Preprocessing Data for Estimates of Realized Response to Selection in *Chamaecrista fasciculata* and Decomposition into Environmental and Genetic Parts

Charles J. Geyer* Mason W. Kulbaba† Seema N. Sheth‡ Rachel E. Pain§
Vincent M. Eckhart¶ Ruth G. Shaw‖

July 12, 2025

Contents

*School of Statistics, University of Minnesota, geyer@umn.edu, https://orcid.org/0000-0003-1471-1703
†St. Mary's University, mason.kulbaba@stmu.ca, https://orcid.org/0000-0003-0619-7089
‡Department of Plant and Microbial Biology, North Carolina State University, ssheth3@ncsu.edu, https://orcid.org/0000-0001-8284-7608
§Ecology, Evolution and Behavior Graduate Program, University of Minnesota, repain@umn.edu
¶Department of Biology, Grinnell College, eckhart@grinnell.edu
‖Department of Ecology, Evolution and Behavior, University of Minnesota, shawx016@umn.edu, https://orcid.org/0000-0001-5980-9291

1 License

The git repository for these analyses is https://github.com/cjgeyer/mf. This repo is not currently public.

2 R

- The version of R used to make this document is 4.5.1.
- The version of the `rmarkdown` package used to make this document is 2.29.
- The version of the `bookdown` package used to make this document is 0.43.
- The version of the `knitr` package used to make this document is 1.50.

3 Data

3.1 Data Files

For the analyses here. the data files are

```
files <- list.files(pattern = "*.csv")
files
```

```
## [1] "CS_2015_planting_data.csv"  "csdat.csv"
## [3] "GC_2015_planting_data.csv"  "GCdat.csv"
## [5] "KW_2015_planting_data.csv"  "KW.planting.data.final.csv"
## [7] "KWdat.csv"
```

where

- CS indicates Conard Environmental Research Area (CERA),
- GC indicates Grey Cloud Dunes Scientific and Natural Area, and
- KW indicates Kellogg-Weaver Dunes, also called McCarthy Lake,

respectively. These files include the same data on the same individuals as in the data files used by Kulbaba *et al.* (2019) and Geyer *et al.* (2022) but also include more individuals, who are offspring of those analyzed before. For more details, see Kulbaba *et al.* (2019) and Section 3.3 below.

As can be seen, we have three kinds of files.

- Those without `"planting"` in their names are the primary data files for the three sites.
- Those with `"planting_data"` in their names are secondary data files used to obtain pedigree information for some analyses.
- The one with `"planting.data"` in its name is a similar secondary data file that we will actually not need to use (but can use for checks).

Distinguish these file types.

```
files.planting <- grep("planting_data", files, value = TRUE)
files.primary <- grep("planting", files, value = TRUE, invert = TRUE)
files.secondary <- files |> setdiff(files.planting) |> setdiff(files.primary)
files.primary
```

```
## [1] "csdat.csv" "GCdat.csv" "KWdat.csv"
```

```
files.planting
```

```
## [1] "CS_2015_planting_data.csv" "GC_2015_planting_data.csv"
## [3] "KW_2015_planting_data.csv"
```

```
files.secondary
```

```
## [1] "KW.planting.data.final.csv"
```

Read in some of these files.

```
data.primary <- lapply(files.primary, read.csv)
data.planting <- lapply(files.planting, read.csv)
names(data.primary) <- substr(files.primary, 1, 2) |> toupper()
names(data.primary)
```

```
## [1] "CS" "GC" "KW"
```

```
names(data.planting) <- substr(files.planting, 1, 2)
names(data.planting)
```

```
## [1] "CS" "GC" "KW"
```

3.2 NA Cohort Labels

The variable `cohort` in these data frames tells us what generation the data are.

- `cohort == "greenhouse"` is the first generation planted. These are pedigreed crosses: both mother and father are known.
- `cohort %in% c("field", "G2YR2", "G2YR3")` is the second generation planted. These are open pollinated: mother is known, father is unknown.
- `cohort == "G3YR2"` is the third generation planted. We do not analyze these data in this analysis.

But some individuals are not labeled as to cohort.

```
sapply(data.primary, function(x) with(x, sum(is.na(cohort))))
```

```
## CS GC KW
##  0  0 10
```

What do these individuals look like?

```
lapply(data.primary, function(x) subset(x, is.na(cohort)))
```

```
## $CS
##  [1] block                position             row
##  [4] maternalID           paternalID           Germ
##  [7] flw                  total.pods           total.pods.collected
## [10] totalseeds           cohort               year
## <0 rows> (or 0-length row.names)
##
## $GC
##  [1] positionID           maternalID           paternalID
##  [4] position             row                  block
##  [7] Germ                 flw                  total.pods
## [10] total.pods.collected totalseeds           cohort
## [13] year                 seedpool
## <0 rows> (or 0-length row.names)
##
```

```
## $KW
##       maternalID paternalID position block row Germ flw total.pods
## 10937          0          0      8.8    1C 107    0   0          0
## 10938          0          0      8.9    1C 107    0   0          0
## 10939          0          0      9.0    1C 107    0   0          0
## 10940          0          0      9.1    1C 107    0   0          0
## 10941          0          0      9.2    1C 107    0   0          0
## 14852          0          0     40.8    5C 503    0   0          0
## 14853          0          0     40.9    5C 503    0   0          0
## 14854          0          0     41.0    5C 503    0   0          0
## 14855          0          0     41.1    5C 503    0   0          0
## 14856          0          0     41.2    5C 503    0   0          0
##       total.pods.collected totalseeds year cohort
## 10937                    0          0 2017   <NA>
## 10938                    0          0 2017   <NA>
## 10939                    0          0 2017   <NA>
## 10940                    0          0 2017   <NA>
## 10941                    0          0 2017   <NA>
## 14852                    0          0 2017   <NA>
## 14853                    0          0 2017   <NA>
## 14854                    0          0 2017   <NA>
## 14855                    0          0 2017   <NA>
## 14856                    0          0 2017   <NA>
```

Delete them.

```
data.primary <- lapply(data.primary, function(x) subset(x, ! is.na(cohort)))
```

3.3 Rationalize Cohort Labels

As we said above, these data have a variable `cohort` that is inconsistent among sites.

```
lapply(data.primary, function(x) with(x, split(year, cohort)) |> lapply(unique))
```

```
## $CS
## $CS$field
## [1] 2016
##
## $CS$G2YR3
## [1] 2017
##
## $CS$greenhouse
## [1] 2015 2016 2017
##
##
## $GC
## $GC$field
## [1] 2016
##
## $GC$G2YR2
## [1] 2017
##
## $GC$G3YR2
## [1] 2017
##
## $GC$greenhouse
```

```
## [1] 2015 2016 2017
##
##
## $KW
## $KW$field
## [1] 2016 2017
##
## $KW$greenhouse
## [1] 2015 2016 2017
```

- In site KW, all offspring individuals are labeled as cohort `"field"`.
- In site CS, offspring in 2016 are labeled as cohort `"field"` and those in 2017 are labeled as cohort `"G2YR3"` for generation 2 year 3 apparently because 2017 is the third year of the whole experiment.
- In site GC, offspring in 2016 are labeled as cohort `"field"` and those in 2017 are labeled as cohort `"G2YR2"` for generation 2 year 2 apparently because 2017 is the second year of the experiment *in which offspring were planted*.
- In site GC, there are also offspring of offspring labeled as cohort `"G3YR2"` for generation 3 year 2. We do not analyze these individuals in this analysis. So we omit them from the output file.

Drop the `"G3YR2"` individuals from GC.

```
data.primary <- lapply(data.primary, function(x) subset(x, cohort != "G3YR2"))
sapply(data.primary, function(x) unique(sort(x$cohort)))
```

```
## $CS
## [1] "field"      "G2YR3"      "greenhouse"
##
## $GC
## [1] "field"      "G2YR2"      "greenhouse"
##
## $KW
## [1] "field"      "greenhouse"
```

Now convert `"G2YR2"` or `"G2YR3"` to "field". These labels being now totally redundant, telling us nothing the year does not also say. In the following code chunk, `"G2YR[23]"` is a regular expression that matches either `"G2YR2"` or `"G2YR3"`.

```
data.primary <- lapply(data.primary, transform, cohort.orig = cohort,
    paternalID.orig = paternalID, maternalID.orig = maternalID) # DEBUG
data.primary <- lapply(data.primary,
    function(x) transform(x, cohort = sub("G2YR[23]", "field", cohort)))
lapply(data.primary, function(x) unique(sort(x$cohort)))
```

```
## $CS
## [1] "field"      "greenhouse"
##
## $GC
## [1] "field"      "greenhouse"
##
## $KW
## [1] "field"      "greenhouse"
```

```
sapply(data.primary, function(x) with(x, is.na(cohort)) |> sum())
```

```
## CS GC KW
##  0  0  0
```

Now `cohort` is consistent between sites.

3.4 Convert Parental ID Zero to NA

Parental ID Zero indicates "unknown" and hence should be converted to `NA` to be more R-ish.

```
data.primary <- lapply(data.primary, function(x) transform(x,
    paternalID = ifelse(paternalID == 0, NA_integer_, paternalID),
    maternalID = ifelse(maternalID == 0, NA_integer_, maternalID)))
```

3.5 Remove Individuals with Unknown Parents

For individuals in the `"greenhouse"` cohort, which were pedigreed crosses, we should have both parents recorded. Check this.

```
sapply(data.primary, function(x) with(x, cohort == "greenhouse" &
    (is.na(paternalID) | is.na(maternalID))) |> sum())
```

```
## CS GC KW
## 99  0 33
```

```
sapply(data.primary, function(x) with(x, cohort == "greenhouse" &
    is.na(paternalID)) |> sum())
```

```
## CS GC KW
## 99  0 33
```

Remove these individuals.

```
sapply(data.primary, nrow)
```

```
##    CS    GC    KW
##  8848 15475 17992
```

```
data.primary <- lapply(data.primary, function(x) subset(x,
    ! (cohort == "greenhouse" & is.na(paternalID))))
sapply(data.primary, nrow)
```

```
##    CS    GC    KW
##  8749 15475 17959
```

For individuals in the `"field"` cohort, which were open pollinated, sires are unknown and this is not a problem but rather by design. So we only remove individuals with sire unknown that are not in cohort `"field"`. We do however need to remove individuals whose mothers are unknown. Check this.

```
sapply(data.primary, function(x) with(x,
    cohort == "field" & is.na(maternalID)) |> sum())
```

```
## CS GC KW
##  1 24 38
```

And do the removal.

```
sapply(data.primary, nrow)
```

```
##    CS    GC    KW
##  8749 15475 17959
```

```
data.primary <- lapply(data.primary, function(x) subset(x,
    ! (cohort == "field" & is.na(maternalID))))
sapply(data.primary, nrow)
```

```
##    CS    GC    KW
##  8748 15451 17921
```

3.6 Final Check for Unknown Parents

```
sapply(data.primary, function(x) with(x, cohort == "greenhouse" &
    (is.na(paternalID) | is.na(maternalID))) |> sum())
```

```
## CS GC KW
##  0  0  0
```

```
sapply(data.primary, function(x) with(x, cohort == "field" &
    is.na(maternalID)) |> sum())
```

```
## CS GC KW
##  0  0  0
```

3.7 Check for Data Inconsistencies

The aster graph for these data is

$$1 \xrightarrow{\text{Ber}} \texttt{Germ} \xrightarrow{\text{Ber}} \texttt{flw} \xrightarrow{\text{Poi}} \texttt{total.pods} \xrightarrow{\text{samp}} \texttt{total.pods.collected} \xrightarrow{\text{Poi}} \texttt{totalseeds}$$

where the node labels correspond to variable names in the data frames and the arrow labels correspond to conditional distributions. Thus all of these variables must be nonnegative-integer-valued.

```
data.primary |> lapply(function(x)
    x[c("Germ", "flw", "total.pods", "total.pods.collected", "totalseeds")]) |>
    lapply(function(x) sapply(x, function(x) is.integer(x) & all(x >= 0)))
```

```
## $CS
##                 Germ                  flw           total.pods
##                 TRUE                 TRUE                 TRUE
## total.pods.collected           totalseeds
##                 TRUE                 TRUE
##
## $GC
##                 Germ                  flw           total.pods
##                 TRUE                 TRUE                 TRUE
## total.pods.collected           totalseeds
##                 TRUE                 TRUE
##
## $KW
##                 Germ                  flw           total.pods
##                 TRUE                 TRUE                 TRUE
## total.pods.collected           totalseeds
##                 TRUE                 TRUE
```

Then we must also have

```
sapply(data.primary, function(x) with(x, all(Germ <= 1)))
```

```
##   CS   GC   KW
## TRUE TRUE TRUE
```

```
sapply(data.primary, function(x) with(x, all(flw <= Germ)))
```

```
##   CS   GC   KW
## TRUE TRUE TRUE
```

```
sapply(data.primary, function(x)
    with(x, all(total.pods.collected <= total.pods)))
```

```
##    CS    GC    KW
##  TRUE FALSE FALSE
```

So this last check fails. We will repair in the following section (Section 3.8).

In the following block, (some logical expression) <= (some other logical expression) is an R-ism for the material implication operator, that is, left-hand logical expression implies right-hand logical expression (if the former is true, then the latter is true). This works because the logical expressions are converted to zero-or-one expressions to be compared with <=. The parentheses are necessary because of R's operator precedence.

```
sapply(data.primary, function(x)
    with(x, all((Germ == 0) <= (flw == 0))))
```

```
##   CS   GC   KW
## TRUE TRUE TRUE
```

```
sapply(data.primary, function(x)
    with(x, all((flw == 0) <= (total.pods == 0))))
```

```
##   CS   GC   KW
## TRUE TRUE TRUE
```

```
sapply(data.primary, function(x)
    with(x, all((total.pods == 0) <= (total.pods.collected == 0))))
```

```
##   CS   GC   KW
## TRUE TRUE TRUE
```

```
sapply(data.primary, function(x)
    with(x, all((total.pods.collected == 0) <= (totalseeds == 0))))
```

```
##   CS   GC   KW
## TRUE TRUE TRUE
```

So all of these checks pass.

How many individuals have problems?

```
lapply(data.primary, function(x)
    subset(x, total.pods.collected > total.pods)[c("total.pods", "total.pods.collected")])
```

```
## $CS
## [1] total.pods           total.pods.collected
## <0 rows> (or 0-length row.names)
##
## $GC
##       total.pods total.pods.collected
## 9623           1                   6
## 11358          1                   5
##
## $KW
##       total.pods total.pods.collected
## 10617          2                   3
## 10817          1                   3
## 10857          1                   7
## 10877          1                   2
```

```
## 11154          4                    5
## 11163          2                    5
## 11171          1                    2
## 13118          4                    5
## 14284          2                    4
## 14285          3                    4
```

3.8 Fix Data Inconsistencies

So we need to fix up these inconsistencies.

```
data.primary <- lapply(data.primary, function(x)
    transform(x, total.pods = pmax(total.pods, total.pods.collected)))
```

And recheck.

```
sapply(data.primary, function(x) with(x, all(Germ <= 1)))
```

```
##   CS   GC   KW
## TRUE TRUE TRUE
```

```
sapply(data.primary, function(x) with(x, all(flw <= Germ)))
```

```
##   CS   GC   KW
## TRUE TRUE TRUE
```

```
sapply(data.primary, function(x)
    with(x, all(total.pods.collected <= total.pods)))
```

```
##   CS   GC   KW
## TRUE TRUE TRUE
```

```
sapply(data.primary, function(x)
    with(x, all((Germ == 0) <= (flw == 0))))
```

```
##   CS   GC   KW
## TRUE TRUE TRUE
```

```
sapply(data.primary, function(x)
    with(x, all((flw == 0) <= (total.pods == 0))))
```

```
##   CS   GC   KW
## TRUE TRUE TRUE
```

```
sapply(data.primary, function(x)
    with(x, all((total.pods == 0) <= (total.pods.collected == 0))))
```

```
##   CS   GC   KW
## TRUE TRUE TRUE
```

```
sapply(data.primary, function(x)
    with(x, all((total.pods.collected == 0) <= (totalseeds == 0))))
```

```
##   CS   GC   KW
## TRUE TRUE TRUE
```

OK.

4 Find Grandfathers

4.1 Introduction

For the `"field"` cohort, we need to find maternal grandfathers (fathers are unknown). This is a complicated process, made even more complicated by having to do this two different ways.

4.2 Method 1 (Only for 2017)

For offspring (`"field"`) planted in 2017, the father is unknown but the variable `paternalID` stores the maternal grandfather (rather than the unknown indicator). This only makes sense for individuals with `cohort == "field"` of course (for other individuals paternal ID means paternal ID. Thus we get grandfathers as follows.

```
data.primary <- lapply(data.primary, function(x) transform(x,
    grandpaternalID = ifelse(cohort == "field" & year == 2017,
        paternalID, NA_integer_)))
```

4.3 Method 2 (Only for 2016)

For offspring (`"field"`) planted in 2016, we need a much more complicated method to find maternal grandfathers.

- The primary data file says where the individual was planted (variables `position` and `row`).
- The planting data file finds that same individual by those same variables (`position` and `row`).
- The planting data file gives for that individual where its mother was planted (variables `maternalposition` and `maternalrow`).
- If we look up up the mother in the primary data file (by position and row), then the `paternalID` for that mother is the required grandfather.

Still using the same data but now we need both primary and secondary data.

```
sites <- names(data.primary)
for (site in sites) {
    x <- data.primary[[site]]
    y <- data.planting[[site]]
    key.indiv <- with(x, paste(row, position, sep = ":"))
    key.plant <- with(y, paste(row, position, sep = ":"))
    idx <- match(key.indiv, key.plant)
    # now idx[i] is row index in planting data file of individual
    # having row index i in primary data file
    key.moms <- with(y, paste(maternalrow, maternalposition, sep = ":"))
    idx.moms <- match(key.moms[idx], key.indiv)
    # now idx.moms[i] is row index in primary data file of mother
    # of individual having row index i in primary data file
    data.primary[[site]] <- transform(x, grandpaternalID =
        ifelse(cohort == "field" & year == 2016,
            paternalID[idx.moms], grandpaternalID))
}
```

4.4 Check Grandfathers

```
sapply(data.primary, function(x)
    with(x, sum(cohort == "field" & is.na(grandpaternalID))))
```

```
## CS GC KW
##  0  3  0
```

Oops!

```
sapply(data.primary, function(x)
    with(x, year[cohort == "field" & is.na(grandpaternalID)]))
```

```
## $CS
## integer(0)
##
## $GC
## [1] 2017 2017 2017
##
## $KW
## integer(0)
```

Who are you?

```
subset(data.primary$GC, cohort == "field" & is.na(grandpaternalID))
```

```
##       positionID maternalID paternalID position row block Germ flw total.pods
## 11427        444        411         NA     38.8 204    2C    0   0          0
## 11428        445        411         NA     38.9 204    2C    0   0          0
## 11429        446        411         NA     39.0 204    2C    0   0          0
##       total.pods.collected totalseeds cohort year seedpool cohort.orig
## 11427                    0          0  field 2017        8       G2YR2
## 11428                    0          0  field 2017        8       G2YR2
## 11429                    0          0  field 2017        8       G2YR2
##       paternalID.orig maternalID.orig grandpaternalID
## 11427               0             411              NA
## 11428               0             411              NA
## 11429               0             411              NA
```

These individuals are misclassified as to "cohort". They are actually generation 3 and should be removed. This is proved by the planting data file for 2016, which is not available in this repo.

```
data.primary$GC <- subset(data.primary$GC,
    ! (cohort == "field" & is.na(grandpaternalID)))
```

5 Make Numerical Variables Categorical

Parental ID variables and `block` are numeric (integer). But statistically they are categorical. So we must make them type `factor` so R will treat them correctly.

```
data.primary <- lapply(data.primary, function(x)
    transform(x, maternalID = as.factor(maternalID),
    paternalID = as.factor(paternalID),
    grandpaternalID = as.factor(grandpaternalID),
    block = as.factor(block)))
```

6 Tidy Data

One more thing to do.

```
lapply(data.primary, names)
```

```
## $CS
##  [1] "block"                "position"             "row"
##  [4] "maternalID"           "paternalID"           "Germ"
##  [7] "flw"                  "total.pods"           "total.pods.collected"
## [10] "totalseeds"           "cohort"               "year"
## [13] "cohort.orig"          "paternalID.orig"      "maternalID.orig"
## [16] "grandpaternalID"
##
## $GC
##  [1] "positionID"           "maternalID"           "paternalID"
##  [4] "position"             "row"                  "block"
##  [7] "Germ"                 "flw"                  "total.pods"
## [10] "total.pods.collected" "totalseeds"           "cohort"
## [13] "year"                 "seedpool"             "cohort.orig"
## [16] "paternalID.orig"      "maternalID.orig"      "grandpaternalID"
##
## $KW
##  [1] "maternalID"           "paternalID"           "position"
##  [4] "block"                "row"                  "Germ"
##  [7] "flw"                  "total.pods"           "total.pods.collected"
## [10] "totalseeds"           "year"                 "cohort"
## [13] "cohort.orig"          "paternalID.orig"      "maternalID.orig"
## [16] "grandpaternalID"
```

```
common.names <- lapply(data.primary, names) |> Reduce(intersect, x = _)
common.names
```

```
##  [1] "block"                "position"             "row"
##  [4] "maternalID"           "paternalID"           "Germ"
##  [7] "flw"                  "total.pods"           "total.pods.collected"
## [10] "totalseeds"           "cohort"               "year"
## [13] "cohort.orig"          "paternalID.orig"      "maternalID.orig"
## [16] "grandpaternalID"
```

```
data.primary <- lapply(data.primary, function(x) x[common.names])
```

7 Write Out Data

```
save(data.primary, file = "mf.rda")
```

References

Geyer, C. J., Kulbaba, M. W., Sheth, S. N., et al. (2022) Correction for Kulbaba et al. (2019). *Evolution*, **76**, 3074. DOI: 10.1111/evo.14607.

Kulbaba, M. W., Sheth, S. N., Pain, R. E., et al. (2019) Additive genetic variance for lifetime fitness and the capacity for adaptation in an annual plant. *Evolution*, **73**, 1746–1758. DOI: 10.1111/evo.13830.