, siteID, eventID, samplingImpractical) %>%
#   dplyr::summarise(
#     plotCount = length(unique(plotID))
#   )
#
```

```
# #   From 'summaryDist' output, select 6 sites with closed-canopy forest
# theSites <- c("HARV","JERC","STEI","UKFS","RMNP","ABBY")
#
# #   Filter CDW data to selected sites, filter to most recent eventID, remove unneeded columns
# tempCDW <- cdw %>%
#   dplyr::filter(
#     plotType=="distributed",
#     siteID %in% theSites
#   ) %>%
#   dplyr::group_by(domainID, siteID) %>%
#   dplyr::filter(
#     yearBoutBegan==max(yearBoutBegan)
#   ) %>%
#   dplyr::arrange(domainID, siteID, plotID) %>%
#   dplyr::ungroup() %>%
#   dplyr::select(-uid, -coordinateUncertainty, -elevationUncertainty, -publicationDate)
#
#
#
# ### Identify plotIDs in CDW dataset that are colocated with MAM sampling grids
# #   First, create MAM 'namedLocation' value for each CDW namedLocation value (e.g., 'HARV_001.mammalG
# #   the goal is to check whether each CDW namedLocation has a colocated MAM namedLocation. The struct
# #   of the namedLocation value for any NEON data product can be determined by downloading data and
# #   checking the namedLocation field.
#
# tempCDW <- tempCDW %>%
#   dplyr::mutate(
#     mamNamedLoc = stringr::str_replace(
#       string = namedLocation,
#       pattern = "basePlot.cdw",
#       replacement = "mammalGrid.mam"
#     ),
#     .after = namedLocation
#   ) %>%
#   as.data.frame()
#
# #   Second, retrieve namedLocation data for each mamNamedLoc value that was constructed above; if a C
# #   namedLocation is also a MAM namedLocation, the geoNEON::getLocByName() function will return a val
# #   the NEON API; if a CDW namedLocation is not also a MAM namedLocation, the function does not retur
# #   value. Note: The 'data' argument must be a data.frame() and CANNOT be a tibble (the latter is com
# #   returned by dplyr).
#
# mamPlots <- geoNEON::getLocByName(
#   data = tempCDW,
#   locCol = "mamNamedLoc",
#   locOnly = TRUE,
#   token = Sys.getenv('NEON_PAT')
# )
#
#
# ##  Save data locally for quicker subsequent read-in
# # saveRDS(mamPlots, file = "cdw_mam_collocation.RDS")
# # mamPlots <- readRDS(file = "cdw_mam_collocation.RDS")
```

```
#
# #   Third, join tempCDW data with mamPlots and create a 'subtype' column to identify colocated samplin
# tempCDW <- tempCDW %>%
#   dplyr::left_join(
#     mamPlots %>%
#       select(plotID, subtype),
#     by = "plotID"
#   ) %>%
#   dplyr::select(-mamNamedLoc)
```

As an aside, rather than programmatically retrieving TOS Spatial Data via the API with the `geoNEON` functions, as is done in the code chunk above, these spatial data may also be manually downloaded:

- First, navigate to https://data.neonscience.org/documents
- Click on 'Spatial Data' and click on the 'All_NEON_TOS_Plots_VX' link to retrieve a zip that contains a .csv with plot data.
- If this approached is used, one can then read in the included .csv file.
  - For example: `tosPlots <- read.csv(file = "All_NEON_TOS_Plot_Centroids_V8.csv", header = TRUE)`

Load Small Mammal Data

Small Mammal sampling grids (n=3 to n=8 per site) are colocated with Distributed base plots that support multiple TOS protocols (including Coarse Downed Wood), and grids are sampled one to several times per year (ideally, 4-6 times per year). Similar to Distributed base plots, mammal grids are allocated using a stratified random design informed by NLCD Vegetation Class.

The NEON 'Small mammal box trapping' data product citation is:

- NEON (National Ecological Observatory Network). Small mammal box trapping, RELEASE-2022 (DP1.10072.001). https://doi.org/10.48443/h3dk-3a71. Dataset accessed from https://data.neonscience.org on March 17, 2022

```
# ### Retrieve small mammal box trapping data for same sites and date range as retrieved for CDW
# mamDP <- neonUtilities::loadByProduct(
#   dpID = "DP1.10072.001",
#   site = theSites, #--> same sites identified for CDW
#   startdate = "2016-01",
#   enddate = "2021-12",
#   package = "basic",
#   release = "RELEASE-2022",
#   tabl = "all",
#   check.size = FALSE,
#   token = Sys.getenv('NEON_PAT')
# )
#
# # #   Save to RDS format for quicker read-in
# # saveRDS(mamDP, file = "mam_allTables_2016-2021.RDS")
# #
# # #   Read in stored MAM data
# # mamDP <- readRDS(file = "mam_allTables_2016-2021.RDS")
#
# #   Convert list tables to dataframes in the Global environment
# list2env(mamDP, envir=.GlobalEnv)
#
```

```
#
# ### Read in master SMALL_MAMMAL taxon table via the NEON API
# #   https://www.neonscience.org/resources/learning-hub/tutorials/neon-api-usage
# #   using the verbose option to get the taxonProtocolCategory field
# mam.req <- httr::GET(
#   "https://data.neonscience.org/api/v0/taxonomy/?taxonTypeCode=SMALL_MAMMAL&verbose=true&offset=0&lim
# )
# mam.list <- jsonlite::fromJSON(httr::content(mam.req, as="text"))
#
```

Data QC Checks

Coarse Downed Wood Tally QC Checks

Note: The word `'log'` is used frequently in the Coarse Downed Wood dataset, and refers to a downed piece of wood, not a logarithm.

For CDW tally data, prior to analysis it is important to:

- Check for duplicates, and
- Remove any tallied logs with a `logDistance` greater than the protocol-specified transect length, and
- Verify that all tallied logs meet 'limiting distance' tally criteria based on reported `equivalentLogDiameter`, `logDistance`, and `volumeFactor` data (see Affleck 2010 for details).
  - The `equivalentLogDiameter` is used so that round and elliptical logs (typically highly decayed) can be assessed with the same criteria. Logs with `logDistance` greater than the limiting distance are removed from the dataset.
  - Because technicians sometimes use look-up tables to determine limiting distance from a subset of `equivalentLogDiameter` values (e.g., when digital ingest devices fail), and look-up tables require choosing the nearest `equivalentLogDiameter` rather than using the actual measured value, it is expected that some logs will have `logDistance` greater than the calculated limiting distance and will be removed.

```
# ### Check for individualID duplicates; duplicate checks are important because log counts are used
# ### to estimate CDW volume
# #   First, construct an individualID for logs < 10 cm diameter that are not tagged; smaller logs are
# #   tagged and are given a unique temporary logID beginning with 'L' since they are less likely to pe
# #   over the 5-year CDW Tally measurement interval; second, construct a primary key from plotID, date
# #   lidsAzimuth, and individualID.
#
# tempCDW <- tempCDW %>%
#   dplyr::mutate(
#     individualID = case_when(
#       is.na(individualID) & targetTaxaPresent=="Y" ~ paste("NEON.CDW", domainID, plotID, logID, sep =
#       TRUE ~ individualID
#     ),
#     key = paste(plotID, date, lidsAzimuth, individualID, sep = "_")
#   )
#
# #   Identify duplicates ('dups') based on 'key' value
# cdwDupsKey <- tempCDW %>%
#   dplyr::filter(duplicated(tempCDW$key)) %>%
#   dplyr::select(key)
#
# cdwDups <- tempCDW %>%
```

```
#   dplyr::filter(key %in% cdwDupsKey$key)
# # Log data assessment for dups: Only first one listed appears to be a real duplicate based on
# # examination of other data fields; ABBY records are mis-indentified by the function as 'dups'
# # because transect was reflected (i.e., not real duplicates).
#
# #   Remove second instance of cdwDups$key=="STEI_002_2016-05-26_320_NEON.CDW.D05.00375"
# cdwRemoveDup <- cdwDups %>%
#   dplyr::distinct(key)
#
# tempCDW <- tempCDW %>%
#   dplyr::filter(!key %in% cdwRemoveDup$key[1]) %>%
#   dplyr::bind_rows(cdwDups[1,]) %>%
#   dplyr::arrange(domainID, siteID, plotID, lidsAzimuth, individualID)
#
#
# ### Perform QC checks for volumeFactor, logDistance, and equivalentLogDiameter
# ##  Volume Factor: Check value matches protocol and flag if no match
# #   Read in CDW volumeFactor and transectLength look-up table derived from protocol (NEON.DOC.001711)
# #   Note that RMNP has different F-values and transect lengths for Distributed and Tower plots due
# #   to distant parcels of land used for each plot type and different forest types. The RMNP values in
# #   this table are specific to Distributed plots since we wish to compare data with Small Mammal data
# cdwParam <- read.csv(file = "cdw_siteParamLookup.csv", header = TRUE)
#
# #   Join look-up table values with CDW data and flag records with incorrect volume factor (i.e., f-va
# tempCDW <- tempCDW %>%
#   dplyr::left_join(
#     cdwParam %>% dplyr::select(siteID, fValue, transectLength),
#     by = "siteID"
#   ) %>%
#   dplyr::mutate(
#     fValQF = case_when(
#       is.na(volumeFactor) ~ 1,
#       volumeFactor==fValue ~ 0,
#       TRUE ~ 1
#     )
#   )
#
# #   Check records with volumeFactor problems and evaluate
# fCheck <- tempCDW %>%
#   dplyr::filter(fValQF==1) %>%
#   dplyr::select(domainID, siteID, volumeFactor, fValue, fValQF)
# #   volumeFactor problems identified for HARV and STEI; volumeFactor is supposed to be assigned
# #   at the site level and should not vary from plot-to-plot. The variation in the data likely
# #   reflects a data entry error. We encourage data users to let NEON staff know about issues
# #   they discover like this one: https://www.neonscience.org/about/contact-us
#
# #   Correct volumeFactor data for HARV, STEI sites based on fCheck results
# tempCDW <- tempCDW %>%
#   dplyr::mutate(volumeFactor = as.numeric(volumeFactor)) %>%
#   dplyr::mutate(
#     volumeFactor = case_when(
#       siteID=="HARV" & (volumeFactor!=5 | is.na(volumeFactor)) ~ 5,
#       siteID=="STEI" & (volumeFactor!=5 | is.na(volumeFactor)) ~ 5,
```

```
#       TRUE ~ volumeFactor
#     )
#   )
#
#
# ### Tally QC checks: Verify that logs in the dataset should be tallied based on protocol criteria
# ##  Log Distance: Check for values that exceed maximum transect length as defined in protocol, then f
# tempCDW <- tempCDW %>%
#   dplyr::mutate(
#     logDistQF = case_when(
#       is.na(logDistance) ~ 0,
#       logDistance > transectLength ~ 1,
#       TRUE ~ 0
#     )
#   )
#
#
# ##  Check whether logDistance is <= limitingDistance, as expected, based on equivalentLogDiameter, an
# #   Use formula to calculate limiting distance from Affleck 2010, Appendix A:
# #   limitingDist = ((pi^2)*(roundDiameter^2))/(8*transectNum*fValue), where
# #   roundDiameter = round diameter of the log (cm), taken from the equivalentLogDiameter field in NEO
# #   transectNum = integer value for the number of transects established from a survey point (3 for NE
# #   fValue = the F value assigned at the site level, taken from the volumeFactor field in NEON data
# tempCDW <- tempCDW %>%
#   dplyr::mutate(
#     limitingDist = round(((pi^2)*(equivalentLogDiameter^2))/(8*3*volumeFactor), digits=2)
#   ) %>%
#   dplyr::mutate(
#     limDistQF = case_when(
#       is.na(limitingDist) ~ 0,
#       logDistance > limitingDist ~ 1,
#       TRUE ~ 0
#     )
#   )
#
#
# ##  Filter out flagged records corresponding to logs that should not have been tallied
# filterCDW <- tempCDW %>%
#   dplyr::filter(
#     limDistQF==0,
#     logDistQF==0
#   )
```

Small Mammal Box Trapping Data QC: Check for duplicates

- `mam_perplotnight`: In this table, duplicate records are defined as those with the same plot and date combination (as captured in an auto-generated nightuid).
- `mam_pertrapnight`: In this table, duplicate records are defined as those with the same nightuid, trap coordinate, and tagID or individualCode; note that the standard `neonOS::removeDups()` function used to remove duplicates cannot account for multiple captures of untagged individuals in a single trap.

```
# ### First, check mam_perplotnight table by nightuid using standard removeDups function
# mam_plotNight_nodups <- neonOS::removeDups(
#   data = mam_perplotnight,
#   variables = variables_10072,
#   table = "mam_perplotnight"
# )
# # Running this check generated the message: Primary key fields contain NA values and/or empty strings
# # Results may be unreliable; check input data carefully. - followed by an error message
#
#
# ##  To troubleshoot, first query the table to find the problematic records
# problems <- which(is.na(mam_perplotnight$nightuid))
#
# #   Next, check the mam_pertrapnight table to see if there are corresponding records that might
# #   have a nightuid
# temp <- mam_pertrapnight %>%
#   dplyr::filter(
#     plotID %in% mam_perplotnight$plotID[problems] & collectDate %in% mam_perplotnight$collectDate[pro
#   )
# # Since the remaining fields were populated and there are captures for each, these records appear to
# # represent a valid night of trapping. So, we will assign custom temporary nightuids in both tables h
# # As an aside, we encourage users to let NEON staff know about issues they discover like this one:
# # (https://www.neonscience.org/about/contact-us).
#
# #   Create custom nightuids for duplicates identified above that are missing primary key value
# mam_perplotnight_adj <- mam_perplotnight %>%
#   dplyr::mutate(nightuid = paste(plotID, '_', collectDate, sep = ''))
#
# mam_pertrapnight_adj <- mam_pertrapnight %>%
#   dplyr::mutate(nightuid = paste(plotID, '_', collectDate, sep = ''))
#
# #   Re-run the duplicate removal function
# mam_plotNight_nodups <- neonOS::removeDups(
#   data = mam_perplotnight_adj,
#   variables = variables_10072,
#   table = "mam_perplotnight"
# )
# # No duplicate key values found
#
#
# ### Second, check the mam_pertrapnight table by nightuid and trapcoordinate using the standard
# ### neonOS::removeDups() function.
#
# #   Note that RELEASE 2022 contains ~142K records for theSites downloaded, so this operation can take
# #   time. In rare cases, multiple animals can be captured in a single trap on the same night and not
# #   receive unique tagIDs or individualCodes. This means that the duplicate function is not effective
# #   these records; here, we remove these records prior to running the standard check, and we then add
# #   records back to the data frame afterwards.
#
# mam_trapNight_multipleCaps <- mam_pertrapnight_adj %>%
#   dplyr::filter(
#     trapStatus == "4 - more than 1 capture in one trap" & is.na(tagID) & is.na(individualCode)
#   )
```

```
#
# mam_trapNight_remainingRecords <- mam_pertrapnight_adj %>%
#   dplyr::filter(!(uid %in% mam_trapNight_multipleCaps$uid))
#
# mam_trapNight_nodups <- neonOS::removeDups(
#   data = mam_trapNight_remainingRecords,
#   variables = variables_10072,
#   table = "mam_pertrapnight"
# )
# #   Output: Two unresolveable duplicates flagged with duplicateRecordQF=2, which indicates records th
# #   identical based on primary keys but contain different values in other fields; see ?neonOS::remove
# #   documentation for additional information on duplicate record flagging.
#
#
# ##  Follow-up trouble-shooting to deal with unresolveable duplicates
# #   Check for unexpected NAs in primary keys
#
# which(is.na(mam_pertrapnight_adj$nightuid)) #--> yields 0
# which(is.na(mam_pertrapnight_adj$trapCoordinate)) #--> yields 0
# # Two lines above mean that trapID and individualCode are the source of the NAs, which is expected
# # based on the protocol.
#
# #   Next, check that the unresolved duplicates will not impact the intended analyses
# dupCheck <- mam_trapNight_nodups %>%
#   dplyr::filter(
#     duplicateRecordQF == 2 & trapStatus != "4 - more than 1 capture in one trap"
#   )
# # No records in the set, so these unresolved duplicates will not impact the intended analyses
#
#
# ##  Add multiple capture records back to dataset
# mam_trapNight_nodups <- mam_trapNight_nodups %>%
#   dplyr::mutate(collectDate = as.Date(collectDate)) %>%
#   dplyr::bind_rows(mam_trapNight_multipleCaps)
```

Analysis of Spatially Integrated Data Products at Plot and Site Scales

In this section, we generate Coarse Downed Wood volume estimates and Small Mammal abundance estimates at both plot and site scales in order to demonstrate the spatial integration of NEON TOS data products. Because CDW volume is generated every 5-years and does not change rapidly, barring fires or other site management activities, we compare with Small Mammal data collected the same year as the CDW data +/- 1 year. For example, for HARV CDW volume data collected in 2019, we compare to Small Mammal data collected from 2018-2020.

Note: NEON publishes site management data via the 'Site management and event reporting' product (DP1.10111.001; https://data.neonscience.org/data-products/DP1.10111.001).

Create Coarse Downed Wood Volume Datasets

Using the LIDS method (Affleck 2008, 2010), CDW volume is calculated at the plot scale as:

`CDW volume density = F * n` (m^3/hectare);

where `F` is the site-specific volume factor, and `n` is the number of qualifying logs tallied across all three transects originating from the centroid of the plot. Importantly, at the plot scale, CDW volume density is colocated with Small Mammal abundance data at a maximum of n=6 plots per site (for theSites included).

At the site scale, mean CDW volume (m^3/hectare) is simply the mean of the plot scale values; the mean site-scale parameter is robust because of the spatially-balanced, stratified-random nature of the TOS Spatial Design. When comparing CDW volume and Small Mammal abundance at the site scale, CDW data from all Distributed plots are used for the calculation.

```
# ### Calculate CDW volume for each plotID in the cleaned dataset
# #   First, focus on plots with targetTaxaPresent=="Y" --> i.e., at least one qualifying log was talli
# #   within the plot
# plotCdwVol <- filterCDW %>%
#   dplyr::filter(is.na(samplingImpractical) | samplingImpractical=="OK") %>% # Remove SamplingImpracti
#   dplyr::filter(targetTaxaPresent=="Y") %>% # Remove records for transects with no logs
#   dplyr::group_by(domainID, siteID, volumeFactor, eventID, plotID) %>%
#   dplyr::summarise(
#     logCount = n()
#   ) %>%
#   dplyr::mutate(cdwVol = volumeFactor * logCount)
#
# #   Second, identify plots where all three transects do not have logs and bind to log count output fr
# noLogs <- filterCDW %>%
#   dplyr::filter(targetTaxaPresent=="N") %>%
#   dplyr::group_by(domainID, siteID, volumeFactor, eventID, plotID) %>%
#   dplyr::summarise(
#     transectCount = n() # Identify plots with all 3 transects with ttP=="N"
#   ) %>%
#   dplyr::filter(transectCount==3) %>%
#   dplyr::mutate(
#     logCount = 0,
#     cdwVol = 0
#   ) %>%
#   dplyr::select(-transectCount)
#
# plotCdwVol <- plotCdwVol %>%
#   dplyr::bind_rows(noLogs) %>%
#   dplyr::arrange(domainID, siteID, eventID, plotID)
#
# #   Finally, add back mammalGrid collocation data
# temp <- tempCDW %>%
#   dplyr::select(plotID, subtype) %>%
#   dplyr::distinct()
#
# plotCdwVol <- plotCdwVol %>%
#   dplyr::left_join(temp, by = "plotID")
#
#
# ##  Display table of plot-level CDW volume estimates
# DT::datatable(
#   data = plotCdwVol %>% dplyr::rename(
#     "Log Count" = logCount,
#     "CDW Vol (m3/ha)" = cdwVol,
#     "Plot subtype" = subtype
#   ),
```

```
#   filter = "none",
#   rownames = FALSE,
#   options = list(pageLength = 10),
#   caption = "CDW volume density estimates by plotID for selected sites and eventIDs. The 'Plot subtyp
#   column indicates whether a given CDW plot is colocated with a Small Mammal sampling grid."
# )
#
#
# ### Calculate CDW Volume for each siteID in the cleaned dataset and display
# #   Calculate site-level mean CDW volume
# siteCdwVol <- plotCdwVol %>%
#   dplyr::group_by(domainID, siteID, eventID) %>%
#   dplyr::summarise(
#     meanCdwVol = round(mean(cdwVol, na.rm = TRUE), digits = 1),
#     sdCdwVol = round(sd(cdwVol, na.rm = TRUE), digits = 1),
#     nCdwVol = n()
#   )
#
# #   Display table of site-level CDW volume estimates
# DT::datatable(
#   data = siteCdwVol %>% dplyr::rename(
#     "Mean CDW Vol (m3/ha)" = meanCdwVol,
#     "StdDev CDW Vol" = sdCdwVol,
#     "Plot Number" = nCdwVol
#   ),
#   filter = "none",
#   rownames = FALSE,
#   options = list(pageLength = 10),
#   caption = "CDW volume density estimates by siteID for selected sites and eventIDs."
# )
#
```

Create Small Mammal Density datasets

Here, we employ the minimum number known alive (MNKA) approach to estimate total small mammal abundance - e.g., as defined in Slade & Blair (2000). This approach assumes that a marked individual is present at all sampling points between its first and last capture dates, even if it was not actually captured in those interim trapping sessions. To make the dataset meet this assumption, we first add those implicit records to the dataset to make them explicit.

Note that it is necessary to create Small Mammal `eventIDs` for legacy data collected prior to 2021. For the Small Mammal product, `eventIDs` correspond to discrete sampling bouts that represent a group of consecutive nights of trapping scheduled around a particular new moon.

```
# ### Create eventID grouping variable using mam_plotNight data and add to mam_trapNight data
# #   Assign the same eventID to each record with an endDate no more than 10 days later than the previo
# #   created record.
# mam_plotNight_nodups <- mam_plotNight_nodups %>%
#   dplyr::mutate(
#     year = as.integer(format(collectDate, "%Y")),
#     .before = eventID
#   ) %>%
#   dplyr::group_by(siteID, year) %>%
#   dplyr::mutate(
```

```
#     priorEndDate = data.table::shift(endDate, fill = endDate[1]),
#     diffDays = difftime(endDate, priorEndDate, units = "days"),
#     boutIncrement = ifelse(diffDays >= 10, 1, 0),
#     boutNum = cumsum(boutIncrement),
#     .after = endDate
#   ) %>%
#   dplyr::group_by(siteID, year, boutNum) %>%
#   dplyr::mutate(
#     weekBoutBegan = lubridate::isoweek(min(endDate)),
#     eventID = case_when(
#       is.na(eventID) ~ paste(siteID, year, weekBoutBegan, sep = "."),
#       TRUE ~ eventID
#     ),
#     .after = boutNum
#   ) %>%
#   dplyr::ungroup() %>%
#   dplyr::select(-priorEndDate, -diffDays, -boutIncrement, -boutNum)
# # Above, shift() defaults to type = "lag", which returns the previous row value for the specified col
#
# #   Add nightuid and eventID from mam_plotNight_nodups to mam_trapNight_nodups
# mam_trapNight_nodups <- mam_trapNight_nodups %>%
#   dplyr::left_join(
#     mam_plotNight_nodups %>%
#       dplyr::select(nightuid, eventID),
#     by = "nightuid"
#   )
#
#
# ### Create and finalize Small Mammal capture dataset for selected sites
#
# ##  First, subset trapping data to only the capture records, i.e., the records that describe a captur
# ##  mammal, including only those taxa for which the trapping protocol is designed. Subsetting can be
# ##  a simple filtering based on 'trapStatus' and 'taxonProtocolCategory'; however, we first check to
# ##  all captures have the correct 'trapStatus' - i.e., check if tagIDs exist but trap status does not
# ##  "capture".
#
# problemRecords <- mam_trapNight_nodups %>%
#   dplyr::filter(
#     !is.na(tagID),
#     !grepl("capture",trapStatus)
#   )
#
# #   If nrow(problemRecords) > 0, update the corresponding trapStatus fields to "5 - capture" or, in t
# #   case where there are multiple captures in one trap, "4 - more than 1 capture in one trap" - this
# #   case does not occur in the current dataset, so it is not addressed here.
# mam_trapNight_nodups <- mam_trapNight_nodups %>%
#   dplyr::mutate(trapStatus = case_when(
#     uid %in% problemRecords$uid ~ "5 - capture",
#     TRUE ~ trapStatus
#   ))
#
#
# ##  Second, create list of target taxa from taxon list
```

```
# targetTaxa <- mam.list$data %>%
#   dplyr::filter(taxonProtocolCategory == "target") %>%
#   dplyr::select(taxonID)
#
#
# ##  Third, simplify dataset to targetTaxa and core fields needed for analysis
# captures <- mam_trapNight_nodups %>%
#   dplyr::filter(grepl("capture", trapStatus) & taxonID %in% targetTaxa$taxonID) %>%
#   dplyr::select(uid, nightuid, plotID, collectDate, tagID)
#
#
# ##  Fourth, following the 'Minimum Number Known Alive' (MNKA) approach (Slade & Blair, 2000).
# #   Generate a column of all of the unique tagIDs included in the dataset
# uTags <- captures %>%
#   dplyr::select(tagID) %>%
#   dplyr::filter(!is.na(tagID)) %>%
#   dplyr::distinct()
#
# #   Create empty data frame to populate with 'for()' loop output
# capsNew <- dplyr::slice(captures, 0)
#
# #   For each tagged individual, add a record for each night of trapping done in the plots in which it
# #   captured between the first and last dates of capture.
#
# for (i in uTags$tagID) {
#   temp <- captures %>% filter(tagID == i)
#   firstCap <- as.Date(min(temp$collectDate), "YYYY-MM-DD", tz = "UTC")
#   lastCap <- as.Date(max(temp$collectDate), "YYYY-MM-DD", tz = "UTC")
#   possibleDates <- seq(as.Date(firstCap), as.Date(lastCap), by="days")
#   plots <- unique(temp$plotID)
#   potentialNights <- mam_plotNight_nodups %>%
#     dplyr::filter(as.character(collectDate) %in% as.character(possibleDates) &
#              plotID %in% plots) %>%
#     dplyr::select(nightuid,plotID, collectDate) %>%
#     dplyr::mutate(tagID = i)
#   temp2 <- dplyr::left_join(potentialNights, temp)
#   capsNew <- dplyr::bind_rows(capsNew, temp2)
# }
#
# # #   To avoid time-consuming processing, save a copy
# # saveRDS(capsNew, file = "mam_final_dataset.RDS")
# #
# # #   Read in processed capsNew dataset
# # capsNew <- readRDS(file = "mam_final_dataset.RDS")
#
# #   Add untagged individuals back to the dataset
# capsNew <- captures %>%
#   dplyr::filter(is.na(tagID)) %>%
#   dplyr::bind_rows(capsNew)
#
# #   Add eventID to enable summarizing by trapping bouts (includes up to 3 nights of trapping for some
# #   in each bout, 1 night for other plots).
# capsNew <- capsNew %>%
```

```
#   dplyr::left_join(
#     mam_plotNight_nodups %>%
#       dplyr::select(eventID, nightuid),
#     by = "nightuid"
#   ) %>%
#   dplyr::relocate(eventID, .after = collectDate) %>%
#   dplyr::arrange(eventID, plotID, collectDate)
```

Next, we define a function to calculate MNKA for each plot within each bout. The function requires the `capsNew` data set generated above as an input, as well as a list of plots of interest (`plotsOI`), in the event that the `capsNew` data contain records from more plots than are needed for the MNKA estimation.

```
# ### Define function to calculate MNKA for each plot within bout; 'plotsOI' is a required input vector
# ### plotIDs for which MNKA per plot per bout is desired.
#
# mnka_per_plot_per_bout <- function(capture_data, plotsOI) {
#   caps <- capture_data %>%
#     dplyr::filter(plotID %in% plotsOI)
#
#   ids_by_plot_bout <- capture_data %>%
#     dplyr::group_by(eventID, plotID) %>%
#     dplyr::distinct(tagID)
#
#   mnka_by_plot_bout <- ids_by_plot_bout %>%
#     dplyr::group_by(eventID, plotID) %>%
#     dplyr::count() %>%
#     dplyr::mutate(
#       siteID = stringr::str_extract(eventID, "^[A-Z]{4}"),
#       year = as.numeric(stringr::str_extract(eventID, "20[0-9]{2}")),
#       .before = eventID
#     ) %>%
#     dplyr::ungroup() %>%
#     as.data.frame()
#
#   return(mnka_by_plot_bout)
# }
# # Output is MNKA (count/hectare), based on 1 ha size of trapping grid
```

We now use the MNKA small mammal density function to calculate MNKA at the plot scale and site scale in order to compare to CDW volume density data calculated above. Because CDW volume does not change rapidly, and small mammal abundance may change substantially both within and among years, here we generate the maximum small mammal abundance for a given plot for a three-year window centered on the year that logs were tallied for CDW.

```
# ### Calculate MNKA per plot per bout for all data in processed capsNew dataset and select 3-year wind
# ### data appropriate for each site in order to match up with CDW volume data.
#
# plotBoutMNKA <- mnka_per_plot_per_bout(
#   capture_data = capsNew,
#   plotsOI = unique(capsNew$plotID)
# )
#
# #   Create table of MAM data within 3-year windows by site based on year of CDW volume data
# mamSiteYears <- plotCdwVol %>%
#   dplyr::ungroup() %>%
```

```
#   dplyr::distinct(domainID, siteID, eventID) %>%
#   dplyr::mutate(
#     cdwYear = as.numeric(stringr::str_extract(eventID, "20[0-9]{2}")),
#     minMamYear = cdwYear - 1,
#     maxMamYear = cdwYear + 1
#   )
#
# #   Calculate plot-specific MNKA across bouts: Join plotBoutMNKA and mamSiteYears, filter to records
# #   3-year window centered on CDW measurement year, and determine maximum small mammal abundance for
# #   plot, as well as other per plot metrics.
# plotMNKA <- plotBoutMNKA %>%
#   dplyr::left_join(
#     mamSiteYears %>%
#       dplyr::select(
#         siteID,
#         minMamYear,
#         maxMamYear
#       ),
#     by = "siteID"
#   ) %>%
#   dplyr::filter(
#     year >= minMamYear,
#     year <= maxMamYear
#   ) %>%
#   dplyr::group_by(siteID, plotID) %>%
#   dplyr::summarise(
#     plotMeanAbundance = round(mean(n), digits = 1),
#     plotMedAbundance = median(n),
#     plotMinAbundance = min(n),
#     plotMaxAbundance = max(n),
#     plotBoutCount = n()
#   ) %>%
#   dplyr::ungroup()
#
#
# ##  Display table of plot-level MNKA parameters
# DT::datatable(
#   data = plotMNKA %>% dplyr::rename(
#     "Mean Abundance (count/ha)" = plotMeanAbundance,
#     "Median Abundance (count/ha)" = plotMedAbundance,
#     "Min Abundance (count/ha)" = plotMinAbundance,
#     "Max Abundance (count/ha)" = plotMaxAbundance,
#     "Bout Count by Plot" = plotBoutCount
#   ),
#   filter = "none",
#   rownames = FALSE,
#   options = list(pageLength = 10),
#   caption = "Small Mammal abundance metrics (count/ha) by plotID for selected sites for a 3-year
#   window centered on the year that CDW volume was measured."
# )
#
#
# ### Calculate MNKA metrics per site based on plot-level data
```

```
# siteMNKA <- plotMNKA %>%
#   dplyr::group_by(siteID) %>%
#   dplyr::summarise(
#     siteMeanAbundance = round(mean(plotMeanAbundance), digits = 1), #--> mean abundance across plots
#     siteMedAbundance = round(median(plotMedAbundance), digits = 1), #--> median abundance across plot
#     siteMinAbundance = round(mean(plotMinAbundance), digits = 1), #--> average min abundance across p
#     siteMaxAbundance = round(mean(plotMaxAbundance), digits = 1) #--> average max abundance across pl
#   ) %>%
#   dplyr::ungroup()
#
#
# ##  Display table of site-level MNKA parameters
# DT::datatable(
#   data = siteMNKA %>% dplyr::rename(
#     "Mean Abundance (count/ha)" = siteMeanAbundance,
#     "Median Abundance (count/ha)" = siteMedAbundance,
#     "Average Min Abundance (count/ha)" = siteMinAbundance,
#     "Average Max Abundance (count/ha)" = siteMaxAbundance
#   ),
#   filter = "none",
#   rownames = FALSE,
#   options = list(pageLength = 10),
#   caption = "Small Mammal abundance metrics (count/ha) by siteID for selected sites for a 3-year
#   window centered on the year that CDW volume was measured."
# )
```

Create unified CDW and MAM datasets

To create plot-level and site-level data sets suitable for plotting and correlation analyses, we join the CDW volume and Small Mammal abundance data at both plot and site scales.

```
# ### Join plot-level CDW and MAM datasets
# plotDF <- plotCdwVol %>%
#   dplyr::left_join(
#     plotMNKA %>%
#       dplyr::select(-plotBoutCount),
#     by = c("siteID", "plotID")
#   ) %>%
#   dplyr::filter(
#     !is.na(plotMaxAbundance)
#   )
#
#
# ### Join site-level CDW and MAM datasets
# siteDF <- siteCdwVol %>%
#   dplyr::left_join(
#     siteMNKA,
#     by = "siteID"
#   )
```

Create graphs of CDW and MAM data at plot and site spatial scales

We first examine the potential relationship between CDW volume and Small Mammal abundance using data from all spatially colocated plots from all of the six forested sites we selected (HARV, JERC, STEI, UKFS, RMNP, and ABBY).

```
# ### Create graph of CDW volume vs. MAM maximum abundance across all plots and sites
# plotsGraph <- ggplot2::ggplot(
#   data = plotDF,
#   mapping = aes(
#     x = cdwVol,
#     y = plotMaxAbundance
#   )
# ) +
#   ggplot2::geom_point(
#     alpha = 0.5,
#     size = 3
#   ) +
#   ggplot2::geom_smooth() +
#   ggplot2::labs(
#     x = "CDW Volume (m3/ha)",
#     y = "Max Small Mammal Abundance (count/ha)",
#     title = "Figure 1. Correlation between CDW Volume and Max Small Mammal
#     Abundance at the plot scale across forested sites."
#   )
#
# plotsGraph
```

- Across a selection of close-canopy forested NEON sites, measurements of CDW volume (m^3/ha) and Small Mammal abundance (count/ha) reveal no clear correlation at the colocated plot-scale.
- The Pearson's correlation coefficient is: "r round(cor(plotDF$cdwVol, plotDF$plotMaxAbundance), digits = 2)".

Next, we check for correlations between CDW volume and Small Mammal abundance within sites using colocated plot-scale data.

```
# ### Create graph of CDW volume vs. MAM maximum abundance across plots within site
# sitePlotsGraph <- ggplot2::ggplot(
#   data = plotDF,
#   mapping = aes(
#     x = cdwVol,
#     y = plotMaxAbundance
#   )
# ) +
#   ggplot2::geom_point(
#     alpha = 0.5,
#     size = 3
#   ) +
#   ggplot2::geom_smooth(method = lm) +
#   ggplot2::facet_wrap(
#     facets = ~siteID,
#     ncol = 3,
#     scales = "free"
#   ) +
#   ggplot2::labs(
#     x = "CDW Volume (m3/ha)",
```

```
#     y = "Max Small Mammal Abundance (count/ha)",
#     title = "Figure 2. Correlation between CDW Volume and Max Small Mammal
#     Abundance by plot within site."
#   )
#
# sitePlotsGraph
```

- Within a selection of close-canopy forested NEON sites, CDW volume (m^3/ha) and Small Mammal abundance (count/ha) show positive (RMNP, $r_{Pearson}$ = "r round(cor(plotDF$cdwVol$[$plotDF$siteID=="RMNP"], plotDF$plotMaxAbundance$[$plotDF$siteID=="RMNP"]), digits = 2)"), negative (ABBY, $r_{Pearson}$ ="r round(cor(plotDF$cdwVol$[$plotDF$siteID=="ABBY"], plotDF$plotMaxAbundance$[$plotDF$siteID=="ABBY"]), digits = 2)"), and weak correlations (HARV, JERC, STEI, UKFS).

Finally, we check for correlations between mean CDW volume and average maximum Small Mammal abundance at the site scale.

```
# ### Create graph of CDW volume vs. MAM maximum abundance across sites
# sitesGraph <- ggplot2::ggplot(
#   data = siteDF,
#   mapping = aes(
#     x = meanCdwVol,
#     y = siteMaxAbundance,
#   )
# ) +
#   ggplot2::geom_point(
#     alpha = 0.5,
#     size = 3
#   ) +
#   ggplot2::geom_smooth(method = lm) +
#   ggplot2::labs(
#     x = "CDW Volume (m3/ha)",
#     y = "Average Max Small Mammal Abundance (count/ha)",
#     title = "Figure 3. Correlation between CDW Volume and average max Small Mammal
#     Abundance across sites."
#   )
#
# sitesGraph
```

- Across a selection of closed-canopy forested NEON sites, CDW volume (m^3/ha) and Small Mammal abundance (count/ha) show a weak, negative correlation at the site scale ($r_{Pearson}$ = "r round(cor(siteDF$meanCdwVol$, $siteDF$siteMaxAbundance), digits = 2)").

Conclusions

- Coarse Downed Wood volume (m^3/ha) and Small Mammal abundance (count/ha) can each be calculated at the plot-level, and spatially colocated plots are identified by a shared `plotID`.
  - It is possible to identify shared `plotID` values programmatically using the `geoNEON::getLocByName()` function (lines 159-164), or by downloading the CDW and Small Mammal datasets and joining via the `plotID` (lines 838-858).
  - Note that more `plotID` values may be identified as colocated via the `geoNEON` function than via the table joining method due to the fact that `getLocByName()` identifieds all `plotIDs` that have ever been used for a given protocol, and not just those locations that are currently in use.
- Plot-scale spatial collocation of TOS data products allows data users to investigate patterns within NEON sites.

- All NEON TOS data products are colocated at the site-scale. Site-level estimates may be useful for investigating patterns at very large spatial scales.

References

Affleck, D.L.R. (2008) A line intersect distance sampling strategy for downed wood inventory. *Canadian Journal of Forest Research*, 38, 2262-2273.

Affleck, D.L.R. (2010) On the efficiency of line intersect distance sampling. *Canadian Journal of Forest Research*, 40, 1086-1094.

Slade, N.A. and Blair, S.M. (2000) An empirical test of using counts of individuals captured as indices of population size. *Journal of Mammalogy*, 81, 1035–1045.

Reproducibility

Code was developed and tested using `R v4.1.2`, and key packages with versions as shown below. Worked example code may generate errors if key package versions are earlier than those indicated. Source code is available via GitHub: https://github.com/NEONScience/mee-tos-2022