SGP Appendix 3: Monte Carlo Random Forest Analyses

Richard G. Stockey

March 2022

1. Filter primary dataset to generate proxy-specific subsets

First, import the primary dataset that was generated in Appendix 1.

```
load("Filtered.trace.toc.full.20230205.RData")
```

```
## Warning in load("Filtered.trace.toc.full.20230205.RData"): strings not
## representable in native encoding will be translated to UTF-8
```

Load required packages.

```
library(dplyr)
library(deeptime)
library(ggplot2)
library(randomForest)
library(doParallel)
library(foreach)
library(progressr)
library(stringi)
library(ggcorrplot)
#handlers(global = TRUE)
```

Update base of the Cryogenian in deeptime.

```
periods.edit <- deeptime::periods
periods.edit[14,2] <- 720
periods.edit[15,3] <- 720
periods.edit$color[13:15] <- c("#FED96A", "#FECC5C", "#FEBF4E")
```

Update non-latin characters for saving files

```
trace.toc.full<- filter(trace.toc.full, !(section.name == "Avalon-2"))
trace.toc.full$section.name <- stri_trans_general(trace.toc.full$section.name, "la
tin-ascii")
```

Categorical variables as factors

It will be useful for the treatment of samples with partial data that categorical geological context variables are factor objects, not character objects. We therefore assign all categorical variables to be factors now.

```
trace.toc.full$site.type <- factor(trace.toc.full$site.type)
trace.toc.full$metamorphic.bin <- factor(trace.toc.full$metamorphic.bin)
trace.toc.full$basin.type <- factor(trace.toc.full$basin.type)
trace.toc.full$environmental.bin <- factor(trace.toc.full$environmental.bin)
trace.toc.full$lithology.name <- factor(trace.toc.full$lithology.name)
```

We then filter this primary dataset to generate the specific subdatasets used in our primary analyses of different proxies. We further calculate the number of rows in each of these subset dataframes.

Molybdenum

For our primary Mo random forest analyses, we are interested in samples deposited in anoxic depositional environments that also have iron speciation (specifically Fepy/FeHR) data. We therefore filter the primary dataset to include only samples that are classified as anoxic based upon the iron speciation proxy (samples without iron speciation data that would be classified as anoxic based on Fe/Al will not have the require variables for the primary random forest analyses). We also remove any samples with no Mo, TOC and/or Fepy/FeHR data.

```
Mo.anox.py.rf <- trace.toc.full %>%
  filter(!is.na(Mo..ppm.) & !is.na(Fe.py.FeHR) & !is.na(TOC..wt..)) %>%
  filter(FeHR.FeT >= 0.38)

nrow(Mo.anox.py.rf)
```

```
## [1] 2355
```

Uranium

For our primary U random forest analyses, we are only interested in samples deposited in anoxic (ferruginous or euxinic) depositional environments. We therefore filter the primary dataset to include only samples that are classified as anoxic based upon iron speciation or Fe/Al ratios. We also remove any samples with no U and/or TOC data.

```
U.anox.rf <- trace.toc.full %>%
  filter(!is.na(U..ppm.) & !is.na(TOC..wt..)) %>%
  filter(FeHR.FeT >= 0.38 | FeT.Al >= 0.53)

nrow(U.anox.rf)
```

```
## [1] 3409
```

Proportion euxinic

For our primary random forest analyses of the proportion of samples that are euxinic, we are only interested in samples deposited in anoxic (ferruginous or euxinic) depositional environments. Given that we use iron speciation to determine the proportion of euxinic samples (and therefore all samples in this analyses must have full iron speciation data), we therefore filter the primary dataset to include only samples that are classified as anoxic based upon iron speciation data. We also remove samples with no Fepy/FeHR data, as with the other analyses (although the requirement of the filtering step to have FeHR/FeT data should achieve this as no samples should have partial iron speciation data).

```
Fepy.anox.rf <- trace.toc.full %>%
  filter(!is.na(Fe.py.FeHR)) %>%
  filter(FeHR.FeT >= 0.38)

nrow(Fepy.anox.rf)
```

```
## [1] 3909
```

For the analysis of the proportion of euxinic samples, we also need to code samples based upon whether they are euxinic (based on iron speciation) in a binary fashion, following Sperling et al. (2015, Nature).

```
Fepy.anox.rf$euxinic.Fe[Fepy.anox.rf$Fe.py.FeHR >= 0.7] <- 1
Fepy.anox.rf$euxinic.Fe[Fepy.anox.rf$Fe.py.FeHR < 0.7] <- 0
```

Total organic carbon

For our primary random forest analyses of total organic carbon we use no redox filter. We just remove samples with no TOC data.

```
TOC.all.rf <- trace.toc.full %>%
  filter(!is.na(TOC..wt..))

nrow(TOC.all.rf)
```

```
## [1] 12503
```

1b. Generate and plot cross-correlation matrices for all datasets.

Start by generating matrices. Need to just identify numerical variables and remove NAs

```
Mo.anox.py.rf.num <- Mo.anox.py.rf %>% select(site.latitude,
                                              site.longitude,
                                              interpreted.age,
                                              Mo..ppm.,
                                              TOC..wt..,
                                              Fe.py.FeHR,
                                              Al..wt..) %>%
  na.omit()


Mo.anox.py.corr <- round(cor(Mo.anox.py.rf.num), 1)



U.anox.rf.num <- U.anox.rf %>% select(site.latitude,
                                              site.longitude,
                                              interpreted.age,
                                              U..ppm.,
                                              TOC..wt..,
                                              Al..wt..) %>%
  na.omit()


U.anox.corr <- round(cor(U.anox.rf.num), 1)

Fepy.anox.rf.num <- Fepy.anox.rf %>% select(site.latitude,
                                              site.longitude,
                                              interpreted.age,
                                              euxinic.Fe,
                                              TOC..wt..,
                                              Al..wt..) %>%
  na.omit()

Fepy.anox.corr <- round(cor(Fepy.anox.rf.num), 1)


TOC.all.rf.num <- TOC.all.rf %>% select(site.latitude,
                                              site.longitude,
                                              interpreted.age,
                                              TOC..wt..,
                                              Al..wt..) %>%
  na.omit()

TOC.all.corr <- round(cor(TOC.all.rf.num), 1)
```

```
Mo.anox.py.corr.plot <- ggcorrplot(Mo.anox.py.corr, hc.order = TRUE, type = "lower
", lab = TRUE, ggtheme = ggplot2::theme_void, colors = c("#6D9EC1", "white", "#E46
726"))

ggsave("Figure Sx cross correlation between variables of Monte Carlo random forest
datasets 20240207 100 iterations Mo.pdf", height=5, width=6)

U.anox.corr.plot <- ggcorrplot(U.anox.corr, hc.order = TRUE, type = "lower", lab =
TRUE, ggtheme = ggplot2::theme_void, colors = c("#6D9EC1", "white", "#E46726"))

ggsave("Figure Sx cross correlation between variables of Monte Carlo random forest
datasets 20240207 100 iterations U.pdf", height=5, width=6)

Fepy.anox.corr.plot <- ggcorrplot(Fepy.anox.corr, hc.order = TRUE, type = "lower",
lab = TRUE, ggtheme = ggplot2::theme_void, colors = c("#6D9EC1", "white", "#E46726
"))

ggsave("Figure Sx cross correlation between variables of Monte Carlo random forest
datasets 20240207 100 iterations Fepy.pdf", height=5, width=6)

TOC.all.corr.plot <- ggcorrplot(TOC.all.corr, hc.order = TRUE, type = "lower", lab
= TRUE, ggtheme = ggplot2::theme_void, colors = c("#6D9EC1", "white", "#E46726"))

ggsave("Figure Sx cross correlation between variables of Monte Carlo random forest
datasets 20240207 100 iterations TOC.pdf", height=5, width=6)

# corr.sum <- ggarrange2(Mo.anox.py.corr.plot,
#             U.anox.corr.plot,
#             Fepy.anox.corr.plot,
#             TOC.all.corr.plot,
#             ncol=2)
#
# ggsave("Figure Sx cross correlation between variables of Monte Carlo random fore
st datasets 20240207 100 iterations.pdf", corr.sum, height=15, width=17)
```

2. Treatment of samples with partial data

Age uncertainty

We define a function to assign age uncertainties to samples with missing maximum and minimum age estimates (i.e. only interpreted age). We add ±5Myrs uncertainty on Phanerozoic ages and ±12.5Myrs on Neoproterozoic ages as default.

```
age.unc <- function(data, Phanerozoic = 5, Proterozoic = 12.5){

  # select which of the vector c(5,12.5) is appropriate by identifying which of th
e samples without min/max age have an interpreted age greater than 541. If TRUE, a
dding 1 to "TRUE" (1) gives 2 (i.e. 12.5Myr error from the vector c(5,12.5)). Addi
ng 1 to "FALSE" (0) gives 1 (i.e. 5Myr error from the vector c(5,12.5)).
    data$min.age[is.na(data$min.age)] <- data$interpreted.age[is.na(data$min.age)]
- c(5,12.5)[(data$interpreted.age[is.na(data$min.age)] > 541) + 1]

  data$max.age[is.na(data$max.age)] <- data$interpreted.age[is.na(data$max.age)] +
c(5,12.5)[(data$interpreted.age[is.na(data$max.age)] > 541) + 1]

  return(data)
}
```

Confirm that age assignment function works as intended using primary Mo subsdataset as an example. Here we look only at the samples with missing ages by filtering the dataset.

```
Mo.anox.py.rf.missing.ages <- Mo.anox.py.rf %>%
  filter(is.na(min.age) & is.na(max.age))

Mo.anox.py.rf.missing.ages <- age.unc(data = Mo.anox.py.rf.missing.ages)
```

Are all Phanerozoic samples that had missing max and min ages in the middle of their assigned age uncertainty?

Generate summary of median of age uncertainty minus interpreted age for newly assigned Phanerozoic samples.

```
summary(
  rowMeans(
    cbind(Mo.anox.py.rf.missing.ages$max.age[Mo.anox.py.rf.missing.ages$interprete
d.age <= 541], Mo.anox.py.rf.missing.ages$min.age[Mo.anox.py.rf.missing.ages$inter
preted.age <= 541])) -
    Mo.anox.py.rf.missing.ages$interpreted.age[Mo.anox.py.rf.missing.ages$interpre
ted.age <= 541]
)
```

```
##       Min.    1st Qu.     Median       Mean    3rd Qu.       Max.
## -5.684e-14  0.000e+00  0.000e+00  5.812e-17  0.000e+00  5.684e-14
```

All values are zero within the precision of R (tiny deviations are floating point errors).

Do all Phanerozoic samples that had missing max and min ages have 10 Myr age uncertainty? Generate summary of age uncertainty for newly assigned Phanerozoic samples.

```
summary(Mo.anox.py.rf.missing.ages$max.age[Mo.anox.py.rf.missing.ages$interpreted.
age <= 541] - Mo.anox.py.rf.missing.ages$min.age[Mo.anox.py.rf.missing.ages$interp
reted.age <= 541])
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##      10      10      10      10      10      10
```

All values are 10Myrs.

Are all Neoproterozoic samples that had missing max and min ages in the middle of their assigned age uncertainty?

Generate summary of median of age uncertainty minus interpreted age for newly assigned Neoproterozoic samples.

```
summary(
  rowMeans(
    cbind(Mo.anox.py.rf.missing.ages$max.age[Mo.anox.py.rf.missing.ages$interpreted.age > 541], Mo.anox.py.rf.missing.ages$min.age[Mo.anox.py.rf.missing.ages$interpreted.age > 541])) -
    Mo.anox.py.rf.missing.ages$interpreted.age[Mo.anox.py.rf.missing.ages$interpreted.age > 541]
)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       0       0       0       0       0       0
```

All values are zero.

Do all Neoproterozoic samples that had missing max and min ages have 25 Myr age uncertainty? Generate summary of age uncertainty for newly assigned Neoproterozoic samples.

```
summary(Mo.anox.py.rf.missing.ages$max.age[Mo.anox.py.rf.missing.ages$interpreted.age > 541] - Mo.anox.py.rf.missing.ages$min.age[Mo.anox.py.rf.missing.ages$interpreted.age > 541])
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##      25      25      25      25      25      25
```

All values are 25Myrs.

Context data

We define a function to randomly assign values for geologic context variables used in random forest models in cases where a sample is missing data for that variable. This function is then used in the Monte Carlo approach detailed below to maximize the number of samples used and evaluate uncertainty associated with missing data by repeating the random assignments and statistical learning models n times per analysis (n = 100 times in this study).

```
partial.context <- function(data,
                            site.type,
                            metamorphic.bin,
                            basin.type,
                            site.latitude,
                            site.longitude,
```

```
                              environmental.bin,
                              lithology.name){

  # Note that this function currently only randomly generates data from factor lev
els within the dataset "data" when generating data for categorical variables. Coul
d hard code although all levels seem to be present in most if not all datasets.

  # Each context variable must be set to "TRUE" (e.g. site.type = TRUE) inside fun
ction in order to be randomly assigned in the function partial.context().

  # levels check - make sure that numerical identification of empty factor name (i
.e. missing data) is correct (just in case of changes in SGP data download structu
re)
  levels.check.total <- 0 # total number of context variables levels are checked f
or
  levels.check.true <- 0 # number of levels that missing name is correctly identif
ied

  if(site.type == TRUE){
    # levels check (see above)
    levels.check.total <- levels.check.total +1
    if(levels(data$site.type)[1] == ""){
      levels.check.true <- levels.check.true + 1
    }
    # random assignment
    data$site.type[data$site.type == "cuttings"] <- "core" ## Combine core and cut
tings
    data$site.type <- factor(data$site.type) ## omit unused factor levels
    data$site.type[data$site.type == ""] <- sample(x = levels(data$site.type)[2:nl
evels(data$site.type)], size = nrow(filter(data, site.type == "")), replace=TRUE)
## assign all empty cells one of "core" or "outcrop". Starting at level 2 omits ce
lls with no data from vector to sample.
    data$site.type <- factor(data$site.type) ## omit unused factor levels
  }

  if(metamorphic.bin == TRUE){
    # levels check (see above)
    levels.check.total <- levels.check.total +1
    if(levels(data$metamorphic.bin)[1] == ""){
      levels.check.true <- levels.check.true + 1
    }
    # random assignment
    data$metamorphic.bin[data$metamorphic.bin == ""] <- sample(x = levels(data$met
amorphic.bin)[2:nlevels(data$metamorphic.bin)], size = nrow(filter(data, metamorph
ic.bin == "")), replace=TRUE) ## assign all empty cells one of "Anchizone", "Diage
netic zone" or "Epizone". Starting at level 2 omits cells with no data from vector
to sample.
    data$metamorphic.bin <- factor(data$metamorphic.bin) ## omit unused factor lev
els
  }

   if(basin.type == TRUE){
    # levels check (see above)
    levels.check.total <- levels.check.total +1
```

```
    if(levels(data$basin.type)[1] == ""){
      levels.check.true <- levels.check.true + 1
    }
    # random assignment
    data$basin.type[data$basin.type == ""] <- sample(x = levels(data$basin.type)[2
:nlevels(data$basin.type)], size = nrow(filter(data, basin.type == "")), replace=T
RUE) ## assign all empty cells one of the other basin types in the dataset. Starti
ng at level 2 omits cells with no data from vector to sample.
    data$basin.type <- factor(data$basin.type) ## omit unused factor levels
  }

   if(site.latitude == TRUE){
    data$site.latitude[is.na(data$site.latitude)] <- runif(nrow(filter(data, is.na
(site.latitude))), -90, 90) ## assign all empty cells a random latitude from a uni
form distribution between -90 and 90 degrees.
   }

   if(site.longitude == TRUE){
    data$site.longitude[is.na(data$site.longitude)] <- runif(nrow(filter(data, is.
na(site.longitude))), -180, 180) ## assign all empty cells a random longitude from
a uniform distribution between -180 and 180 degrees.
   }

   if(environmental.bin == TRUE){
    # levels check (see above)
    levels.check.total <- levels.check.total +1
    if(levels(data$environmental.bin)[1] == ""){
      levels.check.true <- levels.check.true + 1
    }
    # random assignment
    data$environmental.bin[data$environmental.bin == ""] <- sample(x = levels(data
$environmental.bin)[2:nlevels(data$environmental.bin)], size = nrow(filter(data, e
nvironmental.bin == "")), replace=TRUE) ## assign all empty cells one of the other
environmental bins in the dataset. Starting at level 2 omits cells with no data fr
om vector to sample.
    data$environmental.bin <- factor(data$environmental.bin) ## omit unused factor
levels
   }

     if(lithology.name == TRUE){
    # levels check (see above)
    levels.check.total <- levels.check.total +1
    if(levels(data$lithology.name)[1] == ""){
      levels.check.true <- levels.check.true + 1
    }
    # random assignment
    data$lithology.name[data$lithology.name == ""] <- sample(x = levels(data$litho
logy.name)[2:nlevels(data$lithology.name)], size = nrow(filter(data, lithology.nam
e == "")), replace=TRUE) ## assign all empty cells one of the other lithologies in
the dataset. Starting at level 2 omits cells with no data from vector to sample.
    data$lithology.name <- factor(data$lithology.name) ## omit unused factor level
s
     }
  # Check that the level with no factor name has been correctly identified (this s
```

```
hould always be true but is worth checking, especially if applied to different dat
a). Round is used to accommodate numbers very close to 1 due to possible floating
point errors (rounded to 3 decimal places).
  if(round((levels.check.true/levels.check.total), digits = 3) == 1){
    print("Partial context randomly assigned correctly - missing data identified c
orrectly")
  }else{
    print("ERROR - missing data NOT identified correctly")
  }
  return(data)
}
```

Test function on subdataset for primary Mo analyses.

Summary before partial.context function.

```
summary(Mo.anox.py.rf)
```

```
##  sample.identifier sample.original.name  height.depth      section.name
##  Min.   :    9     Length:2355          Min.   :  -76.00   Length:2355
##  1st Qu.: 3038     Class :character     1st Qu.:   20.95   Class :character
##  Median : 6795     Mode  :character     Median :   80.82   Mode  :character
##  Mean   :19037                          Mean   :  554.01
##  3rd Qu.:13388                          3rd Qu.:  252.40
##  Max.   :95921                          Max.   :10359.00
##                                         NA's   :92
##     site.type    site.latitude    site.longitude                  basin.type
##          :   0   Min.   :-33.00   Min.   :-140.94   rift                :1318
##  core    : 725   1st Qu.: 28.49   1st Qu.:-135.62   passive margin      : 345
##  cuttings:  48   Median : 54.07   Median : -74.56                       : 193
##  outcrop :1582   Mean   : 46.43   Mean   : -25.64   intracratonic sag   : 167
##                  3rd Qu.: 65.87   3rd Qu.: 103.06   retro-arc foreland : 143
##                  Max.   : 79.97   Max.   : 139.00   peripheral foreland: 114
##                  NA's   :33       NA's   :33        (Other)             :  75
##        metamorphic.bin interpreted.age    max.age         min.age
##                 : 226   Min.   :301.0   Min.   :329.8   Min.   :329.0
##  Anchizone      :1561   1st Qu.:425.6   1st Qu.:410.8   1st Qu.:404.0
##  Diagenetic zone: 518   Median :503.0   Median :433.4   Median :430.5
##  Epizone        :  50   Mean   :502.5   Mean   :441.3   Mean   :435.4
##                         3rd Qu.:551.0   3rd Qu.:481.7   3rd Qu.:480.0
##                         Max.   :830.0   Max.   :850.0   Max.   :811.5
##                                         NA's   :1605    NA's   :1608
##  long.stratigraphy.name stratigraphy.name    environmental.bin
##  Length:2355            Length:2355                    :  20
##  Class :character       Class :character   basinal     :1730
##  Mode  :character       Mode  :character   inner shelf: 179
##                                            outer shelf: 426
##
##
##
##       lithology.name    Al..wt..         Fe..wt..         Mo..ppm.
##  shale        :1743   Min.   : 0.090   Min.   : 0.050   Min.   :  0.000
##  mudstone     : 337   1st Qu.: 3.700   1st Qu.: 1.440   1st Qu.:  2.000
```

```
##  siltstone    : 139   Median : 5.500   Median : 2.420   Median :  6.537
##               :  94   Mean   : 5.742   Mean   : 2.749   Mean   : 28.400
##  lime mudstone:  36   3rd Qu.: 7.690   3rd Qu.: 3.770   3rd Qu.: 29.607
##  phosphorite  :   6   Max.   :22.790   Max.   :38.500   Max.   :499.000
##  (Other)      :   0   NA's   :260      NA's   :37
##     U..ppm.          FeHR.FeT        Fe.py.FeHR         FeT.Al
##  Min.   :  0.02   Min.   :0.3800   Min.   :0.0000   Min.   : 0.0400
##  1st Qu.:  3.00   1st Qu.:0.5200   1st Qu.:0.2000   1st Qu.: 0.3400
##  Median :  5.64   Median :0.6800   Median :0.5500   Median : 0.4700
##  Mean   : 11.51   Mean   :0.6929   Mean   :0.4881   Mean   : 0.5732
##  3rd Qu.: 13.00   3rd Qu.:0.8300   3rd Qu.:0.7500   3rd Qu.: 0.6300
##  Max.   :295.29   Max.   :6.5000   Max.   :1.0000   Max.   :34.0700
##  NA's   :426                                        NA's   :254
##    TOC..wt..
##  Min.   : 0.000
##  1st Qu.: 0.800
##  Median : 2.020
##  Mean   : 2.979
##  3rd Qu.: 3.855
##  Max.   :31.280
##
```

Run function, then check summary after partial.context function.

```
Mo.anox.py.rf.partial.test <- partial.context(data = Mo.anox.py.rf,
                            site.type = TRUE,
                            metamorphic.bin = TRUE,
                            basin.type = TRUE,
                            site.latitude = TRUE,
                            site.longitude = TRUE,
                            environmental.bin = TRUE,
                            lithology.name = TRUE)
```

```
## [1] "Partial context randomly assigned correctly - missing data identified corr
ectly"
```

```
summary(Mo.anox.py.rf.partial.test)
```

```
##  sample.identifier sample.original.name  height.depth      section.name
##  Min.   :    9     Length:2355          Min.   :  -76.00   Length:2355
##  1st Qu.: 3038     Class :character     1st Qu.:   20.95   Class :character
##  Median : 6795     Mode  :character     Median :   80.82   Mode  :character
##  Mean   :19037                          Mean   :  554.01
##  3rd Qu.:13388                          3rd Qu.:  252.40
##  Max.   :95921                          Max.   :10359.00
##                                         NA's   :92
##    site.type    site.latitude    site.longitude                  basin.type
##  core   : 773   Min.   :-83.55   Min.   :-179.39   rift                :1338
##  outcrop:1582   1st Qu.: 28.29   1st Qu.:-135.62   passive margin      : 371
##                 Median : 53.97   Median : -74.56   intracratonic sag   : 194
##                 Mean   : 45.85   Mean   : -25.63   retro-arc foreland : 166
```

```
##                 3rd Qu.: 65.87   3rd Qu.: 103.06   peripheral foreland: 141
##                 Max.   : 89.10   Max.   : 177.31   back-arc           :  90
##                                                    (Other)            :  55
##         metamorphic.bin interpreted.age    max.age         min.age
##  Anchizone      :1629   Min.   :301.0   Min.   :329.8   Min.   :329.0
##  Diagenetic zone: 593   1st Qu.:425.6   1st Qu.:410.8   1st Qu.:404.0
##  Epizone        : 133   Median :503.0   Median :433.4   Median :430.5
##                         Mean   :502.5   Mean   :441.3   Mean   :435.4
##                         3rd Qu.:551.0   3rd Qu.:481.7   3rd Qu.:480.0
##                         Max.   :830.0   Max.   :850.0   Max.   :811.5
##                                         NA's   :1605    NA's   :1608
##  long.stratigraphy.name stratigraphy.name    environmental.bin
##  Length:2355            Length:2355        basinal    :1737
##  Class :character       Class :character   inner shelf: 184
##  Mode  :character       Mode  :character   outer shelf: 434
##
##
##
##
##        lithology.name    Al..wt..          Fe..wt..          Mo..ppm.
##  shale         :1749   Min.   : 0.090   Min.   : 0.050   Min.   :  0.000
##  mudstone      : 345   1st Qu.: 3.700   1st Qu.: 1.440   1st Qu.:  2.000
##  siltstone     : 147   Median : 5.500   Median : 2.420   Median :  6.537
##  lime mudstone:  43   Mean   : 5.742   Mean   : 2.749   Mean   : 28.400
##  metasiltstone:  11   3rd Qu.: 7.690   3rd Qu.: 3.770   3rd Qu.: 29.607
##  argillite     :  10   Max.   :22.790   Max.   :38.500   Max.   :499.000
##  (Other)       :  50   NA's   :260      NA's   :37
##     U..ppm.           FeHR.FeT        Fe.py.FeHR          FeT.Al
##  Min.   :  0.02   Min.   :0.3800   Min.   :0.0000   Min.   : 0.0400
##  1st Qu.:  3.00   1st Qu.:0.5200   1st Qu.:0.2000   1st Qu.: 0.3400
##  Median :  5.64   Median :0.6800   Median :0.5500   Median : 0.4700
##  Mean   : 11.51   Mean   :0.6929   Mean   :0.4881   Mean   : 0.5732
##  3rd Qu.: 13.00   3rd Qu.:0.8300   3rd Qu.:0.7500   3rd Qu.: 0.6300
##  Max.   :295.29   Max.   :6.5000   Max.   :1.0000   Max.   :34.0700
##  NA's   :426                                        NA's   :254
##    TOC..wt..
##  Min.   : 0.000
##  1st Qu.: 0.800
##  Median : 2.020
##  Mean   : 2.979
##  3rd Qu.: 3.855
##  Max.   :31.280
##
```

Check that different random datasets are being generated each time.

Generate another test dataset and check they are different.

```
Mo.anox.py.rf.partial.test.2 <- partial.context(data = Mo.anox.py.rf,
                            site.type = TRUE,
                            metamorphic.bin = TRUE,
                            basin.type = TRUE,
                            site.latitude = TRUE,
                            site.longitude = TRUE,
                            environmental.bin = TRUE,
                            lithology.name = TRUE)
```

```
## [1] "Partial context randomly assigned correctly - missing data identified corr
ectly"
```

For site type

```
summary(Mo.anox.py.rf.partial.test$site.type)
```

```
##    core outcrop
##     773    1582
```

```
summary(Mo.anox.py.rf.partial.test.2$site.type)
```

```
##    core outcrop
##     773    1582
```

For metamorphic bin:

```
summary(Mo.anox.py.rf.partial.test$metamorphic.bin)
```

```
##       Anchizone Diagenetic zone         Epizone
##            1629             593             133
```

```
summary(Mo.anox.py.rf.partial.test.2$metamorphic.bin)
```

```
##       Anchizone Diagenetic zone         Epizone
##            1640             600             115
```

For basin type:

```
summary(Mo.anox.py.rf.partial.test$basin.type)
```

```
##            back-arc            fore-arc   intracratonic sag      passive margin
##                  90                  30                 194                 371
## peripheral foreland  retro-arc foreland               rift             wrench
##                 141                 166                1338                  25
```

```
summary(Mo.anox.py.rf.partial.test.2$basin.type)
```

```
##            back-arc            fore-arc   intracratonic sag      passive margin
##                  93                  29                 193                 369
## peripheral foreland  retro-arc foreland               rift             wrench
##                 144                 166                1336                  25
```

For latitude:

```
summary(Mo.anox.py.rf.partial.test$site.latitude)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  -83.55   28.29   53.97   45.85   65.87   89.10
```

```
summary(Mo.anox.py.rf.partial.test.2$site.latitude)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  -88.74   28.29   54.07   45.88   65.87   88.67
```

For longitude:

```
summary(Mo.anox.py.rf.partial.test$site.longitude)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## -179.39 -135.62  -74.56  -25.63  103.06  177.31
```

```
summary(Mo.anox.py.rf.partial.test.2$site.longitude)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## -175.92 -135.62  -74.56  -25.28  103.06  172.99
```

For environmental bin:

```
summary(Mo.anox.py.rf.partial.test$environmental.bin)
```

```
##     basinal inner shelf outer shelf
##        1737         184         434
```

```
summary(Mo.anox.py.rf.partial.test.2$environmental.bin)
```

```
##     basinal inner shelf outer shelf
##        1737         184         434
```

For lithology:

```
summary(Mo.anox.py.rf.partial.test$lithology.name)
```

```
##     argillite     claystone  dolomudstone lime mudstone metasiltstone
##            10             7            10            43            11
##      mudstone     oil shale        pelite   phosphorite         shale
##           345             1            10            10          1749
##          silt     siltstone         slate
##             8           147             4
```

```
summary(Mo.anox.py.rf.partial.test.2$lithology.name)
```

```
##     argillite     claystone  dolomudstone lime mudstone metasiltstone
##             8             4             9            41             7
##      mudstone     oil shale        pelite   phosphorite         shale
##           345             5             7            15          1752
##          silt     siltstone         slate
##             7           147             8
```

Al imputation

We define a function to estimate the lithology of samples with no Al concentration based upon lithology.

First, we make and save a lookup table including all samples in the main dataset, summarizing Al concentrations by lithology by calculating the median and the 25th and 75th percentiles (the bounds of the interquartile range). We make extra single-row table for all samples with identified lithologies.

```
Al.lookup <- trace.toc.full %>%
    filter(!(lithology.name == "") & !(is.na(Al..wt..))) %>%
    group_by(lithology.name) %>%
    summarize(median = median(Al..wt.., na.rm = TRUE), perc.25 = quantile(Al..w
t.., prob = 0.25, na.rm = T), perc.75 = quantile(Al..wt.., prob = 0.75, na.rm =
T))

# make extra single-row table for all samples with identified lithologies
Al.lookup.all <- trace.toc.full %>%
    filter(!(lithology.name == "") & !(is.na(Al..wt..))) %>%
    summarize(lithology.name = "all identified samples", median = median(Al..wt..,
na.rm = TRUE), perc.25 = quantile(Al..wt.., prob = 0.25, na.rm = T), perc.75 = qua
ntile(Al..wt.., prob = 0.75, na.rm = T))

save(Al.lookup, Al.lookup.all, file ="Al.lookup_trace.toc.full.RData")
```

We then define the imputation function. For each sample with identified lithology but missing [Al] data we randomly assign that sample an [Al] value from a uniform distribution with the 25th and 75th percentile [Al] concentrations for that lithology as the minimum and maximum bounds. For samples with no identified lithology that are missing [Al] data, we randomly assign that sample an [Al] value from a uniform distribution with the 25th and 75th percentile [Al] concentrations for all samples as the minimum and

maximum bounds. Note that in this dataset there is no Al data for “silt” samples - so we use data for “siltstone” for “silt” (note this only impacts one sample, from the Ediacaran Doushantuo Formation, that is only used in TOC analyses).

```
Al.impute <- function(data){
  load("Al.lookup_trace.toc.full.RData")

  for(row in 1:nrow(data)){
    if(is.na(data$Al..wt..[row]) == TRUE){
    # If lithology is unidentified, sample from full distribution of identified sa
mples
    if(data$lithology.name[row] == ""){
      data$Al..wt..[row] <- runif(1, Al.lookup.all$perc.25[Al.lookup.all$litholog
y.name == "all identified samples"], Al.lookup.all$perc.75[Al.lookup.all$litholog
y.name == "all identified samples"])
    }else if(data$lithology.name[row] == "silt"){
     # There is no Al data for "silt" - use data for "siltstone" for "silt" (note
this only impacts one sample, from the Ediacaran Doushantuo Formation, that is onl
y used in TOC analyses)
      data$Al..wt..[row] <- runif(1, Al.lookup$perc.25[Al.lookup$lithology.name ==
"siltstone"], Al.lookup$perc.75[Al.lookup$lithology.name == "siltstone"])
      }else {
     # Otherwise (except in the case of the two exceptions above), sample from the
lithology-specific Al distribution.
      data$Al..wt..[row] <- runif(1, Al.lookup$perc.25[Al.lookup$lithology.name ==
data$lithology.name[row]], Al.lookup$perc.75[Al.lookup$lithology.name == data$lith
ology.name[row]])
      }
    }
  }
  return(data)
}
```

Test the Al imputation method. Are all of the imputed Al values within the interquartile ranges established for their given lithology? Use primary Mo dataset as an example.

```
# tag samples missing Al with identified lithologies
Mo.anox.py.rf.no.Al.test <- Mo.anox.py.rf

Mo.anox.py.rf.no.Al.test$missing.Al[is.na(Mo.anox.py.rf.no.Al.test$Al..wt..) == TR
UE] <- TRUE
Mo.anox.py.rf.no.Al.test$missing.Al[is.na(Mo.anox.py.rf.no.Al.test$Al..wt..) == FA
LSE] <- FALSE

Mo.anox.py.rf.no.Al.test$missing.lith[Mo.anox.py.rf.no.Al.test$lithology.name ==
""] <- TRUE
Mo.anox.py.rf.no.Al.test$missing.lith[Mo.anox.py.rf.no.Al.test$lithology.name !=
""] <- FALSE
Mo.anox.py.rf.imputed.Al <- Al.impute(data = Mo.anox.py.rf.no.Al.test)


Mo.anox.py.rf.imputed.Al.no.Al.before <- filter(Mo.anox.py.rf.imputed.Al, (missin
g.Al == TRUE & missing.lith == FALSE))

Mo.anox.py.rf.imputed.Al.no.Al.before <- merge(Mo.anox.py.rf.imputed.Al.no.Al.befo
re, Al.lookup, id = c(lithology.name), all.x = TRUE)

for(row in 1:nrow(Mo.anox.py.rf.imputed.Al.no.Al.before)){
Mo.anox.py.rf.imputed.Al.no.Al.before$test[row] <- between(Mo.anox.py.rf.imputed.A
l.no.Al.before$Al..wt..[row], Mo.anox.py.rf.imputed.Al.no.Al.before$perc.25[row],
Mo.anox.py.rf.imputed.Al.no.Al.before$perc.75[row])
}
summary(Mo.anox.py.rf.imputed.Al.no.Al.before$test)
```

```
##    Mode    TRUE
## logical     257
```

It is true for all samples in this test dataset that all of the imputed [Al] values within the interquartile ranges established for their given lithology.

3. Age model

We define a function to randomly assign an age to each sample from within its prescribed age uncertainty. The age is assigned from a uniform distribution bounded by the minimum and maximum ages. This function is then used in the Monte Carlo approach detailed below to evaluate uncertainty associated with geologic ages by repeating the random assignments and statistical learning models n times per analysis (n = 100 in the analyses of this study). In a few cases (82 samples in trace.toc.full), the sample minimum and maximum assigned ages were flipped by SGP contributors during data entry. We correct for samples like this in the function. In even fewer cases (6 in trace.toc.full, from the same two Cambrian formations as the flipped samples above) there are samples where contributors have given samples the same max, min and interpreted age. Those samples are assigned their interpreted age.

```
age.model.basic <- function(data){

    # Default method (vast majority of samples)
    data$age.model[data$max.age > data$min.age] <- runif(n = nrow(filter(data, ma
x.age > min.age)), min = data$min.age[data$max.age > data$min.age], max = data$ma
x.age[data$max.age > data$min.age])

    # Samples with flipped max and min (note that max and min are flipped in runi
f())
    data$age.model[data$min.age > data$max.age] <- runif(n = nrow(filter(data, mi
n.age > max.age)), min = data$max.age[data$min.age > data$max.age], max = data$mi
n.age[data$min.age > data$max.age])

    # Samples with completed age uncertainty but max, min and interpreted age the
same
    data$age.model[data$min.age == data$max.age] <- data$interpreted.age[data$min.
age == data$max.age]
  return(data)
}
```

Test age model function by confirming that all ages in age model are within the age uncertainty for each sample, using the primary Mo subdataset as an example.

```
Mo.anox.py.rf.age.model <- Mo.anox.py.rf

Mo.anox.py.rf.age.model <- age.unc(data = Mo.anox.py.rf.age.model)

Mo.anox.py.rf.age.model <- age.model.basic(data = Mo.anox.py.rf.age.model)

within.age.bounds <- as.character()
for(row in 1:nrow(Mo.anox.py.rf.age.model)){
within.age.bounds[row] <- between(Mo.anox.py.rf.age.model$age.model[row], Mo.anox.
py.rf.age.model$min.age[row], Mo.anox.py.rf.age.model$max.age[row])
}

print("For how many samples is it true that the age model is within the age uncert
ainty bounds?")
```

```
## [1] "For how many samples is it true that the age model is within the age uncer
tainty bounds?"
```

```
summary(as.factor(within.age.bounds))
```

```
## TRUE
## 2355
```

Secondly, consider a different approach to age modeling that forces all samples to be in the correct stratigraphic order for each stratigraphic unit.

```
age.model <- function(data, iteration, dataset = "general-test", plot.strat = TRUE
```

```
, run.w.rf, stepped.threshold = 2){

### Note that we have 6 options (under 3 main headers) for our age models.

# Option 1a: Heights and depths for all samples + age structure
# Option 1b: Heights and depths for all samples + no age structure
# Option 1c: Heights and depths for all samples + stepped age structure
# Option 2a: No measured section + age structure
# Option 2b: No measured section + no age structure
# Option 2c: No measured section + stepped age structure
# Option 3a: Single sample section + measured section
# Option 3b: Single sample section + no measured section

# summarise all sections to create list
sections.sum <- data %>%
  group_by(section.name) %>%
  tally()

if(run.w.rf == FALSE){
# set core/cuttings depth to -ve so can use same script for outcrop and cuttings
data$height.depth[data$site.type == "core" | data$site.type == "cuttings"] <- -dat
a$height.depth[data$site.type == "core" | data$site.type == "cuttings"]

# Fix switched Cambrian ages.
# If relationship between min and max age is as expected (max age greater than min
age), keep columns the same.
# If relationship is wrong way around, switch them
data <- transform(data, max.age = ifelse(max.age >= min.age, max.age, min.age), mi
n.age = ifelse(max.age >= min.age, min.age, max.age))

}

for (row in 1:nrow(sections.sum)) {

    section <- sections.sum$section.name[row]
    #print(section) # print section name for troubleshooting if useful

    # generate subset of the dataframe for specific section
    section.data <- filter(data, section.name == section)

    ####################
    #### SCENARIO 1 #### [and 3a]
    ####################
    # If section has section height/depth (i.e. no NAs for section height/depth):
    if(is.na(mean(section.data$height.depth, na.rm = FALSE)) == FALSE){

    ####################
    #### SCENARIO 3a ####
    ####################
    # If there is height/depth data and only one sample...
    if(nrow(section.data) <2){
    # Basic age model method
    data$age.model[data$section.name == section] <- runif(n = nrow(filter(data, se
ction.name == section)), min = data$min.age[data$section.name == section], max = d
```

```
ata$max.age[data$section.name == section])

    # Otherwise...
    }else{

    ####################
    #### SCENARIO 1b ####
    ####################
    # If there are heights/depths entered but the ages are all the same:
    # apply approach for no obvious age structure (range is less than or equal to
1e-6, used as an approximation of zero to avoid R issues with zeros)
    if(diff(range(section.data$interpreted.age)) <= 1e-6){
    # If there are heights but all of the interpreted ages are the same, assign un
iform age model between randomly assigned minimum and maximum age with principle o
f superposition

    min.height_depth.sample <- section.data[which.min(section.data$height.depth),]
    max.height_depth.sample <- section.data[which.max(section.data$height.depth),]

    min.section.age <- runif(1, min = max.height_depth.sample$min.age, max = max.h
eight_depth.sample$max.age)

    if(min.height_depth.sample$min.age > min.section.age){
    max.section.age <- runif(1, min = min.height_depth.sample$min.age, max = min.h
eight_depth.sample$max.age)
    }else{
    max.section.age <- runif(1, min = min.section.age, max = min.height_depth.samp
le$max.age)
    }

    modeled.section.duration <- max.section.age - min.section.age
    min.section.height.depth <- min.height_depth.sample$height.depth
    total.section.height.depth <- max.height_depth.sample$height.depth  - min.heig
ht_depth.sample$height.depth

    data$min.section.age[data$section.name == section] <- min.section.age
    data$max.section.age[data$section.name == section] <- max.section.age

    data$min.section.height.depth[data$section.name == section] <- min.section.hei
ght.depth
    data$total.section.height.depth[data$section.name == section] <- total.section
.height.depth
    data$modeled.section.duration[data$section.name == section] <- modeled.section
.duration

    data$age.model[data$section.name == section] <- data$max.section.age[data$sect
ion.name == section] - ((data$height.depth[data$section.name == section] - data$mi
n.section.height.depth[data$section.name == section])/data$total.section.height.de
pth[data$section.name == section])*data$modeled.section.duration[data$section.name
== section]

    #print("Troubleshooting: Option 1b")

    # End of actively editing loop... 20230313 14.46
```

```
    #### SCENARIOS 1a+c ####
    }else{

    # if interpreted section age ascends with height/depth, flip section
    sign.heights.depths <- mean(sign(diff(section.data$height.depth)), na.rm =TRUE
)
    sign.interpreted.ages <- mean(sign(diff(section.data$interpreted.age)), na.rm
=TRUE)
    # We anticipate ages to decrease as height increases, therefore signs (-1 or 1
) of heights/depths and intepreted ages should be opposite
          # therefore the product of the two signs should always be negative (-1)
    # if they are not, then make heights/depths negative. This may not be an accur
ate representation of the geological section but that should not matter, this is s
imply for the age model calculations which will now proceed in the correct directi
on.
    # note that we are testing for the majority of the section younging up here.
    if(sign.heights.depths*sign.interpreted.ages > -0.999){
        # check flipping will work  and the section isn't just a total mess. If it
will work (and flipping will make sign very nearly 1), flip
    if(sign.heights.depths*sign.interpreted.ages > +0.999){
    section.data$height.depth <- -section.data$height.depth
    # also change in dataframe for plotting
    data$height.depth[data$section.name == section] <- -data$height.depth[data$sec
tion.name == section]
    }
    }
# now, if ascending trend is still a total mess, do random sample between age unce
rtainty bounds. Need to check signs agan to do this.
    sign.heights.depths <- mean(sign(diff(section.data$height.depth)), na.rm =TRUE
)
    sign.interpreted.ages <- mean(sign(diff(section.data$interpreted.age)), na.rm
=TRUE)
    if(sign.heights.depths*sign.interpreted.ages > -0.999){
          data$age.model[data$section.name == section] <- runif(n = nrow(filter(da
ta, section.name == section)), min = data$min.age[data$section.name == section], m
ax = data$max.age[data$section.name == section])

    }else{
      #######
    # check if interpretted age model is unnaturally stepped
    # (i.e. multiple groups with all the same age - similar to scenario above)

    age.group.sum <- section.data %>%
      group_by(interpreted.age) %>%
      tally()

    # are all samples in this section listed in interpreted age chunks?
    # RGS note - return to stepped age criteria?

    ######################
    #### SCENARIO 1c ####
    ######################
    if(nrow(age.group.sum) <= (nrow(section.data)/stepped.threshold)){
```

```
    # compute age groups - do in reverse order so still go in age order up section
    age.groups <- age.group.sum$interpreted.age[rev(order(age.group.sum$interprete
d.age))]

#    if(mean(age.group.sum$n) > 1){ removed  for code development
    for(age.group.no in 1:length(age.group.sum$n)){

    #print(age.group.no)

    age.group.interpreted.age <- as.numeric(age.groups[age.group.no])

    if(age.group.no == 1){

    age.group.data <- filter(section.data, interpreted.age == age.group.interprete
d.age)

     if(nrow(age.group.data) < 2){
    # Basic age model method
      data$age.model[data$section.name == section & data$interpreted.age == age.gr
oup.interpreted.age] <-
    runif(1, min = data$min.age[data$section.name == section & data$interpreted.ag
e == age.group.interpreted.age], max = data$max.age[data$section.name == section &
data$interpreted.age == age.group.interpreted.age])
     }else{

    min.height_depth.sample <- age.group.data[which.min(age.group.data$height.dept
h),]
    max.height_depth.sample <- age.group.data[which.max(age.group.data$height.dept
h),]

    # note that keep "section" in names rather than "age group" right now
    min.section.age <- runif(1, min = max.height_depth.sample$min.age, max = max.h
eight_depth.sample$max.age)

    if(min.height_depth.sample$min.age > min.section.age){
    max.section.age <- runif(1, min = min.height_depth.sample$min.age, max = min.h
eight_depth.sample$max.age)
    }else{
    max.section.age <- runif(1, min = min.section.age, max = min.height_depth.samp
le$max.age)
    }

    modeled.section.duration <- max.section.age - min.section.age
    min.section.height.depth <- min.height_depth.sample$height.depth
    total.section.height.depth <- max.height_depth.sample$height.depth  - min.heig
ht_depth.sample$height.depth

    data$min.section.age[data$section.name == section & data$interpreted.age == ag
e.group.interpreted.age] <- min.section.age
    data$max.section.age[data$section.name == section & data$interpreted.age == ag
e.group.interpreted.age] <- max.section.age
```

```
    data$min.section.height.depth[data$section.name == section & data$interpreted.
age == age.group.interpreted.age] <- min.section.height.depth
    data$total.section.height.depth[data$section.name == section & data$interprete
d.age == age.group.interpreted.age] <- total.section.height.depth
    data$modeled.section.duration[data$section.name == section & data$interpreted.
age == age.group.interpreted.age] <- modeled.section.duration

    data$age.model[data$section.name == section & data$interpreted.age == age.grou
p.interpreted.age] <- data$max.section.age[data$section.name == section & data$int
erpreted.age == age.group.interpreted.age] - ((data$height.depth[data$section.name
== section & data$interpreted.age == age.group.interpreted.age] - data$min.section
.height.depth[data$section.name == section & data$interpreted.age == age.group.int
erpreted.age])/data$total.section.height.depth[data$section.name == section & data
$interpreted.age == age.group.interpreted.age])*data$modeled.section.duration[data
$section.name == section & data$interpreted.age == age.group.interpreted.age]
     }
    }else{
    age.group.below <- as.numeric(age.groups[age.group.no-1])

    min.model.age.age.group.below <- min(data$age.model[data$section.name == secti
on & data$interpreted.age == age.group.below])

    age.group.data <- filter(section.data, interpreted.age == age.group.interprete
d.age)

    if(nrow(age.group.data) < 2){
    # if only one sample in step then randomly select age
    # Basic age model method - but checking min max age
      data$age.model[data$section.name == section & data$interpreted.age == age.gr
oup.interpreted.age] <-
    runif(1, min = data$min.age[data$section.name == section & data$interpreted.ag
e == age.group.interpreted.age], max = min(c(min.model.age.age.group.below, data$m
ax.age[data$section.name == section & data$interpreted.age == age.group.interprete
d.age])))
         }else{

    min.height_depth.sample <- age.group.data[which.min(age.group.data$height.dept
h),]
    max.height_depth.sample <- age.group.data[which.max(age.group.data$height.dept
h),]

    max.section.age <-  runif(1, min =  min.height_depth.sample$min.age, max = min
(c(min.model.age.age.group.below, min.height_depth.sample$max.age)))

    # note that keep "section" in names rather than "age group" right now
   min.section.age <- runif(1, min = max.height_depth.sample$min.age, max = max.se
ction.age)



    # no need for this if loop used in non-chunked age models as it is always the
case for age groups that the min.section.age will be younger than the minimum age
of the sample at the stratigraphic top of the chunk (?) double check!
#    if(min.height_depth.sample$min.age > min.section.age){
```

```
#    max.section.age <- runif(1, min = min.height_depth.sample$min.age, max = min.
height_depth.sample$max.age)
#    }else{
#    max.section.age <- runif(1, min = min.section.age, max = min.height_depth.sam
ple$max.age)
#    }

    modeled.section.duration <- max.section.age - min.section.age
    min.section.height.depth <- min.height_depth.sample$height.depth
    total.section.height.depth <- max.height_depth.sample$height.depth  - min.heig
ht_depth.sample$height.depth

    data$min.section.age[data$section.name == section & data$interpreted.age == ag
e.group.interpreted.age] <- min.section.age
    data$max.section.age[data$section.name == section & data$interpreted.age == ag
e.group.interpreted.age] <- max.section.age

    data$min.section.height.depth[data$section.name == section & data$interpreted.
age == age.group.interpreted.age] <- min.section.height.depth
    data$total.section.height.depth[data$section.name == section & data$interprete
d.age == age.group.interpreted.age] <- total.section.height.depth
    data$modeled.section.duration[data$section.name == section & data$interpreted.
age == age.group.interpreted.age] <- modeled.section.duration

    data$age.model[data$section.name == section & data$interpreted.age == age.grou
p.interpreted.age] <- data$max.section.age[data$section.name == section & data$int
erpreted.age == age.group.interpreted.age] - ((data$height.depth[data$section.name
== section & data$interpreted.age == age.group.interpreted.age] - data$min.section
.height.depth[data$section.name == section & data$interpreted.age == age.group.int
erpreted.age])/data$total.section.height.depth[data$section.name == section & data
$interpreted.age == age.group.interpreted.age])*data$modeled.section.duration[data
$section.name == section & data$interpreted.age == age.group.interpreted.age]
     }
    }
    #print("Troubleshooting: Option 1c")

    }
    }else{
    ######################
    #### SCENARIO 1a ####
    ######################
    min.height_depth.sample <- section.data[which.min(section.data$height.depth),]
    max.height_depth.sample <- section.data[which.max(section.data$height.depth),]

    # assume section has no unconformities (?)
    min.section.interpreted.age <- max.height_depth.sample$interpreted.age
    max.section.interpreted.age <- min.height_depth.sample$interpreted.age

    interpreted.section.duration <- max.section.interpreted.age - min.section.inte
rpreted.age

    ### THIS NOW SHOULDNT BE NECESSARY??? 20230802
    # If section has been entered wrong and heights/depths do not decrease with ag
e
```

```
    # in this scenario min.height_depth.sample$interpreted.age <= max.height_depth
.sample$interpreted.age
    # default to just random ages.
    # NOTE - alternatively could assume height numbers should be inverted as depth
s.
    if(min.height_depth.sample$interpreted.age <= max.height_depth.sample$interpre
ted.age){
    # Default method
    data$age.model[data$section.name == section] <- runif(n = nrow(filter(data, se
ction.name == section)), min = data$min.age[data$section.name == section], max = d
ata$max.age[data$section.name == section])

    # Alternative would remove the "else" statement below and set height/depth val
ues to be negative
    }else{

    min.section.age <- runif(1, min = max.height_depth.sample$min.age, max = max.h
eight_depth.sample$max.age)

    # a couple of options...
    # 1 - go with interpreted section duration relative to min age
    # 2 - go with random min and max. If min age of max is lower than random age o
f min, use random age of min as min age of max

    ### OPTION 1 ###
    #max.section.age <- min.section.age + abs(min.height_depth.sample$interpreted.
age - max.height_depth.sample$interpreted.age)

    ### OPTION 2 ###

    if(min.height_depth.sample$min.age > min.section.age){
    max.section.age <- runif(1, min = min.height_depth.sample$min.age, max = min.h
eight_depth.sample$max.age)
    }else{
    max.section.age <- runif(1, min = min.section.age, max = min.height_depth.samp
le$max.age)
    }

    modeled.section.duration <- max.section.age - min.section.age

    data$min.section.age[data$section.name == section] <- min.section.age
    data$max.section.age[data$section.name == section] <- max.section.age

    data$modeled.section.duration[data$section.name == section] <- modeled.section
.duration

    data$interpreted.section.duration[data$section.name == section] <- interpreted
.section.duration
    data$min.section.interpreted.age[data$section.name == section] <- min.section.
interpreted.age

    #data$min.section.height[data$section.name == section] <- min.section.height
    #data$max.section.height[data$section.name == section] <- max.section.height
```

```
    # Used abs() here but now cores are negative should always be positive?
    #data$measured.section.height[data$section.name == section] <- abs(max.section
.height - min.section.height)


    data$age.model[data$section.name == section] <- data$min.section.age[data$sect
ion.name == section] + ((data$interpreted.age[data$section.name == section] - data
$min.section.interpreted.age[data$section.name == section])/data$interpreted.secti
on.duration[data$section.name == section])*data$modeled.section.duration[data$sect
ion.name == section]

    #print("Troubleshooting: Option 1a")

    }
    }
    }
    }

      }
# make plots for sections with height/depth data
if(plot.strat == TRUE){
    section.data.for.plot <- filter(data, section.name == section)

    section <- sub("/", "-", section)
    section <- sub(":", "-", section)

      plot <- ggplot()+
        geom_errorbarh(data = section.data.for.plot, aes(y = height.depth, xmin =
min.age, xmax=max.age), colour = 'darkred', height = .5)+
        geom_point(data = section.data.for.plot, aes(y = height.depth, x = age.mod
el), shape=21, fill="grey70", size=6, alpha=0.8)+
        theme_bw()+
        scale_x_reverse()+
        ylab(expression("Height/Depth"))+xlab(expression("Age Model (Ma)"))+
        theme(plot.margin = ggplot2::margin(1,1,1,1,"cm"),panel.border = element_r
ect(fill=NA,color="black", linewidth=2,linetype="solid"),
              axis.line = element_line(lineend = 'square'),
              axis.ticks = element_line(linewidth=1),
              axis.title = element_text(size=34),
              axis.text = element_text( size=26, color="black"),
              legend.title = element_text(size=24),
              legend.text = element_text( size=18),
              axis.ticks.length = unit(5, "points"),
              legend.justification=c(1,1), legend.position=c(.98,.36),
              panel.grid.major = element_blank(),panel.grid.minor = element_blank(
))

    dataset.file.name <- sub("..ppm.|..wt..", "", dataset)
    mainDir <- getwd()
    subDir <- paste("strat.model.plots/", dataset.file.name, sep = "")
    subsubDir <- paste("/iteration-", iteration, sep = "")

    dir.create(file.path(mainDir, subDir), showWarnings = FALSE)
    dir.create(file.path(mainDir, subDir, subsubDir), showWarnings = FALSE)
```

```
    setwd(file.path(mainDir, subDir, subsubDir))

    ggsave(paste0(section ,".pdf"), plot, width=9.3, height=8.5, units="in")

    setwd(file.path(mainDir))
}

}

    ####################
    #### SCENARIO 2 #### [and 3b]
    ####################
    # If section does not have section height/depth (i.e. at least some NAs for se
ction height/depth):
    if(is.na(mean(section.data$height.depth, na.rm = FALSE)) == TRUE){

    ######################
    #### SCENARIO 3b ####
    ######################
    # If there is only one sample...
    if(nrow(section.data) <2){
    # Basic age model method
    data$age.model[data$section.name == section] <- runif(n = nrow(filter(data, se
ction.name == section)), min = data$min.age[data$section.name == section], max = d
ata$max.age[data$section.name == section])

    #print("Troubleshooting: Option 3b")

    # Otherwise...
    }else{
    #### SCENARIO 2 #### [and 3b]

    if(diff(range(section.data$interpreted.age)) <= 1e-6){
    ######################
    #### SCENARIO 2b ####
    ######################
    # no obvious age structure (range is less than approx zero, 1e-6 used to avoid
R issues with zeros)
    # Default age model method
    data$age.model[data$section.name == section] <- runif(n = nrow(filter(data, se
ction.name == section)), min = data$min.age[data$section.name == section], max = d
ata$max.age[data$section.name == section])
      #print("Troubleshooting: Option 2b")
    } else{
    # check if interpretted age model is unnaturally stepped
    # (i.e. multiple groups with all the same age - similar to scenario above)

    age.group.sum <- section.data %>%
      group_by(interpreted.age) %>%
      tally()

      if(nrow(age.group.sum) <= (nrow(section.data)/stepped.threshold)){
      ######################
      #### SCENARIO 2c ####
```

```
      ####################


    # compute age groups - do in reverse order so still go in age order up section
    age.groups <- age.group.sum$interpreted.age[rev(order(age.group.sum$interprete
d.age))]

#    if(mean(age.group.sum$n) > 1){ removed  for code development
    for(age.group.no in 1:length(age.group.sum$n)){

    #print(age.group.no)

    age.group.interpreted.age <- as.numeric(age.groups[age.group.no])

    if(age.group.no == 1){

    age.group.data <- filter(section.data, interpreted.age == age.group.interprete
d.age)

    # Basic age model method
      data$age.model[data$section.name == section & data$interpreted.age == age.gr
oup.interpreted.age] <-
    runif(nrow(age.group.data), min = data$min.age[data$section.name == section &
data$interpreted.age == age.group.interpreted.age], max = data$max.age[data$sectio
n.name == section & data$interpreted.age == age.group.interpreted.age])

    }else{
    age.group.below <- as.numeric(age.groups[age.group.no-1])

    min.model.age.age.group.below <- min(data$age.model[data$section.name == secti
on & data$interpreted.age == age.group.below])

    age.group.data <- filter(section.data, interpreted.age == age.group.interprete
d.age)

    # Basic age model method, but with min age from group below
    data$age.model[data$section.name == section & data$interpreted.age == age.grou
p.interpreted.age] <-
    runif(nrow(age.group.data), min = data$min.age[data$section.name == section &
data$interpreted.age == age.group.interpreted.age], max = min.model.age.age.group.
below)

    }

    }

        #print(section)

        }else{
        ####################
        #### SCENARIO 2a ####
        ####################

    min.age.sample <- section.data[which.min(section.data$interpreted.age),]
```

```
    max.age.sample <- section.data[which.max(section.data$interpreted.age),]

    min.section.interpreted.age <- min.age.sample$interpreted.age
    max.section.interpreted.age <- max.age.sample$interpreted.age

    interpreted.section.duration <- max.section.interpreted.age - min.section.inte
rpreted.age

    min.section.age <- runif(1, min = min.age.sample$min.age, max = min.age.sample
$max.age)

        # a couple of options...
    # 1 - go with interpreted section duration relative to min age
    # 2 - go with random min and max. If min age of max is lower than random age o
f min, use random age of min as min age of max

    ### OPTION 1 ###
    #max.section.age <- min.section.age + abs(max.age.sample$interpreted.age - min
.age.sample$interpreted.age)

    ### OPTION 2 ###

    if(max.age.sample$min.age > min.section.age){
    max.section.age <- runif(1, min = max.age.sample$min.age, max = max.age.sample
$max.age)
    }else{
    max.section.age <- runif(1, min = min.section.age, max = max.age.sample$max.ag
e)
    }

    modeled.section.duration <- max.section.age - min.section.age

    data$min.section.age[data$section.name == section] <- min.section.age
    data$max.section.age[data$section.name == section] <- max.section.age

    # Note that in this version, modeled and interpreted ages are different. In Op
tion 1 version of age model, modeled and interpreted ages are the same value.
    data$modeled.section.duration[data$section.name == section] <- modeled.section
.duration
    data$interpreted.section.duration[data$section.name == section] <- interpreted
.section.duration
    data$min.section.interpreted.age[data$section.name == section] <- min.section.
interpreted.age

    data$age.model[data$section.name == section] <- data$min.section.age[data$sect
ion.name == section] + ((data$interpreted.age[data$section.name == section] - data
$min.section.interpreted.age[data$section.name == section])/data$interpreted.secti
on.duration[data$section.name == section])*data$modeled.section.duration[data$sect
ion.name == section]

    #print("Troubleshooting: Option 2a")
        }
    }
    }
```

```
    if(plot.strat == TRUE){
    section.data.for.plot <- filter(data, section.name == section)

    section <- sub("/", "-", section)
    section <- sub(":", "-", section)

plot <- ggplot()+
        geom_errorbarh(data = section.data.for.plot, aes(y = interpreted.age, xmin
= min.age, xmax=max.age), colour = 'darkred', height = .5)+
        geom_point(data = section.data.for.plot, aes(y = interpreted.age, x = age.
model), shape=21, fill="grey70", size=6, alpha=0.8)+
        theme_bw()+
        scale_y_reverse()+scale_x_reverse()+
        ylab(expression("Interpreted Age (Ma)"))+xlab(expression("Age Model (Ma)")
)+
        theme(plot.margin = ggplot2::margin(1,1,1,1,"cm"),panel.border = element_r
ect(fill=NA,color="black", linewidth=2,linetype="solid"),
              axis.line = element_line(lineend = 'square'),
              axis.ticks = element_line(linewidth=1),
              axis.title = element_text(size=34),
              axis.text = element_text( size=26, color="black"),
              legend.title = element_text(size=24),
              legend.text = element_text( size=18),
              axis.ticks.length = unit(5, "points"),
              legend.justification=c(1,1), legend.position=c(.98,.36),
              panel.grid.major = element_blank(),panel.grid.minor = element_blank(
))

    dataset.file.name <- sub("..ppm.|..wt..", "", dataset)
    mainDir <- getwd()
    subDir <- paste("strat.model.plots/", dataset.file.name, sep = "")
    subsubDir <- paste("/iteration-", iteration, sep = "")

    dir.create(file.path(mainDir, subDir), showWarnings = FALSE)
    dir.create(file.path(mainDir, subDir, subsubDir), showWarnings = FALSE)
    setwd(file.path(mainDir, subDir, subsubDir))

    ggsave(paste0(section ,".pdf"), plot, width=9.3, height=8.5, units="in")

    setwd(file.path(mainDir))
    }
    }




# Going to have to then check if I have covered all options.

    # Samples with completed age uncertainty but max, min and interpreted age the
same
    #data$age.model[data$min.age == data$max.age] <- data$interpreted.age[data$min
.age == data$max.age]
}
```

```
return(data)

}
```

Check age model

```
iteration <- "test"

test.data <- age.unc(Mo.anox.py.rf)

test.data.w.age.model <- age.model(test.data, iteration = iteration, dataset = "M
o.test", plot.strat = TRUE, run.w.rf = FALSE)

# Additional age model test

iteration <- "test.2"

test.data.2 <- age.unc(Fepy.anox.rf)

test.data.w.age.model.2 <- age.model(test.data.2, iteration = iteration, dataset =
"Fepy.test", plot.strat = TRUE, run.w.rf = FALSE)
```

4. Monte Carlo random forest function

We define a function to conduct n random forest model analyses (100 in main text of this study), with the aim of 1) generating partial dependence plots for the variable of interest with respect to geologic time when all other specified variables are held constant, 2) generating variable importance analyses of all other specified variables, 3) evaluating model uncertainties associated with partial context data and geologic age uncertainties.

This function calls the functions age.unc(), AI.impute() and partial.context() above if the options to run them are selected in the function itself.

```
Monte.Carlo.rF <- function(data,
                       resp.var,
                       vars,
                       n,
                       run.age.unc,
                       run.partial.context,
                       run.AI.impute,
                       plot.strat = FALSE,
                       plot.strat.sum = FALSE,
                       run.w.rf = TRUE,
                       stepped.threshold = 2){

  # initiate data frames
  partial.plot.data <-  data.frame(Age=double(), Var=double(), Iteration=double())
  importance.data <-  data.frame(Variable=double(), IncMSE=double(), IncNodePurity
=double(), Iteration=double())

  model.fit.data <-  data.frame(best.nodesize=double(), best.mtry=double(), best.n
tree=double(), best.MSE=double(), best.var=double())
```

```
  # NOTE - moved this up on 20230222 - assume only have to do this once?
  if(run.age.unc == TRUE){
    data <- age.unc(data)
  }

    # set core/cuttings depth to -ve so can use same script for outcrop and cuttin
gs - *used to be in age model*
data$height.depth[data$site.type == "core" | data$site.type == "cuttings"] <- -dat
a$height.depth[data$site.type == "core" | data$site.type == "cuttings"]

# Fix Cambrian ages muck ups - *used to be in age model*
# If relationship between min and max age is as expected (max age greater than min
age), keep columns the same.
# If relationship is wrong way around, switch them
data <- transform(data, max.age = ifelse(max.age >= min.age, max.age, min.age), mi
n.age = ifelse(max.age >= min.age, min.age, max.age))


# If we're going to plot stratigraphic summary figures, need to initiate those fig
ures.
if(plot.strat.sum == TRUE){

  # summarise all sections to create list
  sections.sum <- data %>%
  group_by(section.name) %>%
  tally()

  dataset <- resp.var

  for (row in 1:nrow(sections.sum)){

  # select individual section (looped)
  section <- sections.sum$section.name[row]

  section.data.for.plot <- filter(data, section.name == section)

  # Eliminate symbols that can't be saved in section plot file names (for plotting
)
  section <- sub("/", "-", section)
  section <- sub(":", "-", section)

  # Does section have height/depth data? If so (i.e. if is.na()=FALSE), initiate p
lot with age model on x axis and height/depth on y axis
  if(is.na(mean(section.data.for.plot$height.depth, na.rm = FALSE)) == FALSE){

    # check if height/depths need flipping. Skip sections with one sample.
    if(nrow(section.data.for.plot) >1){
    #print("Step 2")
    # if interpreted section age ascends with height/depth, flip section
    sign.heights.depths <- mean(sign(diff(section.data.for.plot$height.depth)), na
.rm =TRUE)
    sign.interpreted.ages <- mean(sign(diff(section.data.for.plot$interpreted.age)
```

```
), na.rm =TRUE)
    # We anticipate ages to decrease as height increases, therefore signs (-1 or 1
) of heights/depths and intepreted ages should be opposite
    # therefore the product of the two signs should always be negative (-1)
    # if they are not, then make heights/depths negative. This may not be an accur
ate representation of the geological section but that should not matter, this is s
imply for the age model calculations which will now proceed in the correct directi
on.
    # note that we are testing for the majority of the section younging up here.
    if(sign.heights.depths*sign.interpreted.ages > -0.999){
    # check flipping will work  and the section isn't just a total mess. If it wil
l work (and flipping will make sign very nearly 1), flip
    if(sign.heights.depths*sign.interpreted.ages > +0.999){
    section.data.for.plot$height.depth <- -section.data.for.plot$height.depth}
    }
    }

  # In plot calls, reverse x and y and then use coord flip to make plotting age mo
dels easier.

  # make base plot
    plot <- ggplot()+
        geom_errorbar(data = section.data.for.plot, aes(x = height.depth, ymin = m
in.age, ymax=max.age), colour = 'darkred', width = .5)+
      geom_point(data = section.data.for.plot, aes(x = height.depth, y = interpret
ed.age), shape=21, fill="darkorange3", size=5, alpha=1) +
        theme_bw()+
        scale_y_reverse()+coord_flip()+
        xlab(expression("Height/Depth"))+ylab(expression("Age Model (Ma)"))+
        theme(plot.margin = ggplot2::margin(1,1,1,1,"cm"),panel.border = element_r
ect(fill=NA,color="black", linewidth=2,linetype="solid"),
              axis.line = element_line(lineend = 'square'),
              axis.ticks = element_line(linewidth=1),
              axis.title = element_text(size=34),
              axis.text = element_text( size=26, color="black"),
              legend.title = element_text(size=24),
              legend.text = element_text( size=18),
              axis.ticks.length = unit(5, "points"),
              legend.justification=c(1,1), legend.position=c(.98,.36),
              panel.grid.major = element_blank(),panel.grid.minor = element_blank(
))

    # If there is only one sample in the section, label it as such.
        if(nrow(section.data.for.plot) < 2){
    plot <- plot + labs(title = "Option 3a: \nSingle sample section + measured sec
tion")+ theme(plot.title = element_text(size=18))
    # Otherwise...
    }else{
    # Does the section have an age structure?
    # if all of the interpreted ages are essentially the same (within 1 year), the
n the section has no interpreted age structure. Then label it as such.
      if(diff(range(section.data.for.plot$interpreted.age)) <= 1e-6){
    plot <- plot + labs(title = "Option 1b: \nHeights and depths for all samples +
no age structure") + theme(plot.title = element_text(size=18))
```

```
      } else{
    # check if interpreted age model is unnaturally stepped
    # (i.e. multiple groups with all the same age - similar to scenario above)

    age.group.sum.for.plot <- section.data.for.plot %>%
      group_by(interpreted.age) %>%
      tally()

    # If this section meets our criteria for a stepped age structure, label it as
such.
    # RGS note - could revist our criteria for stepped age structure.
    if(nrow(age.group.sum.for.plot) <= (nrow(section.data.for.plot)/stepped.thresh
old)){

        plot <- plot + labs(title = "Option 1c: \nHeights and depths for all sampl
es + stepped age structure") + theme(plot.title = element_text(size=18))

    # If this section has an age structure and it is not stepped, label it as such
. Note, this is essentially our gold standard section.
        }else{
        plot <- plot + labs(title = "Option 1a: \nHeights and depths for all sampl
es + age structure") + theme(plot.title = element_text(size=18))

        }

    }
    }
  }
    # If the section does NOT have height/depth data (i.e. if is.na()=TRUE), initi
ate plot with age model on x axis and interpreted age on y axis
    # In plot calls, reverse x and y and then use coord flip to make plotting age
models easier.
    if(is.na(mean(section.data.for.plot$height.depth, na.rm = FALSE)) == TRUE){
  # make base plot
  plot <- ggplot()+
        geom_errorbar(data = section.data.for.plot, aes(x = interpreted.age, ymin
= min.age, ymax=max.age), colour = 'darkred', width = .5)+ geom_point(data = secti
on.data.for.plot, aes(x = interpreted.age, y = interpreted.age), shape=21, fill="d
arkorange3", size=5, alpha=1) +
        theme_bw()+
        scale_y_reverse()+scale_x_reverse()+coord_flip()+
        xlab(expression("Interpreted Age (Ma)"))+ylab(expression("Age Model (Ma)")
)+
        theme(plot.margin = ggplot2::margin(1,1,1,1,"cm"),panel.border = element_r
ect(fill=NA,color="black", linewidth=2,linetype="solid"),
              axis.line = element_line(lineend = 'square'),
              axis.ticks = element_line(linewidth=1),
              axis.title = element_text(size=34),
              axis.text = element_text( size=26, color="black"),
              legend.title = element_text(size=24),
              legend.text = element_text( size=18),
              axis.ticks.length = unit(5, "points"),
              legend.justification=c(1,1), legend.position=c(.98,.36),
```

```
            panel.grid.major = element_blank(),panel.grid.minor = element_blank(
))

  # if section only has one sample, label it as such.
         if(nrow(section.data.for.plot) < 2){
    plot <- plot + labs(title = "Option 3b: \nSingle sample section + no measured
section") + theme(plot.title = element_text(size=18))
    # Otherwise...
    # Does the section have an age structure?
    # if all of the interpreted ages are essentially the same (within 1 year), the
n the section has no interpreted age structure. Then label it as such.
    }else{
      if(diff(range(section.data.for.plot$interpreted.age)) <= 1e-6){
    plot <- plot + labs(title = "Option 2b: \nNo measured section + no age structu
re") + theme(plot.title = element_text(size=18))

      } else{
    # check if interpretted age model is unnaturally stepped
    # (i.e. multiple groups with all the same age - similar to scenario above)

    age.group.sum.for.plot <- section.data.for.plot %>%
      group_by(interpreted.age) %>%
      tally()

    # If this section meets our criteria for a stepped age structure, label it as
such.
    # RGS note - could revist our criteria for stepped age structure.
    if(nrow(age.group.sum.for.plot) <= (nrow(section.data.for.plot)/stepped.thresh
old)){
        plot <- plot + labs(title = "Option 2c: \nNo measured section + stepped ag
e structure") + theme(plot.title = element_text(size=18))

        }else{
        # If this section has an age structure and it is not stepped, label it as
such.
        plot <- plot + labs(title = "Option 2a: \nNo measured section + age struct
ure") + theme(plot.title = element_text(size=18))

        }
      }

    }

    }

    assign(paste0("section-", section), plot)

    dataset.file.name <- sub("..ppm.|..wt..", "", dataset)
    mainDir <- getwd()
    subDir <- paste("strat.model.plots/", dataset.file.name, sep = "")
    subsubDir <- paste("/stratigraphic-model-summary", sep = "")

    dir.create(file.path(mainDir, subDir), showWarnings = FALSE)
    dir.create(file.path(mainDir, subDir, subsubDir), showWarnings = FALSE)
```

```
    setwd(file.path(mainDir, subDir, subsubDir))

    ggsave(paste0(section ,".pdf"), get(paste0("section-", section)), width=9.3, h
eight=8.5, units="in")

    setwd(file.path(mainDir))
  }
  }


  ###### START OF KEY LOOP ######
  # initiate loop

  # setting up a back end...
# how many cores do we have??

parallel::detectCores()

n.cores <- c((parallel::detectCores() - 1), n)[which.min(c((parallel::detectCores(
) - 1), n))]


#create the cluster
# NOTE - for final version should have options...
# e.g. - if local.unix == TRUE then type = "FORK", else type = "PSOCK"
my.cluster <- parallel::makeCluster(
  n.cores,
  type = "PSOCK" # PSOCK for iridis5  and RGS MacBook
  #type = "FORK" # FORK for ubuntu machines?
)

#check cluster definition (optional)
print(my.cluster)

#register it to be used by %dopar%
doParallel::registerDoParallel(cl = my.cluster)

#check if it is registered (optional)
foreach::getDoParRegistered()

#how many workers are available? (optional)
foreach::getDoParWorkers()

rf.sum.par <- foreach(
      iteration = 1:n,
      .combine = 'c',
      .packages = c("randomForest", "dplyr", "ggplot2")
) %dopar% {

source("SGP-functions-unannotated.R")

  #for (iteration in 1:n){

  # make new data frame for iteration.
```

```
  data_it <- data

  if(run.Al.impute == TRUE){
    data_it <- Al.impute(data_it)
  }

  # Note that Al.impute should be run before partial.context so that samples witho
ut assigned lithology have Al imputed based on all identified samples (not their r
andom assignment in this iteration).
  if(run.partial.context == TRUE){
  data_it <- partial.context(data_it,
                  site.type = TRUE,
                  metamorphic.bin = TRUE,
                  basin.type = TRUE,
                  site.latitude = TRUE,
                  site.longitude = TRUE,
                  environmental.bin = TRUE,
                  lithology.name = TRUE)
  }

  # Run age model for this iteration
  data_it <- age.model(data_it, iteration = iteration, dataset = resp.var, plot.st
rat = plot.strat, run.w.rf = run.w.rf)

  # if plotting stratigraphic summary plots, need to add the age models to our sum
mary plots for each iteration.
  if(plot.strat.sum == TRUE){

  # summarise all sections to create list
  sections.sum <- data %>%
  group_by(section.name) %>%
  tally()

  dataset <- resp.var

  # plot section by section
  for (row in 1:nrow(sections.sum)){

  section <- sections.sum$section.name[row]

  section.data.for.plot <- filter(data_it, section.name == section)
  section <- sub("/", "-", section)
  section <- sub(":", "-", section)

    # for sections with height/depths
    # i.e. Scenarios 1a, 1b, 1c, 3a
    if(is.na(mean(section.data.for.plot$height.depth, na.rm = FALSE)) == FALSE){

    add.on.path <- geom_path(data = section.data.for.plot, aes(x = height.depth, y
= age.model), color="grey5", size=.8, alpha=0.2)
    add.on.point <- geom_point(data = section.data.for.plot, aes(x = height.depth,
y = age.model), shape=15, color="grey40", size=3, alpha=0.2)
    }
    # for sections with no height/depths but interpreted ages
```

```
    # i.e. Scenarios 2a, 2b, 2c, 3b
    if(is.na(mean(section.data.for.plot$height.depth, na.rm = FALSE)) == TRUE){
    add.on.path <- geom_path(data = section.data.for.plot, aes(x = interpreted.age
, y = age.model), shape=21, color="grey5", size=.8, alpha=0.2)
    add.on.point <- geom_point(data = section.data.for.plot, aes(x = interpreted.a
ge, y = age.model), shape=15, color="grey40", size=3, alpha=0.2)
    }

    assign(paste0("section-", section), get(paste0("section-", section)) + add.on.
path + add.on.point)

    dataset.file.name <- sub("..ppm.|..wt..", "", dataset)
    mainDir <- getwd()
    subDir <- paste("strat.model.plots/", dataset.file.name, sep = "")
    subsubDir <- paste("/stratigraphic-model-summary", sep = "")

    dir.create(file.path(mainDir, subDir), showWarnings = FALSE)
    dir.create(file.path(mainDir, subDir, subsubDir), showWarnings = FALSE)
    setwd(file.path(mainDir, subDir, subsubDir))

    ggsave(paste0(section ,".pdf"), get(paste0("section-", section)), width=9.3, h
eight=8.5, units="in")

    setwd(file.path(mainDir))
  }
  }

  # select all of the variables named in the "vars" vector to be columns in the da
taframe data.for.rF
  data.for.rF <- select(data_it, all_of(vars))

  # remove any rows that still contain NAs
  data.for.rF <- na.omit(data.for.rF)

  # define a formula that identifies the response variable defined in function and
uses all other vars as model variables.
  formula <- as.formula(paste(resp.var, "~ ."))

  # Random forest function parameter notes
  # ntree - Number of trees to grow. Default is 500 (used here)
  # mtry=3 - Number of variables randomly sampled as candidates at each split. The
default values are different for classification (sqrt(p) where p is number of vari
ables in x) and regression (p/3). p/3 is therefore what is used here.
  # importance=TRUE - used for generating variable importance plots.
  # proximity=TRUE - used for generating proximity plots (we never do this in this
study).
  # na.action=na.omit - A function to specify the action to be taken if NAs are fo
und. (NOTE: If given, this argument must be named.)
  # Note that replace = TRUE by default.

  ########## ADDING IN HYPERPARAMETER OPTIMISATION ###########
  ### Trying a simple loop for grid search for hyperparameter optimization

train.Mo <- sample(1:nrow(data.for.rF), (nrow(data.for.rF)/3)*2) # # this time do
```

```
it w 2/3... use 4/5 of the data for training
Mo.test <- data.for.rF[-train.Mo, resp.var]

#summary(tree.Mo)

ntree.vec <- 2^seq(0, 8, 1) # this is a reasonable range but check MSE is plateuin
g w increased ntree by the end
mtry.vec <- seq(2, 10, 2)
nodesize.vec.too.large <- 2^seq(1, 20, 1)
nodesize.vec <- nodesize.vec.too.large[nodesize.vec.too.large < (nrow(data.for.rF)
/4)] # essentially saying we want at least 4 nodes?

MSE.sum <- c()

i <- 0

for(ntree.no in ntree.vec){
  for(mtry.no in mtry.vec){
    for(nodesize.no in nodesize.vec){
      i <- i + 1

      rf.Mo <- randomForest(formula, data = data.for.rF, subset = train.Mo, mtry =
mtry.no, ntree=ntree.no, nodesize = nodesize.no, importance = TRUE)

      rf.Mo

      yhat.Mo.rf <- predict(rf.Mo, newdata = data.for.rF[-train.Mo, ])
      plot(yhat.Mo.rf, Mo.test)
      abline(0, 1)

      MSE <- mean((yhat.Mo.rf - Mo.test)^2)
      exp.var <- round(100 * rf.Mo$rsq[length(rf.Mo$rsq)], digits = 2)

      MSE.sum$ntree.no[i] <- ntree.no
      MSE.sum$mtry.no[i] <- mtry.no
      MSE.sum$nodesize.no[i] <- nodesize.no
      MSE.sum$MSE[i] <- MSE
      MSE.sum$exp.var[i] <- exp.var
    }

  }
}

MSE.sum <- as.data.frame(MSE.sum)

best.row <- which.min(MSE.sum$MSE)

MSE.sum.best.row <- MSE.sum[best.row,]

best.nodesize <- MSE.sum$nodesize.no[best.row]
best.mtry <- MSE.sum$mtry.no[best.row]
best.ntree <- MSE.sum$ntree.no[best.row]

best.MSE <- MSE.sum$MSE[best.row]
```

```
best.var <- MSE.sum$exp.var[best.row]

model.fit.row <- cbind(best.nodesize,
             best.mtry,
             best.ntree,
             best.MSE,
             best.var
             )

model.fit.data[1,] <- model.fit.row

ggplot(MSE.sum, aes(y = MSE, x = 1:nrow(MSE.sum)))+
  geom_point(aes(colour = nodesize.no, size = mtry.no, alpha = ntree.no))+
  annotate(geom="point", y = MSE.sum.best.row$MSE, x = best.row, colour = "red")+
  theme_bw()+
  scale_color_viridis_c()

dataset <- resp.var
    dataset.file.name <- sub("..ppm.|..wt..", "", dataset)
    mainDir <- getwd()
    subDir <- paste("hyperparameter.plots/", dataset.file.name, sep = "")

    dir.create(file.path(mainDir, subDir), showWarnings = FALSE)
    setwd(file.path(mainDir, subDir))

    ggsave(paste0("hyperparameter-validation-", iteration ,".pdf"), width=10, heig
ht=8.5, units="in")

    ggplot(MSE.sum, aes(y = exp.var, x = 1:nrow(MSE.sum)))+
  geom_point(aes(colour = nodesize.no, size = mtry.no, alpha = ntree.no))+
  annotate(geom="point", y = best.var, x = best.row, colour = "red")+
  theme_bw()+
  scale_color_viridis_c()

    ggsave(paste0("variance-hyperparameter-validation-", iteration ,".pdf"), width
=10, height=8.5, units="in")

    setwd(file.path(mainDir))


  # suggested new equation model code:
  it.rF <- randomForest(formula, data.for.rF, na.action=na.omit, subset = train.Mo
, mtry = best.mtry, ntree = best.ntree, nodesize = best.nodesize, importance = TRU
E)

  it.partial.plot <- as.data.frame(partialPlot(it.rF, data.for.rF, age.model))

  it.import <- as.data.frame(importance(it.rF))
  it.import <- tibble::rownames_to_column(it.import, "Variable")

  # NOTE - we do not use rbind() to combine random forest iterations in order to m
ake this function more efficient. This slightly reduces readability but increases
efficiency dramatically for high n values.
  #partial.i <- nrow(partial.plot.data)+1
```

```
  #partial.j <- nrow(partial.plot.data)+nrow(it.partial.plot)
  #partial.plot.data[partial.i:partial.j, 1:2] <- it.partial.plot
  #partial.plot.data[partial.i:partial.j, 3] <- rep(iteration, nrow(it.partial.plo
t))

  # rbind version
  names(it.partial.plot) <- c("Age", "Var")
  it.partial.plot$Iteration <- rep(iteration, nrow(it.partial.plot))

  #partial.plot.data <- rbind(partial.plot.data, it.partial.plot)
  partial.plot.data <- it.partial.plot


  #import.i <- nrow(importance.data)+1
  #import.j <- nrow(importance.data)+nrow(it.import)
  #importance.data[import.i:import.j, 1:3] <- it.import
  #importance.data[import.i:import.j, 4] <- rep(iteration, nrow(it.import))

  # rbind version
  names(it.partial.plot) <- c("Variable", "IncMSE", "IncNodePurity")
  it.import$Iteration <- rep(iteration, nrow(it.import))

  #importance.data <- rbind(importance.data, it.import)
  importance.data <- it.import

  # Name variables in variable importance data nicely for plotting etc.
  importance.data$Variable[importance.data$Variable == "age.model"] <- "Age model"
  importance.data$Variable[importance.data$Variable == "site.type"] <- "Site type"
  importance.data$Variable[importance.data$Variable == "metamorphic.bin"] <- "Meta
morphic bin"
  importance.data$Variable[importance.data$Variable == "basin.type"] <- "Basin typ
e"
  importance.data$Variable[importance.data$Variable == "site.latitude"] <- "Latitu
de"
  importance.data$Variable[importance.data$Variable == "site.longitude"] <- "Longi
tude"
  importance.data$Variable[importance.data$Variable == "lithology.name"] <- "Litho
logy"
  importance.data$Variable[importance.data$Variable == "environmental.bin"] <- "En
vironmental bin"
  if(("TOC..wt.." %in% vars == TRUE) & (resp.var != "TOC..wt..")){
  importance.data$Variable[importance.data$Variable == "TOC..wt.."] <- "TOC"
  }
  if(("Fe.py.FeHR" %in% vars == TRUE) & (resp.var != "Fe.py.FeHR")){
  importance.data$Variable[importance.data$Variable == "Fe.py.FeHR"] <- "Fepy/FeHR
"
  }
  if(("Al..wt.." %in% vars == TRUE) & (resp.var != "Al..wt..")){
  importance.data$Variable[importance.data$Variable == "Al..wt.."] <- "Al"
  }

  rf.sum.it <- list(partial.plot.data, importance.data, model.fit.data)
  names(rf.sum.it) <- c("partial.plot.data", "importance.data", "model.fit.data")
  return(rf.sum.it)
```

```
  print(paste("Finished iteration", iteration))
  }
  #rf.sum <- list(partial.plot.data, importance.data)
  #names(rf.sum) <- c("partial.plot.data", "importance.data")

  parallel::stopCluster(cl = my.cluster)

# initiate frames
partial.plot.data <-  data.frame(Age=double(), Var=double(), Iteration=double())
importance.data <-  data.frame(Variable=double(), IncMSE=double(), IncNodePurity=d
ouble(), Iteration=double())
model.fit.data <-  data.frame(best.nodesize=double(), best.mtry=double(), best.ntr
ee=double(), best.MSE=double(), best.var=double())

  # quick assimilation loop
  for(frame in seq(1, length(rf.sum.par), 3)){
    partial.plot.data <- rbind(partial.plot.data, as.data.frame(rf.sum.par[frame])
)
    importance.data <- rbind(importance.data, as.data.frame(rf.sum.par[frame+1]))
    model.fit.data <- rbind(model.fit.data, as.data.frame(rf.sum.par[frame+2]))

  }

  names(partial.plot.data) <- c("Age", "Var", "Iteration")
  names(importance.data) <- c("Variable", "IncMSE", "IncNodePurity", "Iteration")
  names(model.fit.data) <- c("best.nodesize", "best.mtry", "best.ntree", "best.MSE
", "best.var")


  # lots of repetition of partial.plot.data and importance.data (renamed multiple
times) - rename for tidiness? should be fine though, just not great practice.
  rf.sum <- list(partial.plot.data, importance.data, model.fit.data)
  names(rf.sum) <- c("partial.plot.data", "importance.data", "model.fit.data")

  return(rf.sum)
}
```

Test Monte Carlo random forest function using primary Mo analyses as an example, with 3 iterations.

```
test <- Monte.Carlo.rF(data = Mo.anox.py.rf,
                       resp.var = "Mo..ppm.",
                       vars = c("age.model",
                                "site.type",
                                "metamorphic.bin",
                                "basin.type",
                                "site.latitude",
                                "site.longitude",
                                "lithology.name",
                                "environmental.bin",
                                "Mo..ppm.",
                                "TOC..wt..",
                                "Fe.py.FeHR",
                                "Al..wt.."),
                       n = 3,
                       run.age.unc = TRUE,
                       run.partial.context = TRUE,
                       run.Al.impute = TRUE,
                       plot.strat = FALSE)
```

```
## socket cluster with 3 nodes on host 'localhost'
```

```
summary(test)
```

```
##                    Length Class      Mode
## partial.plot.data 3      data.frame list
## importance.data   4      data.frame list
## model.fit.data    5      data.frame list
```

```
summary(test$partial.plot.data)
```

```
##       Age             Var          Iteration
##  Min.   :299.8   Min.   :21.57   Min.   :1
##  1st Qu.:434.7   1st Qu.:24.88   1st Qu.:1
##  Median :568.5   Median :26.09   Median :2
##  Mean   :570.4   Mean   :28.88   Mean   :2
##  3rd Qu.:708.1   3rd Qu.:28.55   3rd Qu.:3
##  Max.   :842.8   Max.   :45.98   Max.   :3
```

```
summary(test$importance.data)
```

```
##    Variable            IncMSE       IncNodePurity       Iteration
##  Length:33          Min.   : 0.603   Min.   :  26168   Min.   :1
##  Class :character   1st Qu.: 3.083   1st Qu.:  62027   1st Qu.:1
##  Mode  :character   Median : 4.928   Median : 283097   Median :2
##                     Mean   : 6.343   Mean   : 381078   Mean   :2
##                     3rd Qu.: 7.215   3rd Qu.: 387821   3rd Qu.:3
##                     Max.   :29.874   Max.   :2287082   Max.   :3
```

```
summary(test$model.fit.data)
```

```
##  best.nodesize   best.mtry       best.ntree        best.MSE
##  Min.   :2     Min.   :2.000   Min.   : 16.00   Min.   : 908.4
##  1st Qu.:2     1st Qu.:3.000   1st Qu.: 40.00   1st Qu.: 908.7
##  Median :2     Median :4.000   Median : 64.00   Median : 909.0
##  Mean   :2     Mean   :3.333   Mean   : 69.33   Mean   :1020.0
##  3rd Qu.:2     3rd Qu.:4.000   3rd Qu.: 96.00   3rd Qu.:1075.7
##  Max.   :2     Max.   :4.000   Max.   :128.00   Max.   :1242.4
##     best.var
##  Min.   :57.26
##  1st Qu.:60.21
##  Median :63.16
##  Mean   :62.59
##  3rd Qu.:65.25
##  Max.   :67.34
```

5. Running primary analyses

Set number of iterations. Default for main text analyses is n = 100.

```
n <- 100
```

Molybdenum

For our primary Mo analyses we include all key geologic context variables (“site.type”, “metamorphic.bin”, “basin.type”, “site.latitude”, “site.longitude”, “lithology.name”, “environmental.bin”) as well as geologic age, TOC, Fepy/FeHR and [Al] in our random forest model. We incorporate TOC and Fepy/FeHR to control for the impacts of organic carbon and sulfide availability on Mo reduction rates (note that only anoxic samples are used in these analyses). [Al] is incorporated to control for detrital input.

Note that in all of these random forest function calls figures and results are hidden to prevent extended printouts in the R markdown files, but “fig.show=‘hide’” and “results=‘hide’” can be deleted to see full output.

```
Mo.anox.py.rf.results <- Monte.Carlo.rF(data = Mo.anox.py.rf,
                        resp.var = "Mo..ppm.",
                        vars = c("age.model",
                                 "site.type",
                                 "metamorphic.bin",
                                 "basin.type",
                                 "site.latitude",
                                 "site.longitude",
                                 "lithology.name",
                                 "environmental.bin",
                                 "Mo..ppm.",
                                 "TOC..wt..",
                                 "Fe.py.FeHR",
                                 "Al..wt.."),
                        n = n,
                        run.age.unc = TRUE,
                        run.partial.context = TRUE,
                        run.Al.impute = TRUE)

Mo.anox.py.rf.partial <- Mo.anox.py.rf.results$partial.plot.data
Mo.anox.py.rf.import <- Mo.anox.py.rf.results$importance.data
Mo.anox.py.rf.model <- Mo.anox.py.rf.results$model.fit.data
```

Uranium

For our primary U analyses we include all key geologic context variables (“site.type”, “metamorphic.bin”, “basin.type”, “site.latitude”, “site.longitude”, “lithology.name”, “environmental.bin”) as well as geologic age, TOC and [Al] in our random forest model. We incorporate TOC to control for the impacts of organic carbon availability on U reduction rates (note that only anoxic samples are used in these analyses). [Al] is incorporated to control for detrital input.

```
U.anox.rf.results <- Monte.Carlo.rF(data = U.anox.rf,
                        resp.var = "U..ppm.",
                        vars = c("age.model",
                                 "site.type",
                                 "metamorphic.bin",
                                 "basin.type",
                                 "site.latitude",
                                 "site.longitude",
                                 "lithology.name",
                                 "environmental.bin",
                                 "U..ppm.",
                                 "TOC..wt..",
                                 "Al..wt.."),
                        n = n,
                        run.age.unc = TRUE,
                        run.partial.context = TRUE,
                        run.Al.impute = TRUE)

U.anox.rf.partial <- U.anox.rf.results$partial.plot.data
U.anox.rf.import <- U.anox.rf.results$importance.data
U.anox.rf.model <- U.anox.rf.results$model.fit.data
```

Proportion euxinic

For our primary proportion euxinic analyses we include all key geologic context variables (“site.type”, “metamorphic.bin”, “basin.type”, “site.latitude”, “site.longitude”, “lithology.name”, “environmental.bin”) as well as geologic age and [Al] in our random forest model. [Al] is incorporated to control for detrital input. Note that only anoxic samples are used in these analyses.

Note that unless warnings are hidden (as they are in this R Markdown file) a warning appears asking if we are sure we want to do a regression because of the binary response variable. In this case we do, following a similar approach in Sperling et al. 2015 (Nature). For completeness, this warning would read: “## Warning in randomForest.default(m, y, …): The response has five or fewer unique values. Are you sure you want to do regression?” if it was not hidden.

```
Fepy.anox.rf.results <- Monte.Carlo.rF(data = Fepy.anox.rf,
                        resp.var = "euxinic.Fe",
                        vars = c("age.model",
                                 "site.type",
                                 "metamorphic.bin",
                                 "basin.type",
                                 "site.latitude",
                                 "site.longitude",
                                 "lithology.name",
                                 "environmental.bin",
                                 "euxinic.Fe",
                                 "Al..wt.."),
                        n = n,
                        run.age.unc = TRUE,
                        run.partial.context = TRUE,
                        run.Al.impute = TRUE)

Fepy.anox.rf.partial <- Fepy.anox.rf.results$partial.plot.data
Fepy.anox.rf.import <- Fepy.anox.rf.results$importance.data
Fepy.anox.rf.model <- Fepy.anox.rf.results$model.fit.data
```

Total Organic Carbon

For our primary TOC analyses we include all key geologic context variables (“site.type”, “metamorphic.bin”, “basin.type”, “site.latitude”, “site.longitude”, “lithology.name”, “environmental.bin”) as well as geologic age and [Al] in our random forest model. [Al] is incorporated to control for detrital input.

```
TOC.all.rf.results <- Monte.Carlo.rF(data = TOC.all.rf,
                        resp.var = "TOC..wt..",
                        vars = c("age.model",
                                 "site.type",
                                 "metamorphic.bin",
                                 "basin.type",
                                 "site.latitude",
                                 "site.longitude",
                                 "lithology.name",
                                 "environmental.bin",
                                 "TOC..wt..",
                                 "Al..wt.."),
                        n = n,
                        run.age.unc = TRUE,
                        run.partial.context = TRUE,
                        run.Al.impute = TRUE)

TOC.all.rf.partial <- TOC.all.rf.results$partial.plot.data
TOC.all.rf.import <- TOC.all.rf.results$importance.data
TOC.all.rf.model <- TOC.all.rf.results$model.fit.data
```

6. Function to interpolate partial

dependence plot lines

To plot envelopes that summarizing all n (100) random forests, we need to interpolate the lines drawn between points defined in partial dependence plots. We can then evaluate the distribution of partial dependence plot values at each timestep.

```
interp.pdp <- function(data, age.rounding.factor, n){
  # data - partial dependence plot dataframe
  # age.rounding.factor - number of digits (decimal places) for millions of years
  # n - number of iterations

  # initiate dataframe to store interpolated PDP values
  PDPs.interpolated <-  data.frame(Age=double(), Var=double(), Iteration=double())

  # Before starting, round age splits to nearest x Myrs (if age.rounding factor =
0, 1 Myrs; if age.rounding factor = 1, 0.1 Myrs)
  data$Age.rounded <- round(data$Age, digits=age.rounding.factor)

  # loop through n iterations of Monte Carlo random forest analysis
  for (It in 1:n){
  # create subset for specific iteration
  It.subset <- filter(data, Iteration == It)

  # interpolate values at each timestep
  Interpolated.Age.Var <- approx(x = It.subset$Age.rounded,
                                 y = It.subset$Var,
                                 xout = seq(min(It.subset$Age.rounded, na.rm = T),
                                            max(It.subset$Age.rounded, na.rm = T),
                                            10^-age.rounding.factor),
                                 method = "linear",
                                 ties = mean) %>%
    as.data.frame() %>%
    setNames(c("Age", "Var")) # only used for visual inspection of this dataframe

  # Add interpolated data to summary dataframe
  i <- nrow(PDPs.interpolated)+1
  j <- nrow(PDPs.interpolated)+nrow(Interpolated.Age.Var)
  PDPs.interpolated[i:j , 1:2] <- Interpolated.Age.Var
  PDPs.interpolated[i:j , 3] <- It
  }

  # Summarize interpolated PDPs
  # convert age to factor and then convert back otherwise end up with occasional w
eird duplicates using group_by() on a numerical variable.
  Var.sum <- PDPs.interpolated %>%
  group_by(as.factor(Age)) %>%
  summarize(min = min(Var, na.rm=T),
            perc.05 = quantile(Var, probs=0.05, na.rm=T),
            perc.25 = quantile(Var, probs=0.25, na.rm=T),
            median = median(Var, na.rm=T),
            mean = mean(Var, na.rm=T),
            perc.75 = quantile(Var, probs=0.75, na.rm=T),
            perc.95 = quantile(Var, probs=0.95, na.rm=T),
```

```
            max = max(Var, na.rm=T)) %>%
  as.data.frame()

  # rename as.factor(Age) column "Age"
  names(Var.sum)[1] <- "Age"
  # make Age numeric again
  Var.sum$Age <- as.numeric(paste(Var.sum$Age))

  return(Var.sum)
}
```

Test function using primary Mo analyses. Do lines based on the (sparse) partial dependence plot points generated by the random forest function sit within the min-max envelope generated by our function? In this study we use an age rounding factor of 1 (to nearest 0.1Myrs) for the sake of computational efficiency but the patterns are observed at rounding factors of 2 (0.01Myrs) and 0 (1Myrs). Note that really coarse rounding factors (e.g. a factor of -1, or 10Myrs) will not accurately reproduce the partial dependence plots generated by the random forest function.

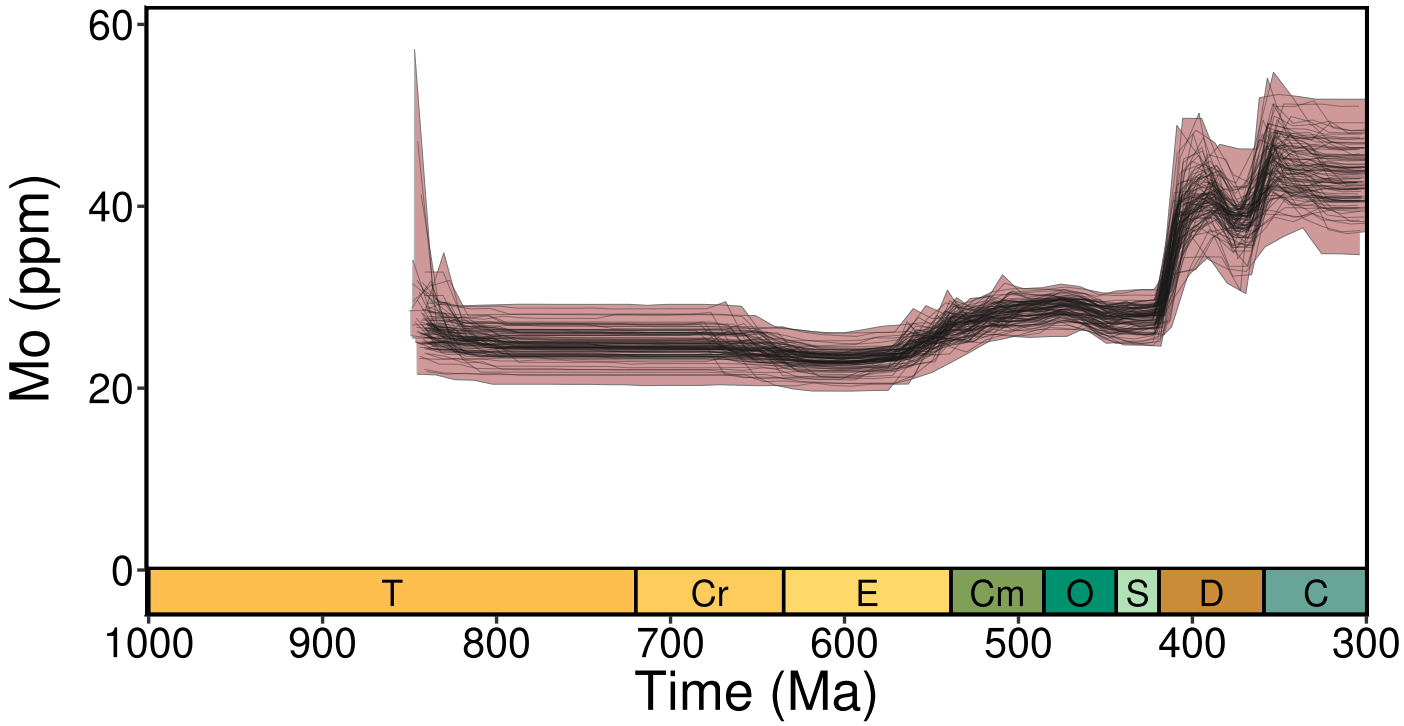
```
mo.pdp.test <- interp.pdp(data = Mo.anox.py.rf.partial, age.rounding.factor = 1, n
= n)

Mo.test.plot <- ggplot()+
  geom_ribbon(data = mo.pdp.test, aes(x = Age, ymin = min, ymax = max), alpha=.4,
fill="darkred", color="grey70", size=.3)+
  geom_path(data = Mo.anox.py.rf.partial, aes(x = Age, y = Var, group = Iteration)
, alpha = 0.4, size = 0.3, color = "grey10")+
  theme_bw()+
  coord_geo(xlim=rev(c(298.9,1002)), expand=FALSE, ylim=c(0,62),
            pos = as.list(rep("bottom", 1)),
            abbrv=list(TRUE),
            dat = list(periods.edit),
            height = list(unit(2, "lines")),
            bord=list(c("left", "bottom", "right")), lwd=as.list(c(1)), size=8)+
  scale_x_reverse(breaks=c(300,400,500,600,700,800,900,1000))+
  ylab("Mo (ppm)")+xlab("Time (Ma)")+
  theme(plot.margin = ggplot2::margin(.3,1,.3,1,"cm"),panel.border = element_rect(
fill=NA,color="black", size=2,linetype="solid"),
        axis.ticks = element_line(size=1),
        axis.line = element_line(lineend = 'square'),
        axis.title = element_text(size=34),
        axis.text = element_text( size=26, color="black"),
        legend.title = element_text(size=20),
        legend.text = element_text( size=16),
        axis.ticks.length = unit(5, "points"),
        legend.position="top",         legend.justification = c(0, 0),
        axis.title.y = element_text(vjust=3),
        panel.grid.major = element_blank(),panel.grid.minor = element_blank())
```

```
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use `linewidth` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```
## Warning: The `size` argument of `element_line()` is deprecated as of ggplot2 3.
4.0.
## i Please use the `linewidth` argument instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```
## Warning: The `size` argument of `element_rect()` is deprecated as of ggplot2 3.
4.0.
## i Please use the `linewidth` argument instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```
Mo.test.plot
```



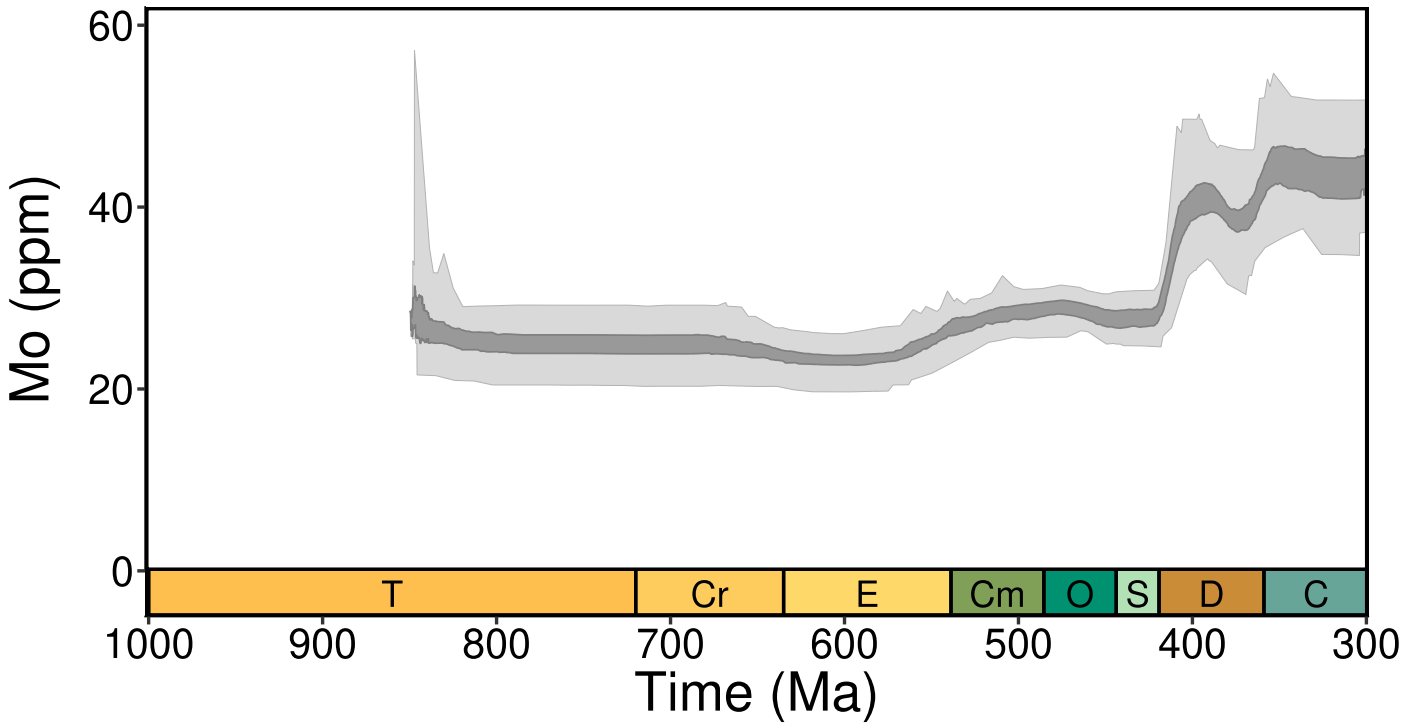
All of the individual lines lie within the envelope of minimum and maximum values defined by our interp.pdp() function.

7. Plotting primary analyses

In these plots we summarize the 100 partial dependence plots run for each analysis as envelopes. The lighter gray envelope describes the maximum and minimum values for each timestep and the darker gray envelope describes the 25th and 75th percentiles.

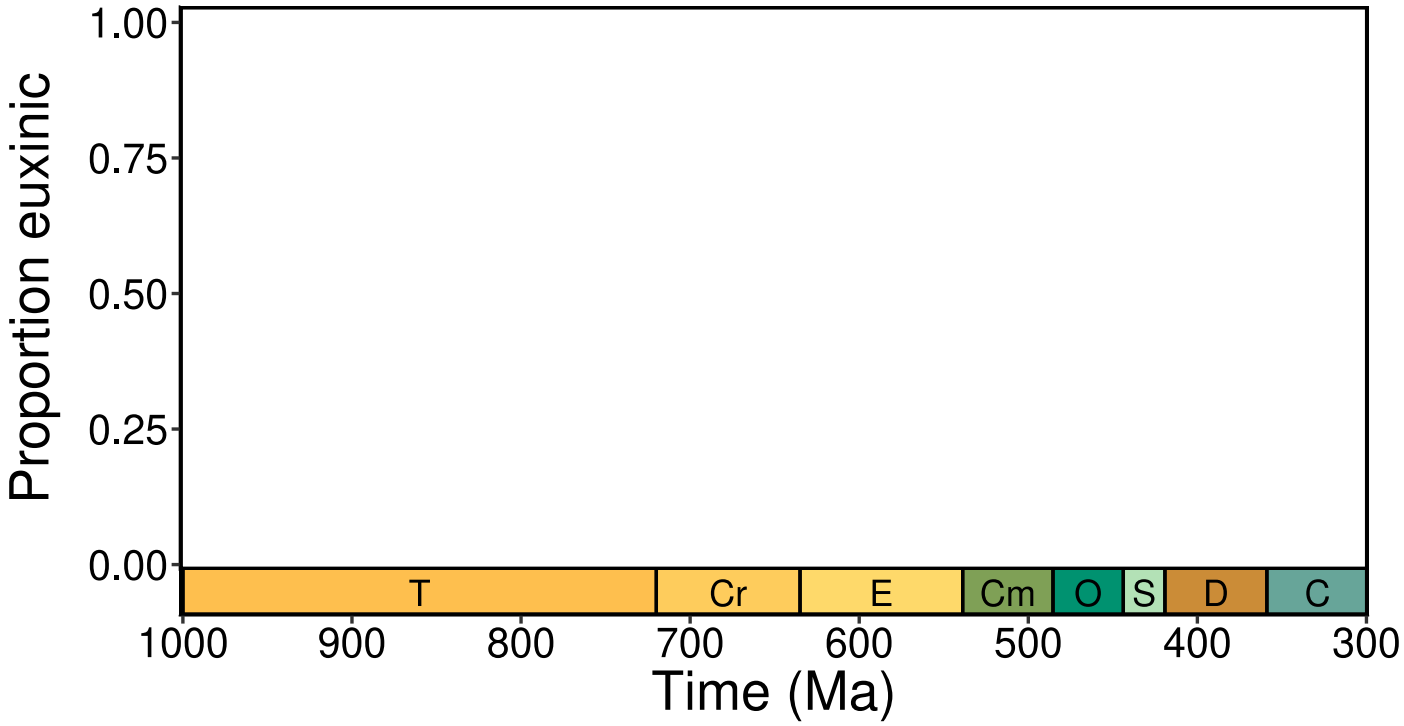
Molybdenum

```
mo.pdp <- interp.pdp(data = Mo.anox.py.rf.partial, age.rounding.factor = 1, n = n)

Mo.plot <- ggplot()+
  geom_ribbon(data = mo.pdp, aes(x = Age, ymin = min, ymax = max), alpha=1, fill="
grey85", color="grey70", size=.3)+
  geom_ribbon(data = mo.pdp, aes(x = Age, ymin = perc.25, ymax = perc.75), alpha=1
, fill="grey60", color="grey50", size=.5)+
  theme_bw()+
  coord_geo(xlim=rev(c(298.9,1002)), expand=FALSE, ylim=c(0,62),
            pos = as.list(rep("bottom", 1)),
            abbrv=list(TRUE),
            dat = list(periods.edit),
            height = list(unit(2, "lines")),
            bord=list(c("left", "bottom", "right")), lwd=as.list(c(1)), size=8)+
  scale_x_reverse(breaks=c(300,400,500,600,700,800,900,1000))+
  ylab("Mo (ppm)")+xlab("Time (Ma)")+
  theme(plot.margin = ggplot2::margin(.3,1,.3,1,"cm"),panel.border = element_rect(
fill=NA,color="black", size=2,linetype="solid"),
        axis.ticks = element_line(size=1),
        axis.line = element_line(lineend = 'square'),
        axis.title = element_text(size=34),
        axis.text = element_text( size=26, color="black"),
        legend.title = element_text(size=20),
        legend.text = element_text( size=16),
        axis.ticks.length = unit(5, "points"),
        legend.position="top",         legend.justification = c(0, 0),
        axis.title.y = element_text(vjust=3),
        panel.grid.major = element_blank(),panel.grid.minor = element_blank())

Mo.plot
```

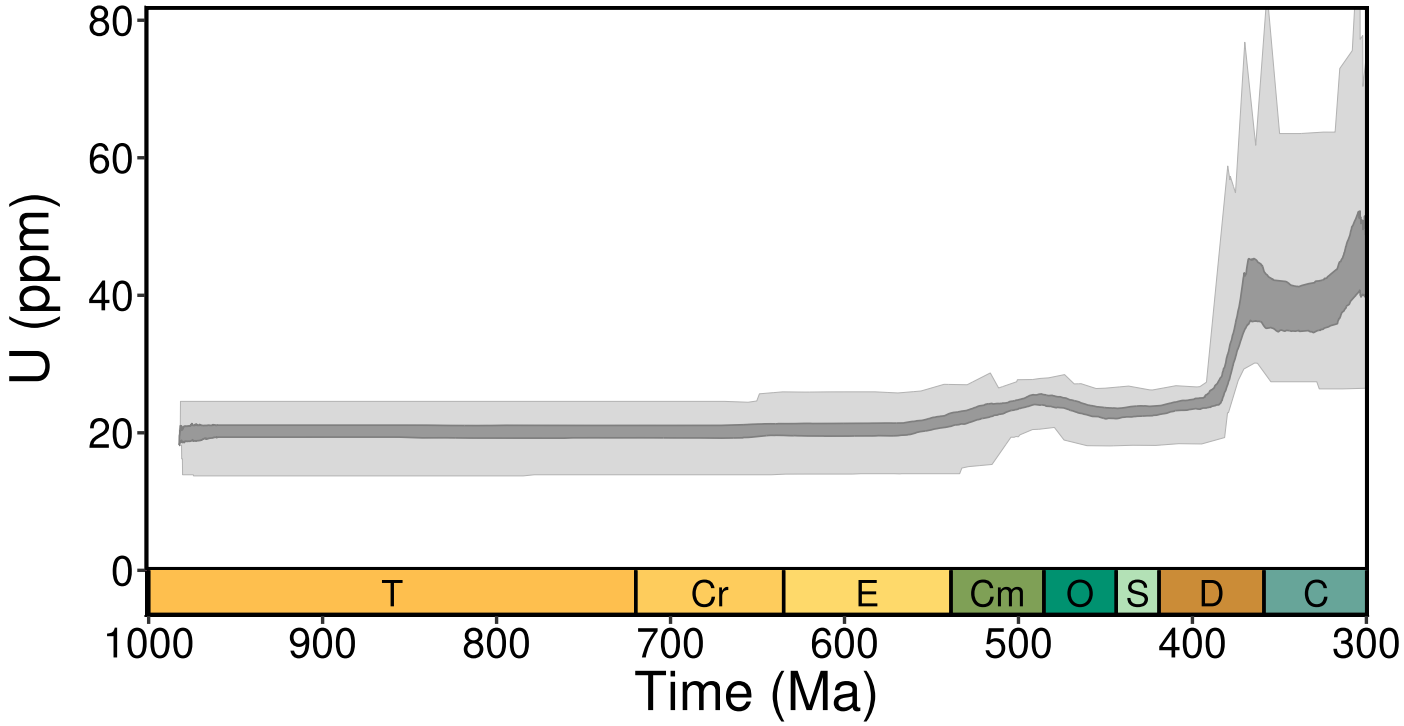


Uranium

```
u.pdp <- interp.pdp(data = U.anox.rf.partial, age.rounding.factor = 1, n = n)

U.plot <- ggplot()+
  geom_ribbon(data = u.pdp, aes(x = Age, ymin = min, ymax = max), alpha=1, fill="g
rey85", color="grey70", size=.3)+
  geom_ribbon(data = u.pdp, aes(x = Age, ymin = perc.25, ymax = perc.75), alpha=1,
fill="grey60", color="grey50", size=.5)+
  theme_bw()+
  coord_geo(xlim=rev(c(298.9,1002)), expand=FALSE, ylim=c(0,82),
            pos = as.list(rep("bottom", 1)),
            abbrv=list(TRUE),
            dat = list(periods.edit),
            height = list(unit(2, "lines")),
            bord=list(c("left", "bottom", "right")), lwd=as.list(c(1)), size=8)+
  scale_x_reverse(breaks=c(300,400,500,600,700,800,900,1000))+
  ylab("U (ppm)")+xlab("Time (Ma)")+
  theme(plot.margin = ggplot2::margin(.3,1,.3,1,"cm"),panel.border = element_rect(
fill=NA,color="black", size=2,linetype="solid"),
        axis.ticks = element_line(size=1),
        axis.line = element_line(lineend = 'square'),
        axis.title = element_text(size=34),
        axis.text = element_text( size=26, color="black"),
        legend.title = element_text(size=20),
        legend.text = element_text( size=16),
        axis.ticks.length = unit(5, "points"),
        legend.position="top",         legend.justification = c(0, 0),
        axis.title.y = element_text(vjust=3),
        panel.grid.major = element_blank(),panel.grid.minor = element_blank())

ggplot()+
  geom_line(data = u.pdp, aes(x = Age, y=min), alpha=1, color="grey70", size=.3)+
  theme_bw()+
  coord_geo(xlim=rev(c(298.9,1002)), expand=FALSE, ylim=c(-.01,1.03),
            pos = as.list(rep("bottom", 1)),
            abbrv=list(TRUE),
            dat = list(periods.edit),
            height = list(unit(2, "lines")),
            bord=list(c("left", "bottom", "right")), lwd=as.list(c(1)), size=8)+
  scale_x_reverse(breaks=c(300,400,500,600,700,800,900,1000))+
  ylab("Proportion euxinic")+xlab("Time (Ma)")+
  theme(plot.margin = ggplot2::margin(.3,1,.3,1,"cm"),panel.border = element_rect(
fill=NA,color="black", size=2,linetype="solid"),
        axis.ticks = element_line(size=1),
        axis.line = element_line(lineend = 'square'),
        axis.title = element_text(size=34),
        axis.text = element_text( size=26, color="black"),
        legend.title = element_text(size=20),
        legend.text = element_text( size=16),
        axis.ticks.length = unit(5, "points"),
        legend.position="top",         legend.justification = c(0, 0),
        axis.title.y = element_text(vjust=3),
        panel.grid.major = element_blank(),panel.grid.minor = element_blank())
```



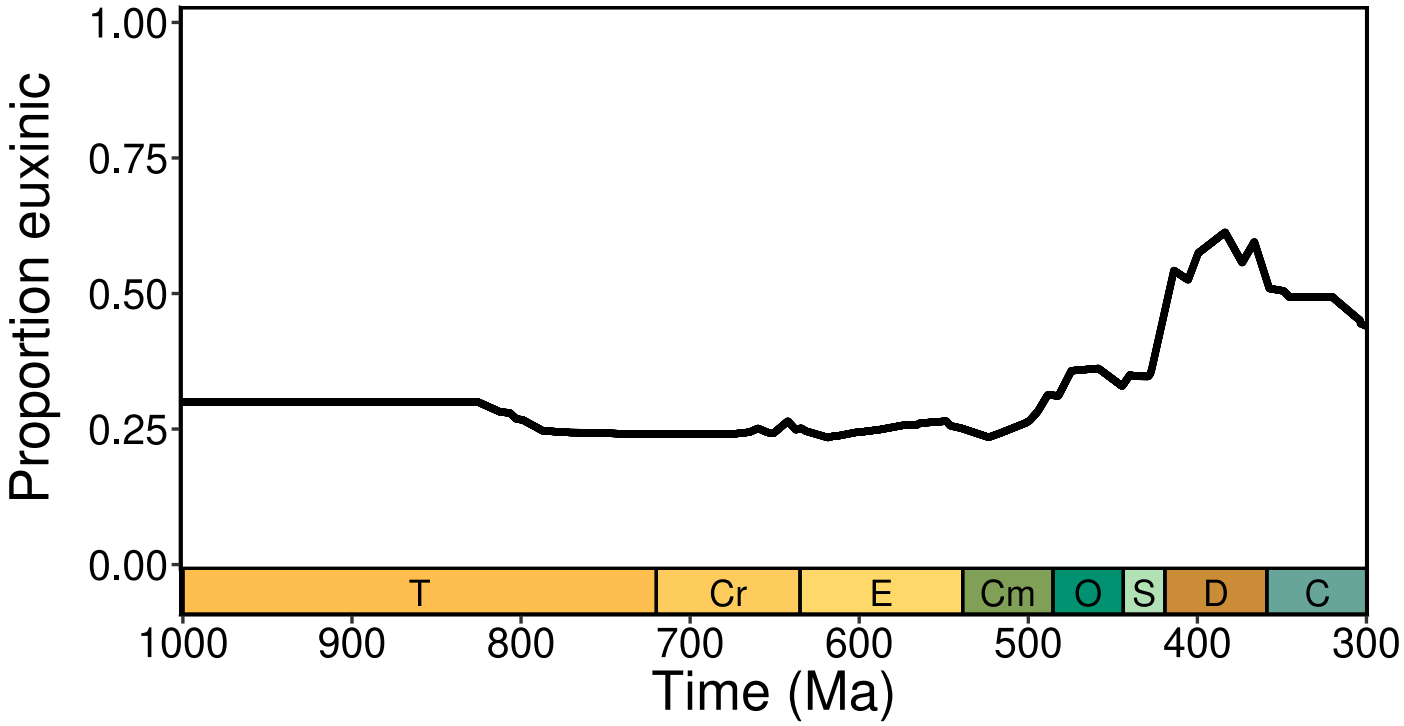

```
U.plot
```



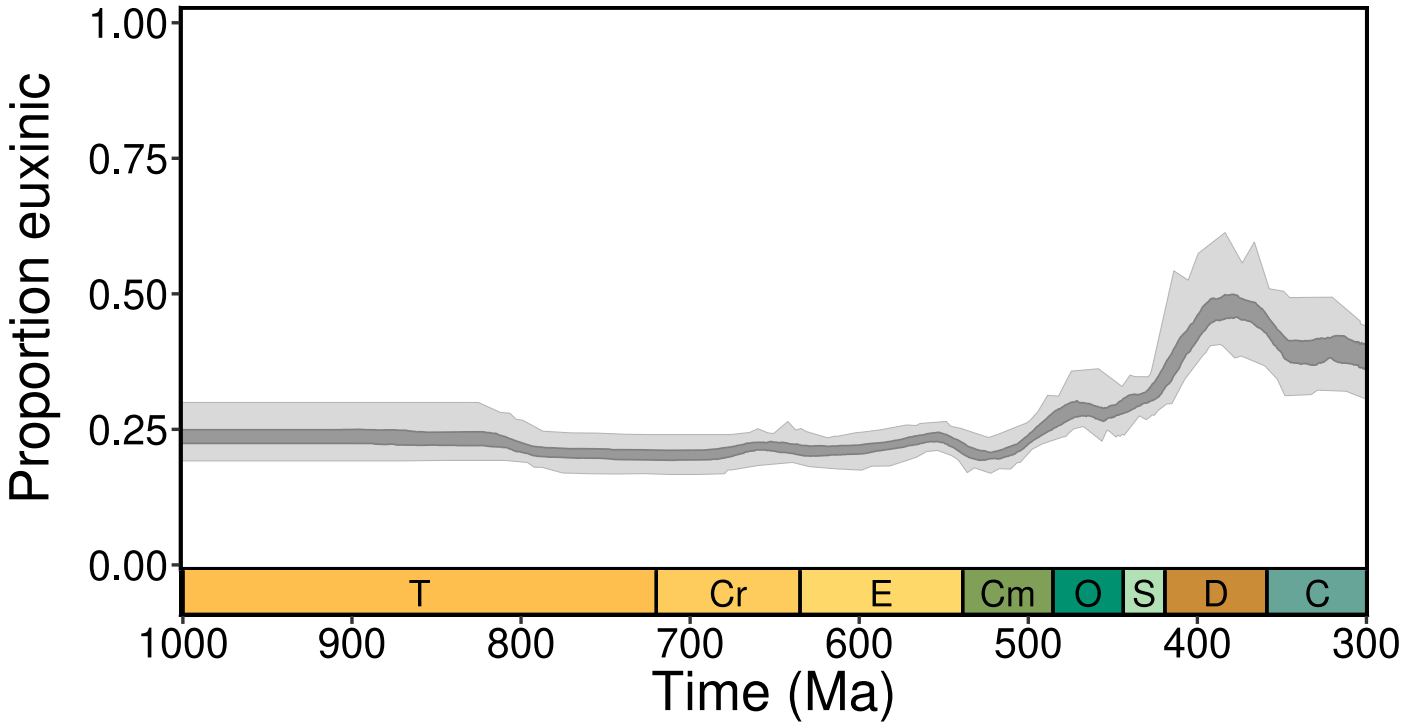
Proportion euxinic

Note that we hide warnings here because of random forest warning about using binary variable in a regression (here we do want to do a regression to replicate the broad approach of Sperling et al. 2015, Nature - see similar comments above).

```
prop_eux.pdp <- interp.pdp(data = Fepy.anox.rf.partial, age.rounding.factor = 1, n
= n)

prop_eux.plot <- ggplot()+
  geom_ribbon(data = prop_eux.pdp, aes(x = Age, ymin = min, ymax = max), alpha=1,
fill="grey85", color="grey70", size=.3)+
  geom_ribbon(data = prop_eux.pdp, aes(x = Age, ymin = perc.25, ymax = perc.75), a
lpha=1, fill="grey60", color="grey50", size=.5)+
  theme_bw()+
  coord_geo(xlim=rev(c(298.9,1002)), expand=FALSE, ylim=c(-.01,1.03),
            pos = as.list(rep("bottom", 1)),
            abbrv=list(TRUE),
            dat = list(periods.edit),
            height = list(unit(2, "lines")),
            bord=list(c("left", "bottom", "right")), lwd=as.list(c(1)), size=8)+
  scale_x_reverse(breaks=c(300,400,500,600,700,800,900,1000))+
  ylab("Proportion euxinic")+xlab("Time (Ma)")+
  theme(plot.margin = ggplot2::margin(.3,1,.3,1,"cm"),panel.border = element_rect(
fill=NA,color="black", size=2,linetype="solid"),
        axis.ticks = element_line(size=1),
        axis.line = element_line(lineend = 'square'),
        axis.title = element_text(size=34),
        axis.text = element_text( size=26, color="black"),
        legend.title = element_text(size=20),
        legend.text = element_text( size=16),
        axis.ticks.length = unit(5, "points"),
        legend.position="top",         legend.justification = c(0, 0),
        axis.title.y = element_text(vjust=3),
        panel.grid.major = element_blank(),panel.grid.minor = element_blank())

ggplot()+
  geom_point(data = prop_eux.pdp, aes(x = Age, y=max))+
  theme_bw()+
  coord_geo(xlim=rev(c(298.9,1002)), expand=FALSE, ylim=c(-.01,1.03),
            pos = as.list(rep("bottom", 1)),
            abbrv=list(TRUE),
            dat = list(periods.edit),
            height = list(unit(2, "lines")),
            bord=list(c("left", "bottom", "right")), lwd=as.list(c(1)), size=8)+
  scale_x_reverse(breaks=c(300,400,500,600,700,800,900,1000))+
  ylab("Proportion euxinic")+xlab("Time (Ma)")+
  theme(plot.margin = ggplot2::margin(.3,1,.3,1,"cm"),panel.border = element_rect(
fill=NA,color="black", size=2,linetype="solid"),
        axis.ticks = element_line(size=1),
        axis.line = element_line(lineend = 'square'),
        axis.title = element_text(size=34),
        axis.text = element_text( size=26, color="black"),
        legend.title = element_text(size=20),
        legend.text = element_text( size=16),
        axis.ticks.length = unit(5, "points"),
        legend.position="top",         legend.justification = c(0, 0),
        axis.title.y = element_text(vjust=3),
        panel.grid.major = element_blank(),panel.grid.minor = element_blank())
```

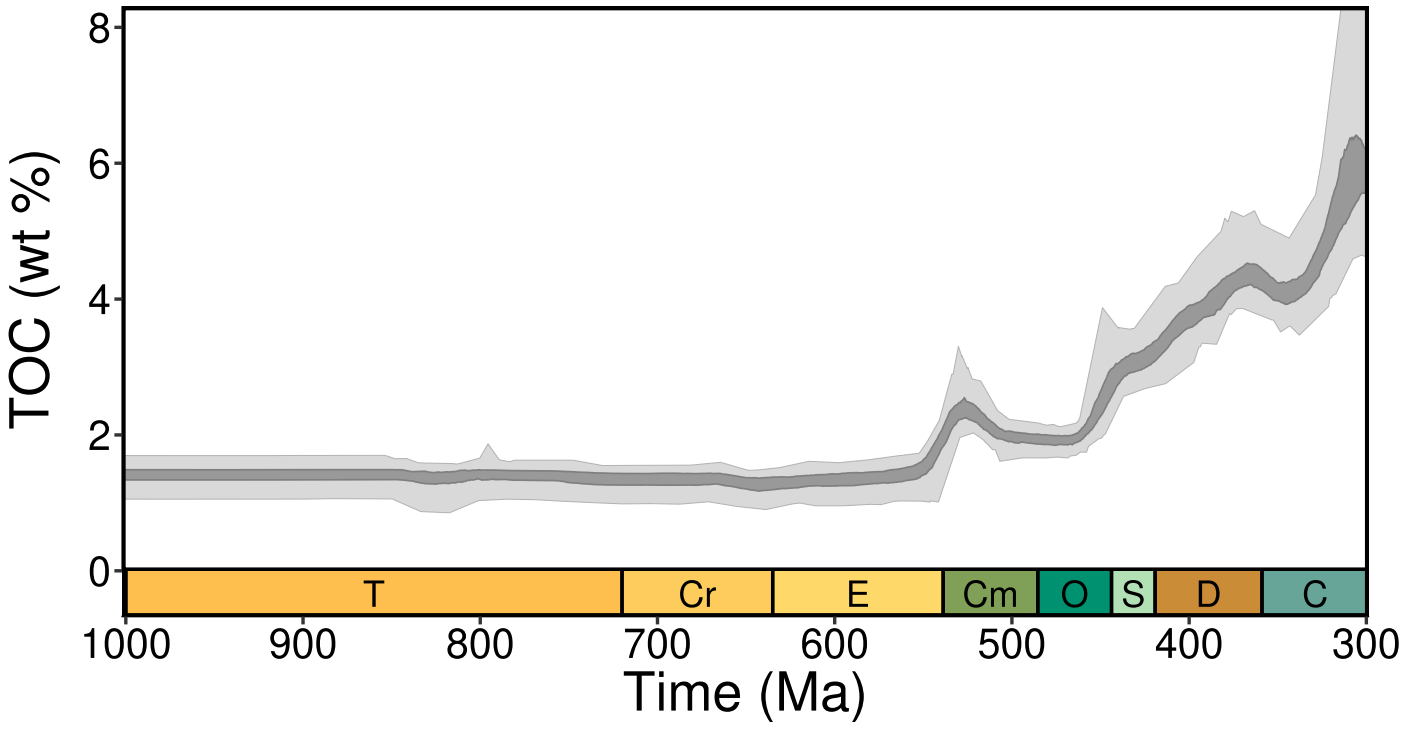



```
prop_eux.plot
```



Total Organic Carbon

```
TOC.pdp <- interp.pdp(data = TOC.all.rf.partial, age.rounding.factor = 1, n = n)

TOC.plot <- ggplot()+
  geom_ribbon(data = TOC.pdp, aes(x = Age, ymin = min, ymax = max), alpha=1, fill=
"grey85", color="grey70", size=.3)+
  geom_ribbon(data = TOC.pdp, aes(x = Age, ymin = perc.25, ymax = perc.75), alpha=
1, fill="grey60", color="grey50", size=.5)+
  theme_bw()+
  coord_geo(xlim=rev(c(298.9,1002)), expand=FALSE, ylim=c(0,8.3),
            pos = as.list(rep("bottom", 1)),
            abbrv=list(TRUE),
            dat = list(periods.edit),
            height = list(unit(2, "lines")),
            bord=list(c("left", "bottom", "right")), lwd=as.list(c(1)), size=8)+
  scale_x_reverse(breaks=c(300,400,500,600,700,800,900,1000))+
  ylab("TOC (wt %)")+xlab("Time (Ma)")+
  theme(plot.margin = ggplot2::margin(.3,1,.3,1,"cm"),panel.border = element_rect(
fill=NA,color="black", size=2,linetype="solid"),
        axis.ticks = element_line(size=1),
        axis.line = element_line(lineend = 'square'),
        axis.title = element_text(size=34),
        axis.text = element_text( size=26, color="black"),
        legend.title = element_text(size=20),
        legend.text = element_text( size=16),
        axis.ticks.length = unit(5, "points"),
        legend.position="top",         legend.justification = c(0, 0),
        axis.title.y = element_text(vjust=3),
        panel.grid.major = element_blank(),panel.grid.minor = element_blank())

TOC.plot
```



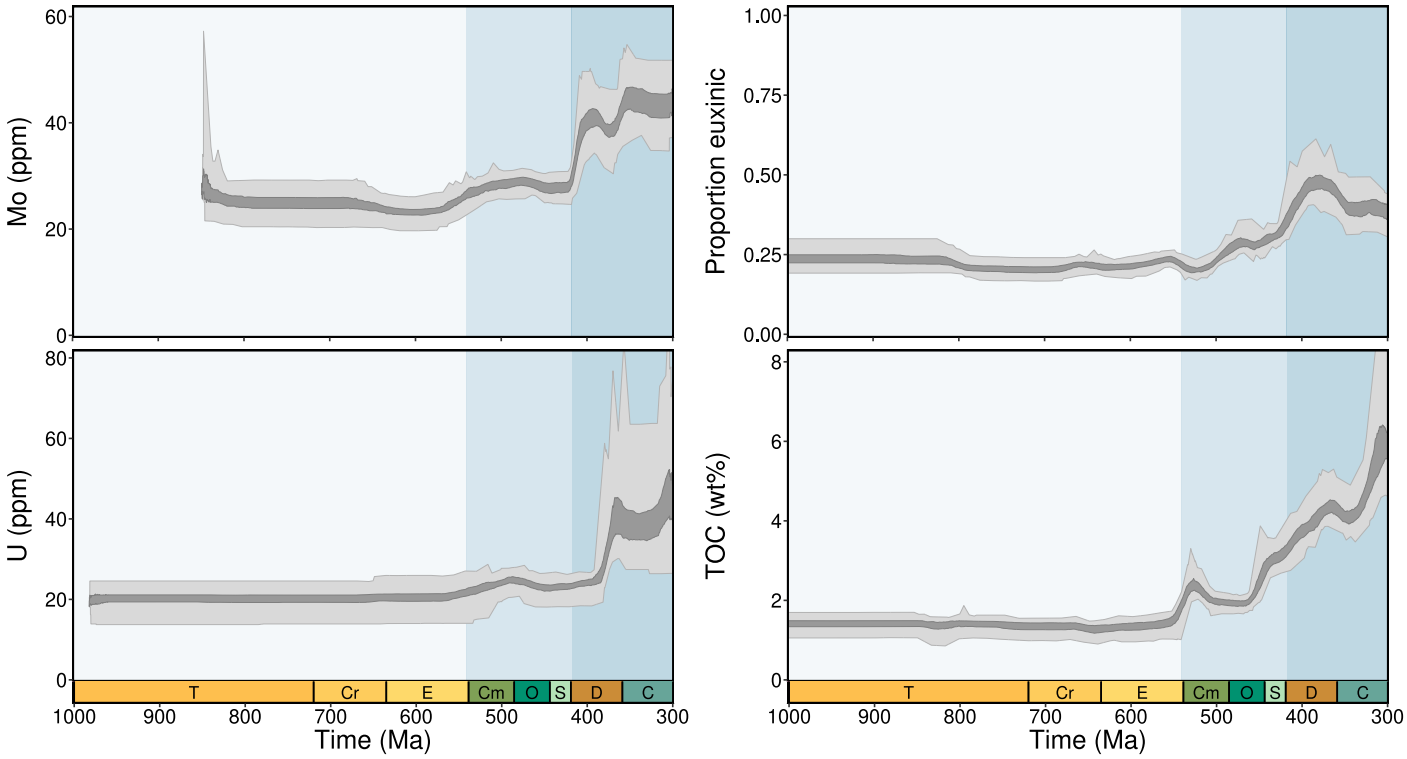
Combine to generate primary summary plot

In these plots we have delineated the three main phases of marine biogeochemistry between 1000 and 300Ma - with transitions around the base of of the Cambrian and Devonian periods.

```
Mo.plot.for.sum <- ggplot(mo.pdp, aes(x=Age))+
  annotate(geom="rect", xmax=Inf, xmin=541, ymin=-Inf, ymax=Inf, fill="#C9DEE8", a
lpha=0.2)+
  annotate(geom="rect", xmax=541, xmin=418, ymin=-Inf, ymax=Inf, fill="#9DC2D6", a
lpha=0.4)+
  annotate(geom="rect", xmax=418, xmin=-Inf, ymin=-Inf, ymax=Inf, fill="#619FBB",
alpha=0.4)+
  geom_ribbon(aes(ymin = min, ymax = max), alpha=1, fill="grey85", color="grey70",
size=.3)+
  geom_ribbon(aes(ymin = perc.25, ymax = perc.75), alpha=1, fill="grey60", color="
grey50", size=.5)+
  theme_bw()+
  coord_cartesian(xlim=rev(c(298.9,1000)), ylim=c(-.4,62),expand=FALSE)+
  scale_x_reverse(breaks=c(300,400,500,600,700,800,900,1000))+
  ylab("Mo (ppm)")+xlab("Time (Ma)")+
  theme(plot.margin = ggplot2::margin(.1,1,.3,1,"cm"),panel.border = element_rect(
fill=NA,color="black", size=2,linetype="solid"),
        axis.ticks = element_line(size=1),
        axis.line = element_line(lineend = 'square'),
        axis.title = element_text(size=34),
        axis.text = element_text( size=26, color="black"),
        legend.title = element_text(size=20),
        legend.text = element_text( size=16),
        axis.ticks.length = unit(5, "points"),
        legend.position="top",         legend.justification = c(0, 0),
        axis.title.x = element_blank(),
        axis.text.x = element_blank(),
        axis.title.y = element_text(vjust=3),
        panel.grid.major = element_blank(),panel.grid.minor = element_blank())

U.plot.for.sum <- ggplot(u.pdp, aes(x=Age))+
  annotate(geom="rect", xmax=Inf, xmin=541, ymin=-Inf, ymax=Inf, fill="#C9DEE8", a
lpha=0.2)+
  annotate(geom="rect", xmax=541, xmin=418, ymin=-Inf, ymax=Inf, fill="#9DC2D6", a
lpha=0.4)+
  annotate(geom="rect", xmax=418, xmin=-Inf, ymin=-Inf, ymax=Inf, fill="#619FBB",
alpha=0.4)+
  geom_ribbon(aes(ymin = min, ymax = max), alpha=1, fill="grey85", color="grey70",
size=.3)+
  geom_ribbon(aes(ymin = perc.25, ymax = perc.75), alpha=1, fill="grey60", color="
grey50", size=.5)+
  theme_bw()+
  coord_geo(xlim=rev(c(298.9,1002)), expand=FALSE, ylim=c(-.4,82),
            pos = as.list(rep("bottom", 1)),
            abbrv=list(TRUE),
            dat = list(periods.edit),
            height = list(unit(2, "lines")),
            bord=list(c("left", "bottom", "right")), lwd=as.list(c(1)), size=8)+
  scale_x_reverse(breaks=c(300,400,500,600,700,800,900,1000))+
  ylab("U (ppm)")+xlab("Time (Ma)")+
```

```
  theme(plot.margin = ggplot2::margin(.1,1,.3,1,"cm"),panel.border = element_rect(
fill=NA,color="black", size=2,linetype="solid"),
        axis.ticks = element_line(size=1),
        axis.line = element_line(lineend = 'square'),
        axis.title = element_text(size=34),
        axis.text = element_text( size=26, color="black"),
        legend.title = element_text(size=20),
        legend.text = element_text( size=16),
        axis.ticks.length = unit(5, "points"),
        legend.position="top",
        legend.justification = c(0, 0),
        axis.title.y = element_text(vjust=3),
        panel.grid.major = element_blank(),panel.grid.minor = element_blank())

prop_eux.plot.for.sum <- ggplot(prop_eux.pdp, aes(x=Age))+
  annotate(geom="rect", xmax=Inf, xmin=541, ymin=-Inf, ymax=Inf, fill="#C9DEE8", a
lpha=0.2)+
  annotate(geom="rect", xmax=541, xmin=418, ymin=-Inf, ymax=Inf, fill="#9DC2D6", a
lpha=0.4)+
  annotate(geom="rect", xmax=418, xmin=-Inf, ymin=-Inf, ymax=Inf, fill="#619FBB",
alpha=0.4)+
  geom_ribbon(aes(ymin = min, ymax = max), alpha=1, fill="grey85", color="grey70",
size=.3)+
  geom_ribbon(aes(ymin = perc.25, ymax = perc.75), alpha=1, fill="grey60", color="
grey50", size=.5)+
  theme_bw()+
  coord_cartesian(xlim=rev(c(298.9,1000)), ylim=c(-.01,1.03),expand=FALSE)+
  scale_x_reverse(breaks=c(300,400,500,600,700,800,900,1000))+
  ylab("Proportion euxinic")+xlab("Time (Ma)")+
  theme(plot.margin = ggplot2::margin(.1,1,.3,1,"cm"),panel.border = element_rect(
fill=NA,color="black", size=2,linetype="solid"),
        axis.ticks = element_line(size=1),
        axis.line = element_line(lineend = 'square'),
        axis.title = element_text(size=34),
        axis.text = element_text( size=26, color="black"),
        legend.title = element_text(size=20),
        legend.text = element_text( size=16),
        axis.ticks.length = unit(5, "points"),
        legend.position="top",         legend.justification = c(0, 0),
        axis.title.x = element_blank(),
        axis.text.x = element_blank(),
        axis.title.y = element_text(vjust=3),
        panel.grid.major = element_blank(),panel.grid.minor = element_blank())

TOC.plot.for.sum <- ggplot(TOC.pdp, aes(x=Age))+
  annotate(geom="rect", xmax=Inf, xmin=541, ymin=-Inf, ymax=Inf, fill="#C9DEE8", a
lpha=0.2)+
  annotate(geom="rect", xmax=541, xmin=418, ymin=-Inf, ymax=Inf, fill="#9DC2D6", a
lpha=0.4)+
  annotate(geom="rect", xmax=418, xmin=-Inf, ymin=-Inf, ymax=Inf, fill="#619FBB",
alpha=0.4)+
  geom_ribbon(aes(ymin = min, ymax = max), alpha=1, fill="grey85", color="grey70",
size=.3)+
  geom_ribbon(aes(ymin = perc.25, ymax = perc.75), alpha=1, fill="grey60", color="
```

```
grey50", size=.5)+
  theme_bw()+
  coord_geo(xlim=rev(c(298.9,1002)), expand=FALSE, ylim=c(-0.04,8.3),
            pos = as.list(rep("bottom", 1)),
            abbrv=list(TRUE),
            dat = list(periods.edit),
            height = list(unit(2, "lines")),
            bord=list(c("left", "bottom", "right")), lwd=as.list(c(1)), size=8)+
  scale_x_reverse(breaks=c(300,400,500,600,700,800,900,1000))+
  ylab("TOC (wt%)")+xlab("Time (Ma)")+
  theme(plot.margin = ggplot2::margin(.1,1,.3,1,"cm"),panel.border = element_rect(
fill=NA,color="black", size=2,linetype="solid"),
        axis.ticks = element_line(size=1),
        axis.line = element_line(lineend = 'square'),
        axis.title = element_text(size=34),
        axis.text = element_text( size=26, color="black"),
        legend.title = element_text(size=20),
        legend.text = element_text( size=16),
        axis.ticks.length = unit(5, "points"),
        legend.position="top",
        legend.justification = c(0, 0),
        axis.title.y = element_text(vjust=3),
        panel.grid.major = element_blank(),panel.grid.minor = element_blank())

summary.plot <- ggarrange2(Mo.plot.for.sum, prop_eux.plot.for.sum, U.plot.for.sum,
TOC.plot.for.sum, ncol=2, heights=c(1,1))
```



```
ggsave("Figure 2 Partial Dependence Plot with shading 20240207 100 iterations.pd
f", summary.plot, height=14, width=26)


save(mo.pdp, u.pdp, prop_eux.pdp, TOC.pdp, file = "primary-rf-analyses-20240207-10
0-iterations.RData")
```

8. Expanded analyses varying redox-sensitive predictors

Anoxic Mo (no pyrite)

Here, we generate an anoxic Mo equivalent to our primary U dataset (not restricting to anoxic samples with iron speciation). Fepy/FeHR is not included as a predictor variable.

```
Mo.anox.rf <- trace.toc.full %>%
  filter(!is.na(Mo..ppm.)) %>%
  filter(FeHR.FeT >= 0.38 | FeT.Al >= 0.53)

nrow(Mo.anox.rf)

Mo.anox.rf.results <- Monte.Carlo.rF(data = Mo.anox.rf,
                        resp.var = "Mo..ppm.",
                        vars = c("age.model",
                                 "site.type",
                                 "metamorphic.bin",
                                 "basin.type",
                                 "site.latitude",
                                 "site.longitude",
                                 "lithology.name",
                                 "environmental.bin",
                                 "Mo..ppm.",
                                 "TOC..wt..",
                                 "Al..wt.."),
                        n = n,
                        run.age.unc = TRUE,
                        run.partial.context = TRUE,
                        run.Al.impute = TRUE)

Mo.anox.rf.partial <- Mo.anox.rf.results$partial.plot.data
Mo.anox.rf.import <- Mo.anox.rf.results$importance.data

mo.no_py.pdp <- interp.pdp(data = Mo.anox.rf.partial, age.rounding.factor = 1, n =
n)
```

We combine these analyses with our primary Mo analyses for plotting.

```
mo.pdp$treatment <- "Anoxic samples + TOC + Fepy/FeHR"
mo.no_py.pdp$treatment <- "Anoxic samples + TOC"

mo.pdp.sum <- rbind(mo.pdp,
                    mo.no_py.pdp)
```

Mo - testing ensitivity to iron speciation thresholds (include FeHR/FeT and Fepy/FeHR as predictor variables, rather than using as filter cutoffs)

```
Mo.thresh.test.rf <- trace.toc.full %>%
  filter(!is.na(Mo..ppm.) & !is.na(FeHR.FeT) & !is.na(Fe.py.FeHR))

nrow(Mo.thresh.test.rf)

Mo.thresh.test.rf.results <- Monte.Carlo.rF(data = Mo.thresh.test.rf,
                        resp.var = "Mo..ppm.",
                        vars = c("age.model",
                                 "site.type",
                                 "metamorphic.bin",
                                 "basin.type",
                                 "site.latitude",
                                 "site.longitude",
                                 "lithology.name",
                                 "environmental.bin",
                                 "Mo..ppm.",
                                 "TOC..wt..",
                                 "FeHR.FeT",
                                 "Fe.py.FeHR",
                                 "Al..wt.."),
                        n = n,
                        run.age.unc = TRUE,
                        run.partial.context = TRUE,
                        run.Al.impute = TRUE)

Mo.thresh.test.rf.partial <- Mo.thresh.test.rf.results$partial.plot.data
Mo.thresh.test.rf.import <- Mo.thresh.test.rf.results$importance.data

mo.thresh.test.pdp <- interp.pdp(data = Mo.thresh.test.rf.partial, age.rounding.fa
ctor = 1, n = n)
```

We combine these analyses with our primary Mo analyses for plotting.

```
mo.pdp$treatment <- "Fe-speciation filter approach"
mo.thresh.test.pdp$treatment <- "Fe-speciation predictor approach"

mo.pdp.sum.thresh <- rbind(mo.pdp,
                    mo.thresh.test.pdp)
```

Anoxic U with pyrite

Here, we generate an anoxic U equivalent to our primary Mo dataset (restricting to anoxic samples with iron speciation). Fepy/FeHR is included as a predictor variable.

```
U.anox.py.rf <- trace.toc.full %>%
  filter(!is.na(U..ppm.) & !is.na(Fe.py.FeHR)) %>%
  filter(FeHR.FeT >= 0.38)

nrow(U.anox.py.rf)

U.anox.py.rf.results <- Monte.Carlo.rF(data = U.anox.py.rf,
                        resp.var = "U..ppm.",
                        vars = c("age.model",
                                 "site.type",
                                 "metamorphic.bin",
                                 "basin.type",
                                 "site.latitude",
                                 "site.longitude",
                                 "lithology.name",
                                 "environmental.bin",
                                 "U..ppm.",
                                 "TOC..wt..",
                                 "Fe.py.FeHR",
                                 "Al..wt.."),
                        n = n,
                        run.age.unc = TRUE,
                        run.partial.context = TRUE,
                        run.Al.impute = TRUE)

U.anox.py.rf.partial <- U.anox.py.rf.results$partial.plot.data
U.anox.py.rf.import <- U.anox.py.rf.results$importance.data

u.w_py.pdp <- interp.pdp(data = U.anox.py.rf.partial, age.rounding.factor = 1, n =
n)
```

We combine these analyses with our primary U analyses for plotting.

```
u.pdp$treatment <- "Anoxic samples + TOC"
u.w_py.pdp$treatment <- "Anoxic samples + TOC + Fepy/FeHR"

u.pdp.sum <- rbind(u.pdp,
                   u.w_py.pdp)
```

U - testing ensitivity to iron speciation thresholds (include FeHR/FeT as predictor variable, rather than using as filter cutoffs)

```
U.thresh.test.rf <- trace.toc.full %>%
  filter(!is.na(U..ppm.) & !is.na(FeHR.FeT))

nrow(U.thresh.test.rf)

U.thresh.test.rf.results <- Monte.Carlo.rF(data = U.thresh.test.rf,
                        resp.var = "U..ppm.",
                        vars = c("age.model",
                                 "site.type",
                                 "metamorphic.bin",
                                 "basin.type",
                                 "site.latitude",
                                 "site.longitude",
                                 "lithology.name",
                                 "environmental.bin",
                                 "U..ppm.",
                                 "TOC..wt..",
                                 "FeHR.FeT",
                                 "Al..wt.."),
                        n = n,
                        run.age.unc = TRUE,
                        run.partial.context = TRUE,
                        run.Al.impute = TRUE)

U.thresh.test.rf.partial <- U.thresh.test.rf.results$partial.plot.data
U.thresh.test.rf.import <- U.thresh.test.rf.results$importance.data

u.thresh.test.pdp <- interp.pdp(data = U.thresh.test.rf.partial, age.rounding.fact
or = 1, n = n)
```

We combine these analyses with our primary U analyses for plotting.

```
u.pdp$treatment <- "Fe-speciation filter approach"
u.thresh.test.pdp$treatment <- "Fe-speciation predictor approach"

u.pdp.sum.thresh <- rbind(u.pdp,
                    u.thresh.test.pdp)
```

Proportion euxinic with TOC

Note that we hide warnings here because of random forest warning about using binary variable in a regression (here we do want to do a regression to replicate the broad approach of Sperling et al. 2015, Nature).

```
Fepy.anox.w_TOC.rf <- trace.toc.full %>%
  filter(!is.na(Fe.py.FeHR) & !is.na(TOC..wt..)) %>%
  filter(FeHR.FeT >= 0.38)

nrow(Fepy.anox.w_TOC.rf)

# For the analysis of the proportion of euxinic samples, we also need to code samp
les based # upon whether they are euxinic (based on iron speciation) in a binary f
ashion.

Fepy.anox.w_TOC.rf$euxinic.Fe[Fepy.anox.w_TOC.rf$Fe.py.FeHR >= 0.7] <- 1
Fepy.anox.w_TOC.rf$euxinic.Fe[Fepy.anox.w_TOC.rf$Fe.py.FeHR < 0.7] <- 0

Fepy.anox.w_TOC.rf.results <- Monte.Carlo.rF(data = Fepy.anox.w_TOC.rf,
                        resp.var = "euxinic.Fe",
                        vars = c("age.model",
                                 "site.type",
                                 "metamorphic.bin",
                                 "basin.type",
                                 "site.latitude",
                                 "site.longitude",
                                 "lithology.name",
                                 "environmental.bin",
                                 "euxinic.Fe",
                                 "Al..wt..",
                                 "TOC..wt.."),
                        n = n,
                        run.age.unc = TRUE,
                        run.partial.context = TRUE,
                        run.Al.impute = TRUE)

Fepy.anox.w_TOC.rf.partial <- Fepy.anox.w_TOC.rf.results$partial.plot.data
Fepy.anox.w_TOC.rf.import <- Fepy.anox.w_TOC.rf.results$importance.data

prop_eux.w_TOC.pdp <- interp.pdp(data = Fepy.anox.w_TOC.rf.partial, age.rounding.f
actor = 1, n = n)
```

We combine these analyses with our primary proportion euxinic analyses for plotting.

```
prop_eux.pdp$treatment <- "Anoxic samples"
prop_eux.w_TOC.pdp$treatment <- "Anoxic samples + TOC"

prop_eux.pdp.sum <- rbind(prop_eux.pdp,
                    prop_eux.w_TOC.pdp)
```

Anoxic TOC (no pyrite)

```
TOC.anox.rf <- trace.toc.full %>%
  filter(!is.na(TOC..wt..))  %>%
  filter(FeHR.FeT >= 0.38 | FeT.Al >= 0.53)

nrow(TOC.anox.rf)

TOC.anox.rf.results <- Monte.Carlo.rF(data = TOC.anox.rf,
                        resp.var = "TOC..wt..",
                        vars = c("age.model",
                                 "site.type",
                                 "metamorphic.bin",
                                 "basin.type",
                                 "site.latitude",
                                 "site.longitude",
                                 "lithology.name",
                                 "environmental.bin",
                                 "TOC..wt..",
                                 "Al..wt.."),
                        n = n,
                        run.age.unc = TRUE,
                        run.partial.context = TRUE,
                        run.Al.impute = TRUE)

TOC.anox.rf.partial <- TOC.anox.rf.results$partial.plot.data
TOC.anox.rf.import <- TOC.anox.rf.results$importance.data

TOC.anox.pdp <- interp.pdp(data = TOC.anox.rf.partial, age.rounding.factor = 1, n
= n)
```

Anoxic TOC with pyrite

```
TOC.anox.py.rf <- trace.toc.full %>%
  filter(!is.na(TOC..wt..) & !is.na(Fe.py.FeHR))  %>%
  filter(FeHR.FeT >= 0.38)

nrow(TOC.anox.rf)

TOC.anox.py.rf.results <- Monte.Carlo.rF(data = TOC.anox.py.rf,
                        resp.var = "TOC..wt..",
                        vars = c("age.model",
                                 "site.type",
                                 "metamorphic.bin",
                                 "basin.type",
                                 "site.latitude",
                                 "site.longitude",
                                 "lithology.name",
                                 "environmental.bin",
                                 "TOC..wt..",
                                 "Al..wt..",
                                 "Fe.py.FeHR"),
                        n = n,
                        run.age.unc = TRUE,
                        run.partial.context = TRUE,
                        run.Al.impute = TRUE)

TOC.anox.py.rf.partial <- TOC.anox.py.rf.results$partial.plot.data
TOC.anox.py.rf.import <- TOC.anox.py.rf.results$importance.data

TOC.anox.py.pdp <- interp.pdp(data = TOC.anox.py.rf.partial, age.rounding.factor =
1, n = n)
```

We combine these analyses with our primary TOC analyses for plotting.

```
TOC.pdp$treatment <- "All samples"
TOC.anox.pdp$treatment <- "Anoxic samples"
TOC.anox.py.pdp$treatment <- "Anoxic samples + Fepy/FeHR"

TOC.pdp.sum <- rbind(TOC.pdp,
                     TOC.anox.pdp,
                     TOC.anox.py.pdp)
```

Summary plotting of analyses including varying redox treatments

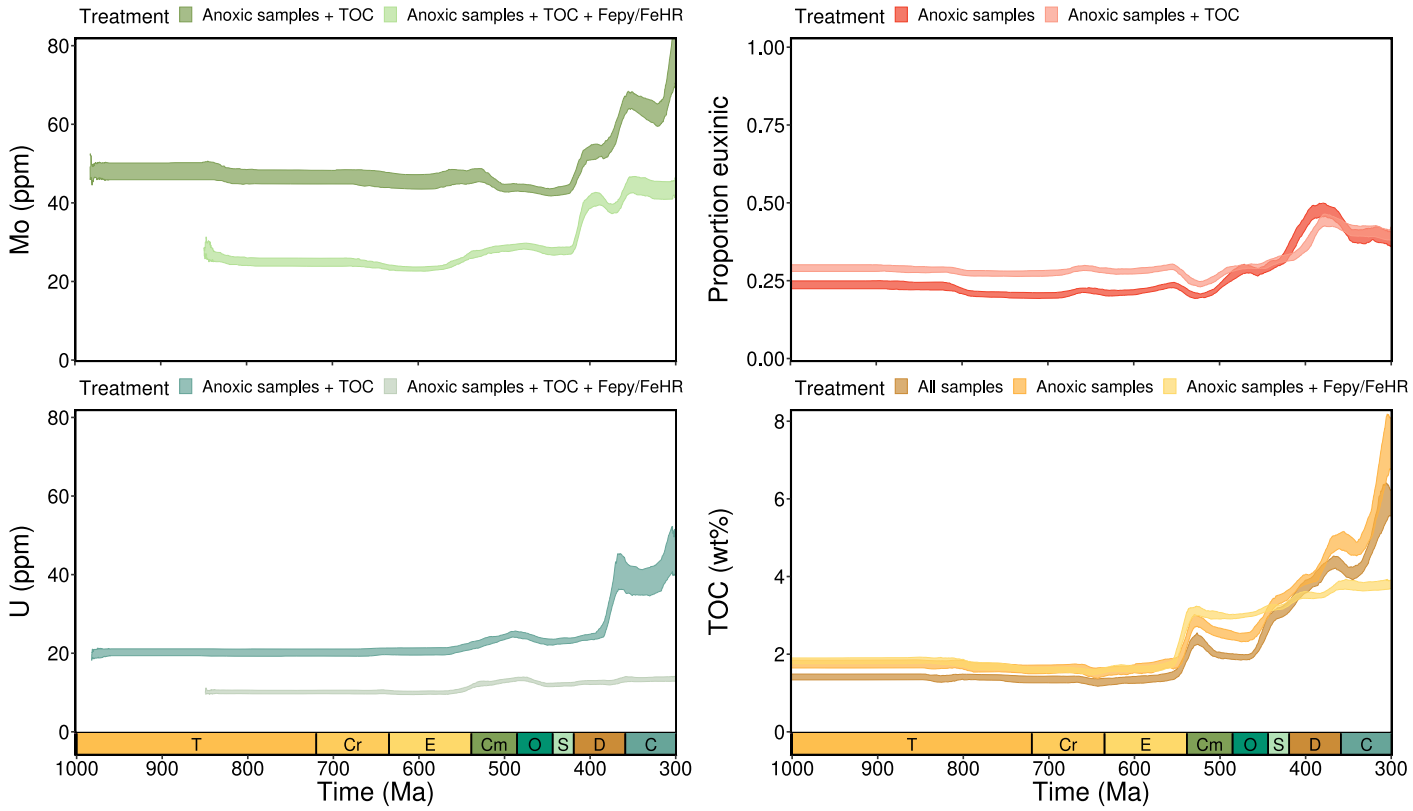
In these summary plots we just plot the interquartile ranges for our analyses to ease the comparison of trends in central tendancy.

```
Mo.plot.for.redox.var.sum <- ggplot(mo.pdp.sum, aes(x=Age))+
  #annotate(geom="rect", xmax=Inf, xmin=541, ymin=-Inf, ymax=Inf, fill="#C9DEE8",
alpha=0.2)+
  #annotate(geom="rect", xmax=541, xmin=418, ymin=-Inf, ymax=Inf, fill="#9DC2D6",
alpha=0.4)+
```

```
  #annotate(geom="rect", xmax=418, xmin=-Inf, ymin=-Inf, ymax=Inf, fill="#619FBB",
alpha=0.4)+
  geom_ribbon(aes(ymin = perc.25, ymax = perc.75, fill=treatment, color=treatment)
, alpha=.7, size=.5)+
  scale_fill_manual(values = c(rgb(127, 160, 86, maxColorValue = 255),
                               rgb(179, 224, 149, maxColorValue = 255)),
                    name = "Treatment")+
  scale_color_manual(values = c(rgb(127, 160, 86, maxColorValue = 255),
                               rgb(179, 224, 149, maxColorValue = 255)),
                    name = "Treatment")+
  theme_bw()+
  coord_cartesian(xlim=rev(c(298.9,1000)), ylim=c(-.4,82),expand=FALSE)+
  scale_x_reverse(breaks=c(300,400,254,600,700,800,900,1000))+
  ylab("Mo (ppm)")+xlab("Time (Ma)")+
  theme(plot.margin = ggplot2::margin(.1,1,.3,1,"cm"),panel.border = element_rect(
fill=NA,color="black", size=2,linetype="solid"),
        axis.ticks = element_line(size=1),
        axis.line = element_line(lineend = 'square'),
        axis.title = element_text(size=34),
        axis.text = element_text( size=26, color="black"),
        legend.title = element_text(size=26),
        legend.text = element_text( size=22),
        axis.ticks.length = unit(5, "points"),
        legend.position="top",         legend.justification = c(0, 0),
        axis.title.x = element_blank(),
        axis.text.x = element_blank(),
        axis.title.y = element_text(vjust=3),
        panel.grid.major = element_blank(),panel.grid.minor = element_blank())

U.plot.for.redox.var.sum <- ggplot(u.pdp.sum, aes(x=Age))+
  #annotate(geom="rect", xmax=Inf, xmin=541, ymin=-Inf, ymax=Inf, fill="#C9DEE8",
alpha=0.2)+
  #annotate(geom="rect", xmax=541, xmin=418, ymin=-Inf, ymax=Inf, fill="#9DC2D6",
alpha=0.4)+
  #annotate(geom="rect", xmax=418, xmin=-Inf, ymin=-Inf, ymax=Inf, fill="#619FBB",
alpha=0.4)+
  geom_ribbon(aes(ymin = perc.25, ymax = perc.75, fill=treatment, color=treatment)
, alpha=.7, size=.5)+
  scale_fill_manual(values = c(rgb(103, 165, 153, maxColorValue = 255),
                               rgb(191, 208, 186, maxColorValue = 255)),
                    name = "Treatment")+
  scale_color_manual(values = c(rgb(103, 165, 153, maxColorValue = 255),
                               rgb(191, 208, 186, maxColorValue = 255)),
                    name = "Treatment")+
  theme_bw()+
  coord_geo(xlim=rev(c(298.9,1002)), expand=FALSE, ylim=c(-.4,82),
            pos = as.list(rep("bottom", 1)),
            abbrv=list(TRUE),
            dat = list(periods.edit),
            height = list(unit(2, "lines")),
            bord=list(c("left", "bottom", "right")), lwd=as.list(c(1)), size=8)+
  scale_x_reverse(breaks=c(300,400,500,600,700,800,900,1000))+
  ylab("U (ppm)")+xlab("Time (Ma)")+
  theme(plot.margin = ggplot2::margin(.1,1,.3,1,"cm"),panel.border = element_rect(
```

```
fill=NA,color="black", size=2,linetype="solid"),
        axis.ticks = element_line(size=1),
        axis.line = element_line(lineend = 'square'),
        axis.title = element_text(size=34),
        axis.text = element_text( size=26, color="black"),
        legend.title = element_text(size=26),
        legend.text = element_text( size=22),
        axis.ticks.length = unit(5, "points"),
        legend.position="top",
        legend.justification = c(0, 0),
        axis.title.y = element_text(vjust=3),
        panel.grid.major = element_blank(),panel.grid.minor = element_blank())

prop_eux.plot.for.redox.var.sum <- ggplot(prop_eux.pdp.sum, aes(x=Age))+
  #annotate(geom="rect", xmax=Inf, xmin=541, ymin=-Inf, ymax=Inf, fill="#C9DEE8",
alpha=0.2)+
  #annotate(geom="rect", xmax=541, xmin=418, ymin=-Inf, ymax=Inf, fill="#9DC2D6",
alpha=0.4)+
  #annotate(geom="rect", xmax=418, xmin=-Inf, ymin=-Inf, ymax=Inf, fill="#619FBB",
alpha=0.4)+
  geom_ribbon(aes(ymin = perc.25, ymax = perc.75, fill=treatment, color=treatment)
, alpha=.7, size=.5)+
  scale_fill_manual(values = c(rgb(240, 64, 40, maxColorValue = 255),
                               rgb(251, 154, 133, maxColorValue = 255)),
                    name = "Treatment")+
  scale_color_manual(values = c(rgb(240, 64, 40, maxColorValue = 255),
                               rgb(251, 154, 133, maxColorValue = 255)),
                    name = "Treatment")+
  theme_bw()+
  coord_cartesian(xlim=rev(c(298.9,1000)), ylim=c(-.01,1.03),expand=FALSE)+
  scale_x_reverse(breaks=c(300,400,500,600,700,800,900,1000))+
  ylab("Proportion euxinic")+xlab("Time (Ma)")+
  theme(plot.margin = ggplot2::margin(.1,1,.3,1,"cm"),panel.border = element_rect(
fill=NA,color="black", size=2,linetype="solid"),
        axis.ticks = element_line(size=1),
        axis.line = element_line(lineend = 'square'),
        axis.title = element_text(size=34),
        axis.text = element_text( size=26, color="black"),
        legend.title = element_text(size=26),
        legend.text = element_text( size=22),
        axis.ticks.length = unit(5, "points"),
        legend.position="top",         legend.justification = c(0, 0),
        axis.title.x = element_blank(),
        axis.text.x = element_blank(),
        axis.title.y = element_text(vjust=3),
        panel.grid.major = element_blank(),panel.grid.minor = element_blank())

TOC.plot.for.redox.var.sum <- ggplot(TOC.pdp.sum, aes(x=Age))+
  #annotate(geom="rect", xmax=Inf, xmin=541, ymin=-Inf, ymax=Inf, fill="#C9DEE8",
alpha=0.2)+
  #annotate(geom="rect", xmax=541, xmin=418, ymin=-Inf, ymax=Inf, fill="#9DC2D6",
alpha=0.4)+
  #annotate(geom="rect", xmax=418, xmin=-Inf, ymin=-Inf, ymax=Inf, fill="#619FBB",
alpha=0.4)+
```

```
  geom_ribbon(aes(ymin = perc.25, ymax = perc.75, fill=treatment, color=treatment)
, alpha=.7, size=.5)+
  scale_fill_manual(values = c(rgb(203, 140, 55, maxColorValue = 255),
                               rgb(254, 179, 66, maxColorValue = 255),
                               rgb(254, 217, 106, maxColorValue = 255)),
                    name = "Treatment")+
  scale_color_manual(values = c(rgb(203, 140, 55, maxColorValue = 255),
                               rgb(254, 179, 66, maxColorValue = 255),
                               rgb(254, 217, 106, maxColorValue = 255)),
                    name = "Treatment")+
  theme_bw()+
  coord_geo(xlim=rev(c(298.9,1002)), expand=FALSE, ylim=c(-0.04,8.3),
            pos = as.list(rep("bottom", 1)),
            abbrv=list(TRUE),
            dat = list(periods.edit),
            height = list(unit(2, "lines")),
            bord=list(c("left", "bottom", "right")), lwd=as.list(c(1)), size=8)+
  scale_x_reverse(breaks=c(300,400,500,600,700,800,900,1000))+
  ylab("TOC (wt%)")+xlab("Time (Ma)")+
  theme(plot.margin = ggplot2::margin(.1,1,.3,1,"cm"),panel.border = element_rect(
fill=NA,color="black", size=2,linetype="solid"),
        axis.ticks = element_line(size=1),
        axis.line = element_line(lineend = 'square'),
        axis.title = element_text(size=34),
        axis.text = element_text( size=26, color="black"),
        legend.title = element_text(size=26),
        legend.text = element_text( size=22),
        axis.ticks.length = unit(5, "points"),
        legend.position="top",
        legend.justification = c(0, 0),
        axis.title.y = element_text(vjust=3),
        panel.grid.major = element_blank(),panel.grid.minor = element_blank())

summary.redox.var.plot <- ggarrange2(Mo.plot.for.redox.var.sum, prop_eux.plot.for.
redox.var.sum, U.plot.for.redox.var.sum, TOC.plot.for.redox.var.sum, ncol=2, heigh
ts=c(1,1))
```




```
ggsave("Figure Sx Partial Dependence Plot (varying redox treatments) 20240207 100
iterations.pdf", summary.redox.var.plot, height=16, width=26)
```

Also compile Fe spectiation threshold vs predictor test plots.

```
Mo.plot.for.thresh.sum <- ggplot(mo.pdp.sum.thresh, aes(x=Age))+
  #annotate(geom="rect", xmax=Inf, xmin=541, ymin=-Inf, ymax=Inf, fill="#C9DEE8",
alpha=0.2)+
  #annotate(geom="rect", xmax=541, xmin=418, ymin=-Inf, ymax=Inf, fill="#9DC2D6",
alpha=0.4)+
  #annotate(geom="rect", xmax=418, xmin=-Inf, ymin=-Inf, ymax=Inf, fill="#619FBB",
alpha=0.4)+
  geom_ribbon(aes(ymin = perc.25, ymax = perc.75, fill=treatment, color=treatment)
, alpha=.7, size=.5)+
  scale_fill_manual(values = c(rgb(127, 160, 86, maxColorValue = 255),
                               rgb(179, 224, 149, maxColorValue = 255)),
                    name = "Treatment")+
  scale_color_manual(values = c(rgb(127, 160, 86, maxColorValue = 255),
                               rgb(179, 224, 149, maxColorValue = 255)),
                    name = "Treatment")+
  theme_bw()+
  coord_cartesian(xlim=rev(c(298.9,1000)), ylim=c(-.4,82),expand=FALSE)+
  scale_x_reverse(breaks=c(300,400,254,600,700,800,900,1000))+
  ylab("Mo (ppm)")+xlab("Time (Ma)")+
  theme(plot.margin = ggplot2::margin(.1,1,.3,1,"cm"),panel.border = element_rect(
fill=NA,color="black", size=2,linetype="solid"),
        axis.ticks = element_line(size=1),
        axis.line = element_line(lineend = 'square'),
        axis.title = element_text(size=34),
        axis.text = element_text( size=26, color="black"),
        legend.title = element_text(size=26),
```

```
        legend.text = element_text( size=22),
        axis.ticks.length = unit(5, "points"),
        legend.position="top",         legend.justification = c(0, 0),
        axis.title.x = element_blank(),
        axis.text.x = element_blank(),
        axis.title.y = element_text(vjust=3),
        panel.grid.major = element_blank(),panel.grid.minor = element_blank())

U.plot.for.thresh.sum <- ggplot(u.pdp.sum.thresh, aes(x=Age))+
  #annotate(geom="rect", xmax=Inf, xmin=541, ymin=-Inf, ymax=Inf, fill="#C9DEE8",
alpha=0.2)+
  #annotate(geom="rect", xmax=541, xmin=418, ymin=-Inf, ymax=Inf, fill="#9DC2D6",
alpha=0.4)+
  #annotate(geom="rect", xmax=418, xmin=-Inf, ymin=-Inf, ymax=Inf, fill="#619FBB",
alpha=0.4)+
  geom_ribbon(aes(ymin = perc.25, ymax = perc.75, fill=treatment, color=treatment)
, alpha=.7, size=.5)+
  scale_fill_manual(values = c(rgb(103, 165, 153, maxColorValue = 255),
                               rgb(191, 208, 186, maxColorValue = 255)),
                    name = "Treatment")+
  scale_color_manual(values = c(rgb(103, 165, 153, maxColorValue = 255),
                               rgb(191, 208, 186, maxColorValue = 255)),
                    name = "Treatment")+
  theme_bw()+
  coord_geo(xlim=rev(c(298.9,1002)), expand=FALSE, ylim=c(-.4,82),
            pos = as.list(rep("bottom", 1)),
            abbrv=list(TRUE),
            dat = list(periods.edit),
            height = list(unit(2, "lines")),
            bord=list(c("left", "bottom", "right")), lwd=as.list(c(1)), size=8)+
  scale_x_reverse(breaks=c(300,400,500,600,700,800,900,1000))+
  ylab("U (ppm)")+xlab("Time (Ma)")+
  theme(plot.margin = ggplot2::margin(.1,1,.3,1,"cm"),panel.border = element_rect(
fill=NA,color="black", size=2,linetype="solid"),
        axis.ticks = element_line(size=1),
        axis.line = element_line(lineend = 'square'),
        axis.title = element_text(size=34),
        axis.text = element_text( size=26, color="black"),
        legend.title = element_text(size=26),
        legend.text = element_text( size=22),
        axis.ticks.length = unit(5, "points"),
        legend.position="top",
        legend.justification = c(0, 0),
        axis.title.y = element_text(vjust=3),
        panel.grid.major = element_blank(),panel.grid.minor = element_blank())



Mo.plot.for.thresh.solo <- ggplot(mo.pdp.sum.thresh, aes(x=Age))+
  #annotate(geom="rect", xmax=Inf, xmin=541, ymin=-Inf, ymax=Inf, fill="#C9DEE8",
alpha=0.2)+
  #annotate(geom="rect", xmax=541, xmin=418, ymin=-Inf, ymax=Inf, fill="#9DC2D6",
alpha=0.4)+
  #annotate(geom="rect", xmax=418, xmin=-Inf, ymin=-Inf, ymax=Inf, fill="#619FBB",
```

```
alpha=0.4)+
  geom_ribbon(aes(ymin = perc.25, ymax = perc.75, fill=treatment, color=treatment)
, alpha=.7, size=.5)+
  scale_fill_manual(values = c(rgb(127, 160, 86, maxColorValue = 255),
                               rgb(179, 224, 149, maxColorValue = 255)),
                    name = "Treatment")+
  scale_color_manual(values = c(rgb(127, 160, 86, maxColorValue = 255),
                               rgb(179, 224, 149, maxColorValue = 255)),
                    name = "Treatment")+
  theme_bw()+
  coord_geo(xlim=rev(c(298.9,1002)), expand=FALSE, ylim=c(-.4,82),
            pos = as.list(rep("bottom", 1)),
            abbrv=list(TRUE),
            dat = list(periods.edit),
            height = list(unit(2, "lines")),
            bord=list(c("left", "bottom", "right")), lwd=as.list(c(1)), size=8)+
  scale_x_reverse(breaks=c(300,400,500,600,700,800,900,1000))+
  ylab("Mo (ppm)")+xlab("Time (Ma)")+
  theme(plot.margin = ggplot2::margin(.1,1,.3,1,"cm"),panel.border = element_rect(
fill=NA,color="black", size=2,linetype="solid"),
        axis.ticks = element_line(size=1),
        axis.line = element_line(lineend = 'square'),
        axis.title = element_text(size=34),
        axis.text = element_text( size=26, color="black"),
        legend.title = element_text(size=26),
        legend.text = element_text( size=22),
        axis.ticks.length = unit(5, "points"),
        legend.position="top",
        legend.justification = c(0, 0),
        axis.title.y = element_text(vjust=3),
        panel.grid.major = element_blank(),panel.grid.minor = element_blank())


summary.thresh.plot <- ggarrange2(Mo.plot.for.thresh.sum, U.plot.for.thresh.sum, n
col=1, heights=c(1,1))
```

```
ggsave("Figure Sx Partial Dependence Plot (testing Fe-speciation thresholds) 20240
207 100 iterations.pdf", summary.thresh.plot, height=15, width=13)

ggsave("Figure Sx Partial Dependence Plot (testing Fe-speciation thresholds) 20240
207 100 iterations Mo only.pdf", Mo.plot.for.thresh.solo, height=8.6, width=13)
```

9. Primary analyses without Al

We also re-run our analyses without incorporating [Al] as a broad proxy for detrital input.

Molybdenum

```
Mo.anox.py.no_Al.rf.results <- Monte.Carlo.rF(data = Mo.anox.py.rf,
                        resp.var = "Mo..ppm.",
                        vars = c("age.model",
                                 "site.type",
                                 "metamorphic.bin",
                                 "basin.type",
                                 "site.latitude",
                                 "site.longitude",
                                 "lithology.name",
                                 "environmental.bin",
                                 "Mo..ppm.",
                                 "TOC..wt..",
                                 "Fe.py.FeHR"),
                        n = n,
                        run.age.unc = TRUE,
                        run.partial.context = TRUE,
                        run.Al.impute = FALSE)

Mo.anox.py.no_Al.rf.partial <- Mo.anox.py.no_Al.rf.results$partial.plot.data
Mo.anox.py.no_Al.rf.import <- Mo.anox.py.no_Al.rf.results$importance.data

mo.no_Al.pdp <- interp.pdp(data = Mo.anox.py.no_Al.rf.partial, age.rounding.factor
= 1, n = n)
```

Uranium

```
U.anox.no_Al.rf.results <- Monte.Carlo.rF(data = U.anox.rf,
                        resp.var = "U..ppm.",
                        vars = c("age.model",
                                 "site.type",
                                 "metamorphic.bin",
                                 "basin.type",
                                 "site.latitude",
                                 "site.longitude",
                                 "lithology.name",
                                 "environmental.bin",
                                 "U..ppm.",
                                 "TOC..wt.."),
                        n = n,
                        run.age.unc = TRUE,
                        run.partial.context = TRUE,
                        run.Al.impute = FALSE)

U.anox.no_Al.rf.partial <- U.anox.no_Al.rf.results$partial.plot.data
U.anox.no_Al.rf.import <- U.anox.no_Al.rf.results$importance.data

u.no_Al.pdp <- interp.pdp(data = U.anox.no_Al.rf.partial, age.rounding.factor = 1,
n = n)
```

Proportion euxinic

Note that we hide warnings here because of random forest warning about using binary variable in a regression (here we do want to do a regression to replicate the broad approach of Sperling et al. 2015, Nature).

```
Fepy.anox.no_Al.rf.results <- Monte.Carlo.rF(data = Fepy.anox.rf,
                        resp.var = "euxinic.Fe",
                        vars = c("age.model",
                                 "site.type",
                                 "metamorphic.bin",
                                 "basin.type",
                                 "site.latitude",
                                 "site.longitude",
                                 "lithology.name",
                                 "environmental.bin",
                                 "euxinic.Fe"),
                        n = n,
                        run.age.unc = TRUE,
                        run.partial.context = TRUE,
                        run.Al.impute = FALSE)

Fepy.anox.no_Al.rf.partial <- Fepy.anox.no_Al.rf.results$partial.plot.data
Fepy.anox.no_Al.rf.import <- Fepy.anox.no_Al.rf.results$importance.data

prop_eux.no_Al.pdp <- interp.pdp(data = Fepy.anox.no_Al.rf.partial, age.rounding.f
actor = 1, n = n)
```

Total Organic Carbon

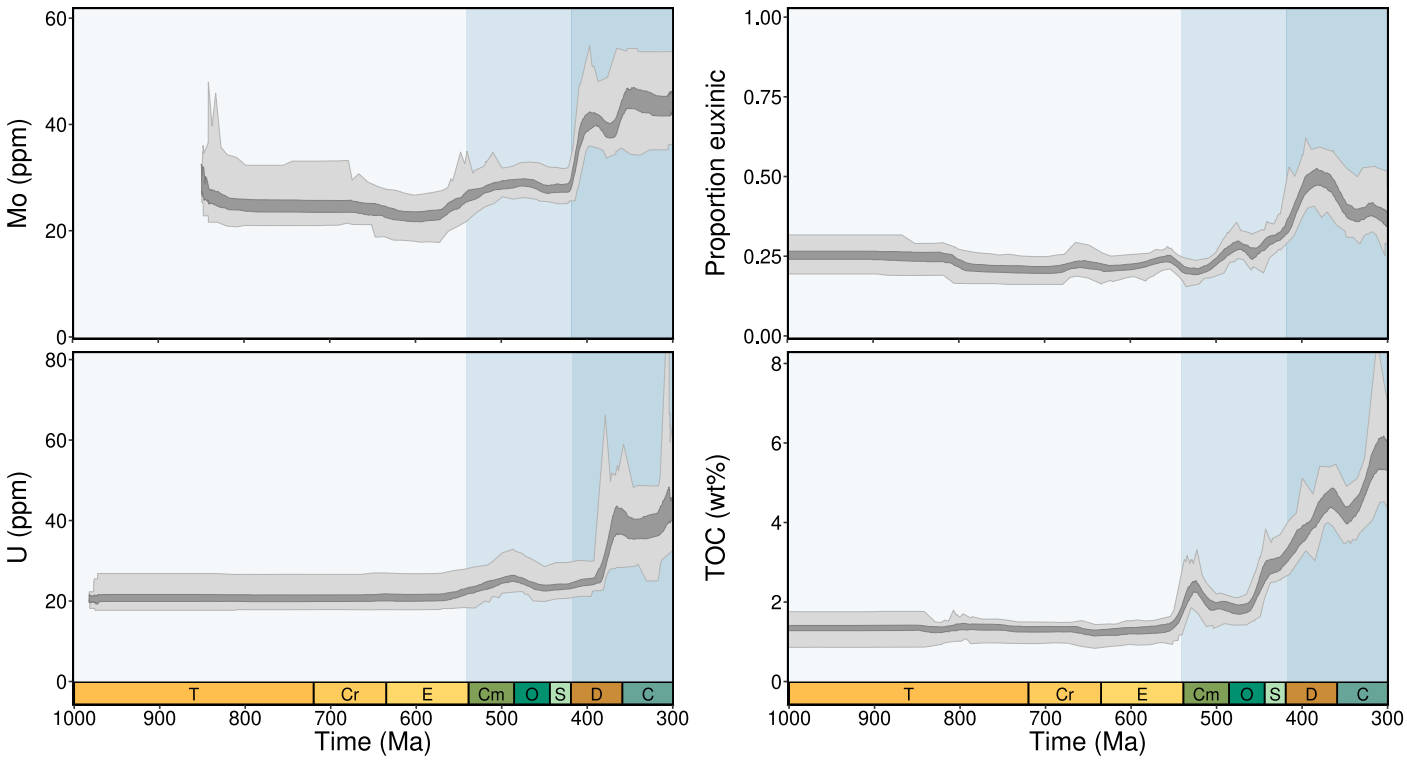
```
TOC.all.no_Al.rf.results <- Monte.Carlo.rF(data = TOC.all.rf,
                        resp.var = "TOC..wt..",
                        vars = c("age.model",
                                 "site.type",
                                 "metamorphic.bin",
                                 "basin.type",
                                 "site.latitude",
                                 "site.longitude",
                                 "lithology.name",
                                 "environmental.bin",
                                 "TOC..wt.."),
                        n = n,
                        run.age.unc = TRUE,
                        run.partial.context = TRUE,
                        run.Al.impute = FALSE)

TOC.all.no_Al.rf.partial <- TOC.all.no_Al.rf.results$partial.plot.data
TOC.all.no_Al.rf.import <- TOC.all.no_Al.rf.results$importance.data

TOC.no_Al.pdp <- interp.pdp(data = TOC.all.no_Al.rf.partial, age.rounding.factor =
1, n = n)
```

Combine to generate summary plot

```
Mo.no_Al.plot.for.sum <- ggplot(mo.no_Al.pdp, aes(x=Age))+
  annotate(geom="rect", xmax=Inf, xmin=541, ymin=-Inf, ymax=Inf, fill="#C9DEE8", a
lpha=0.2)+
  annotate(geom="rect", xmax=541, xmin=418, ymin=-Inf, ymax=Inf, fill="#9DC2D6", a
lpha=0.4)+
  annotate(geom="rect", xmax=418, xmin=-Inf, ymin=-Inf, ymax=Inf, fill="#619FBB",
alpha=0.4)+
  geom_ribbon(aes(ymin = min, ymax = max), alpha=1, fill="grey85", color="grey70",
size=.3)+
  geom_ribbon(aes(ymin = perc.25, ymax = perc.75), alpha=1, fill="grey60", color="
grey50", size=.5)+
  theme_bw()+
  coord_cartesian(xlim=rev(c(298.9,1000)), ylim=c(-.4,62),expand=FALSE)+
  scale_x_reverse(breaks=c(300,400,500,600,700,800,900,1000))+
  ylab("Mo (ppm)")+xlab("Time (Ma)")+
  theme(plot.margin = ggplot2::margin(.1,1,.3,1,"cm"),panel.border = element_rect(
fill=NA,color="black", size=2,linetype="solid"),
        axis.ticks = element_line(size=1),
        axis.line = element_line(lineend = 'square'),
        axis.title = element_text(size=34),
        axis.text = element_text( size=26, color="black"),
        legend.title = element_text(size=20),
        legend.text = element_text( size=16),
        axis.ticks.length = unit(5, "points"),
        legend.position="top",         legend.justification = c(0, 0),
        axis.title.x = element_blank(),
        axis.text.x = element_blank(),
        axis.title.y = element_text(vjust=3),
        panel.grid.major = element_blank(),panel.grid.minor = element_blank())

U.no_Al.plot.for.sum <- ggplot(u.no_Al.pdp, aes(x=Age))+
  annotate(geom="rect", xmax=Inf, xmin=541, ymin=-Inf, ymax=Inf, fill="#C9DEE8", a
lpha=0.2)+
  annotate(geom="rect", xmax=541, xmin=418, ymin=-Inf, ymax=Inf, fill="#9DC2D6", a
lpha=0.4)+
  annotate(geom="rect", xmax=418, xmin=-Inf, ymin=-Inf, ymax=Inf, fill="#619FBB",
alpha=0.4)+
  geom_ribbon(aes(ymin = min, ymax = max), alpha=1, fill="grey85", color="grey70",
size=.3)+
  geom_ribbon(aes(ymin = perc.25, ymax = perc.75), alpha=1, fill="grey60", color="
grey50", size=.5)+
  theme_bw()+
  coord_geo(xlim=rev(c(298.9,1002)), expand=FALSE, ylim=c(-.4,82),
            pos = as.list(rep("bottom", 1)),
            abbrv=list(TRUE),
            dat = list(periods.edit),
            height = list(unit(2, "lines")),
            bord=list(c("left", "bottom", "right")), lwd=as.list(c(1)), size=8)+
  scale_x_reverse(breaks=c(300,400,500,600,700,800,900,1000))+
  ylab("U (ppm)")+xlab("Time (Ma)")+
  theme(plot.margin = ggplot2::margin(.1,1,.3,1,"cm"),panel.border = element_rect(
fill=NA,color="black", size=2,linetype="solid"),
        axis.ticks = element_line(size=1),
```

```
        axis.line = element_line(lineend = 'square'),
        axis.title = element_text(size=34),
        axis.text = element_text( size=26, color="black"),
        legend.title = element_text(size=20),
        legend.text = element_text( size=16),
        axis.ticks.length = unit(5, "points"),
        legend.position="top",
        legend.justification = c(0, 0),
        axis.title.y = element_text(vjust=3),
        panel.grid.major = element_blank(),panel.grid.minor = element_blank())

prop_eux.no_Al.plot.for.sum <- ggplot(prop_eux.no_Al.pdp, aes(x=Age))+
  annotate(geom="rect", xmax=Inf, xmin=541, ymin=-Inf, ymax=Inf, fill="#C9DEE8", a
lpha=0.2)+
  annotate(geom="rect", xmax=541, xmin=418, ymin=-Inf, ymax=Inf, fill="#9DC2D6", a
lpha=0.4)+
  annotate(geom="rect", xmax=418, xmin=-Inf, ymin=-Inf, ymax=Inf, fill="#619FBB",
alpha=0.4)+
  geom_ribbon(aes(ymin = min, ymax = max), alpha=1, fill="grey85", color="grey70",
size=.3)+
  geom_ribbon(aes(ymin = perc.25, ymax = perc.75), alpha=1, fill="grey60", color="
grey50", size=.5)+
  theme_bw()+
  coord_cartesian(xlim=rev(c(298.9,1000)), ylim=c(-.01,1.03),expand=FALSE)+
  scale_x_reverse(breaks=c(300,400,500,600,700,800,900,1000))+
  ylab("Proportion euxinic")+xlab("Time (Ma)")+
  theme(plot.margin = ggplot2::margin(.1,1,.3,1,"cm"),panel.border = element_rect(
fill=NA,color="black", size=2,linetype="solid"),
        axis.ticks = element_line(size=1),
        axis.line = element_line(lineend = 'square'),
        axis.title = element_text(size=34),
        axis.text = element_text( size=26, color="black"),
        legend.title = element_text(size=20),
        legend.text = element_text( size=16),
        axis.ticks.length = unit(5, "points"),
        legend.position="top",         legend.justification = c(0, 0),
        axis.title.x = element_blank(),
        axis.text.x = element_blank(),
        axis.title.y = element_text(vjust=3),
        panel.grid.major = element_blank(),panel.grid.minor = element_blank())

TOC.no_Al.plot.for.sum <- ggplot(TOC.no_Al.pdp, aes(x=Age))+
  annotate(geom="rect", xmax=Inf, xmin=541, ymin=-Inf, ymax=Inf, fill="#C9DEE8", a
lpha=0.2)+
  annotate(geom="rect", xmax=541, xmin=418, ymin=-Inf, ymax=Inf, fill="#9DC2D6", a
lpha=0.4)+
  annotate(geom="rect", xmax=418, xmin=-Inf, ymin=-Inf, ymax=Inf, fill="#619FBB",
alpha=0.4)+
  geom_ribbon(aes(ymin = min, ymax = max), alpha=1, fill="grey85", color="grey70",
size=.3)+
  geom_ribbon(aes(ymin = perc.25, ymax = perc.75), alpha=1, fill="grey60", color="
grey50", size=.5)+
  theme_bw()+
  coord_geo(xlim=rev(c(298.9,1002)), expand=FALSE, ylim=c(-0.04,8.3),
```

```
            pos = as.list(rep("bottom", 1)),
            abbrv=list(TRUE),
            dat = list(periods.edit),
            height = list(unit(2, "lines")),
            bord=list(c("left", "bottom", "right")), lwd=as.list(c(1)), size=8)+
  scale_x_reverse(breaks=c(300,400,500,600,700,800,900,1000))+
  ylab("TOC (wt%)")+xlab("Time (Ma)")+
  theme(plot.margin = ggplot2::margin(.1,1,.3,1,"cm"),panel.border = element_rect(
fill=NA,color="black", size=2,linetype="solid"),
        axis.ticks = element_line(size=1),
        axis.line = element_line(lineend = 'square'),
        axis.title = element_text(size=34),
        axis.text = element_text( size=26, color="black"),
        legend.title = element_text(size=20),
        legend.text = element_text( size=16),
        axis.ticks.length = unit(5, "points"),
        legend.position="top",
        legend.justification = c(0, 0),
        axis.title.y = element_text(vjust=3),
        panel.grid.major = element_blank(),panel.grid.minor = element_blank())

summary.plot <- ggarrange2(Mo.no_Al.plot.for.sum, prop_eux.no_Al.plot.for.sum, U.n
o_Al.plot.for.sum, TOC.no_Al.plot.for.sum, ncol=2, heights=c(1,1))
```




```
ggsave("Figure Sx Partial Dependence Plot (no Al) with shading 20240207 100 iterat
ions.pdf", summary.plot, height=14, width=26)

save(mo.pdp.sum, u.pdp.sum, prop_eux.pdp.sum, TOC.pdp.sum, mo.pdp.sum.thresh, mo.n
o_Al.pdp, u.no_Al.pdp, prop_eux.no_Al.pdp, TOC.no_Al.pdp, file = "supplementary-rf
-analyses-20240207-100-iterations.RData")
```

10. Variable importance plots

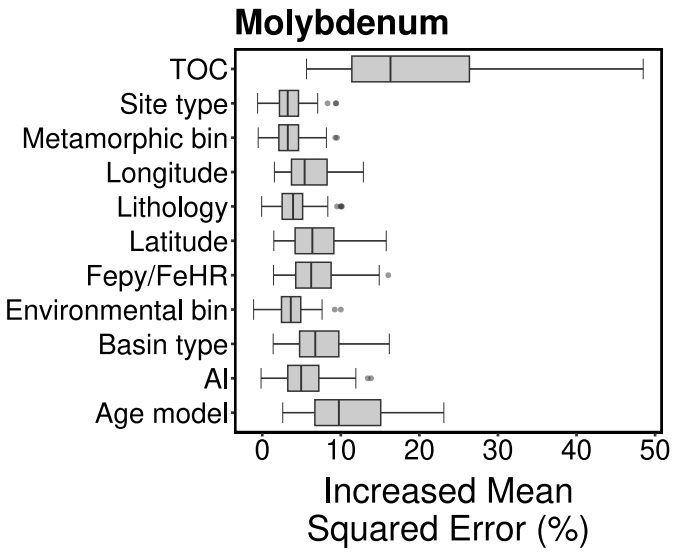
We will summarize the variable importance of the predictor variables in our primary Monte Carlo random forest analyses, using box and whisker plots to illustrate the distributions generated by our Monte Carlo approach.

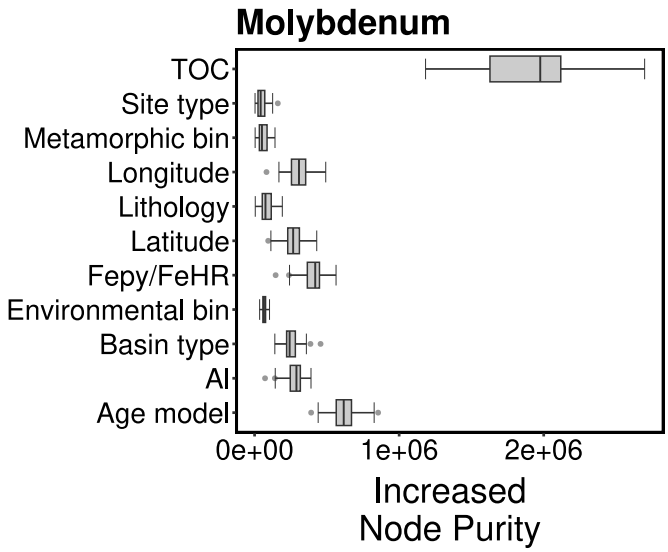
```
Mo.MSE.plot <- ggplot(Mo.anox.py.rf.import, aes(x = Variable, y = IncMSE))+stat_bo
xplot(geom ='errorbar', width = 0.6, size=0.6, color="grey20", lwd=.1)+ geom_boxpl
ot(outlier.alpha = .5, outlier.shape=16, outlier.size=2, outlier.color="grey20", c
olor="grey20", lwd=0.6, fill="grey80", fatten = 1.3)+coord_flip()+
  theme_bw()+
  ylab("Increased Mean\nSquared Error (%)")+
  ggtitle("Molybdenum")+
  theme(panel.border = element_rect(fill=NA,color="black", size=2,linetype="solid"
),
        axis.ticks = element_line(size=1.1),
        axis.title = element_text(size=30),
        axis.text =  element_text(size=24, colour="black"),
        plot.title = element_text(size=30, face = "bold"),
        plot.margin = ggplot2::margin(10,30,10,10),
        legend.position="none",
        axis.title.y = element_blank(),
        axis.title.x = element_text(margin = ggplot2::margin(t = 10, r = 0, b = 0,
l = 0)),
        panel.grid.major = element_blank(),panel.grid.minor = element_blank())

Mo.Node.plot <- ggplot(Mo.anox.py.rf.import, aes(x = Variable, y = IncNodePurity))
+stat_boxplot(geom ='errorbar', width = 0.6, size=0.6, color="grey20", lwd=.1)+ ge
om_boxplot(outlier.alpha = .5, outlier.shape=16, outlier.size=2, outlier.color="gr
ey20", color="grey20", lwd=0.6, fill="grey80", fatten = 1.3)+coord_flip()+
  theme_bw()+
  ylab("Increased\nNode Purity")+
  ggtitle("Molybdenum")+
  theme(panel.border = element_rect(fill=NA,color="black", size=2,linetype="solid"
),
        axis.ticks = element_line(size=1.1),
        axis.title = element_text(size=30),
        axis.text =  element_text(size=24, colour="black"),
        plot.title = element_text(size=30, face = "bold"),
        plot.margin = ggplot2::margin(10,30,10,10),
        legend.position="none",
        axis.title.y = element_blank(),
        axis.title.x = element_text(margin = ggplot2::margin(t = 10, r = 0, b = 0,
l = 0)),
        panel.grid.major = element_blank(),panel.grid.minor = element_blank())

U.MSE.plot <- ggplot(U.anox.rf.import, aes(x = Variable, y = IncMSE))+stat_boxplot
(geom ='errorbar', width = 0.6, size=0.6, color="grey20", lwd=.1)+ geom_boxplot(ou
tlier.alpha = .5, outlier.shape=16, outlier.size=2, outlier.color="grey20", color=
"grey20", lwd=0.6, fill="grey80", fatten = 1.3)+coord_flip()+
  theme_bw()+
  ylab("Increased Mean\nSquared Error (%)")+
  ggtitle("Uranium")+
```

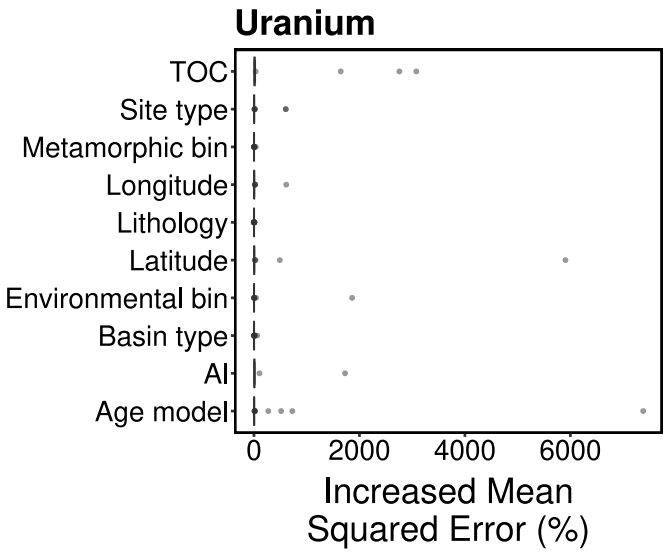
```
  theme(panel.border = element_rect(fill=NA,color="black", size=2,linetype="solid"
),
        axis.ticks = element_line(size=1.1),
        axis.title = element_text(size=30),
        axis.text =  element_text(size=24, colour="black"),
        plot.title = element_text(size=30, face = "bold"),
        plot.margin = ggplot2::margin(10,30,10,10),
        legend.position="none",
        axis.title.y = element_blank(),
        axis.title.x = element_text(margin = ggplot2::margin(t = 10, r = 0, b = 0,
l = 0)),
        panel.grid.major = element_blank(),panel.grid.minor = element_blank())

U.Node.plot <- ggplot(U.anox.rf.import, aes(x = Variable, y = IncNodePurity))+stat
_boxplot(geom ='errorbar', width = 0.6, size=0.6, color="grey20", lwd=.1)+ geom_bo
xplot(outlier.alpha = .5, outlier.shape=16, outlier.size=2, outlier.color="grey20"
, color="grey20", lwd=0.6, fill="grey80", fatten = 1.3)+coord_flip()+
  theme_bw()+
  ylab("Increased\nNode Purity")+
  ggtitle("Uranium")+
  theme(panel.border = element_rect(fill=NA,color="black", size=2,linetype="solid"
),
        axis.ticks = element_line(size=1.1),
        axis.title = element_text(size=30),
        axis.text =  element_text(size=24, colour="black"),
        plot.title = element_text(size=30, face = "bold"),
        plot.margin = ggplot2::margin(10,30,10,10),
        legend.position="none",
        axis.title.y = element_blank(),
        axis.title.x = element_text(margin = ggplot2::margin(t = 10, r = 0, b = 0,
l = 0)),
        panel.grid.major = element_blank(),panel.grid.minor = element_blank())

prop_eux.MSE.plot <- ggplot(Fepy.anox.rf.import, aes(x = Variable, y = IncMSE))+st
at_boxplot(geom ='errorbar', width = 0.6, size=0.6, color="grey20", lwd=.1)+ geom_
boxplot(outlier.alpha = .5, outlier.shape=16, outlier.size=2, outlier.color="grey2
0", color="grey20", lwd=0.6, fill="grey80", fatten = 1.3)+coord_flip()+
  theme_bw()+
  ylab("Increased Mean\nSquared Error (%)")+
  ggtitle("Proportion Euxinic")+
  theme(panel.border = element_rect(fill=NA,color="black", size=2,linetype="solid"
),
        axis.ticks = element_line(size=1.1),
        axis.title = element_text(size=30),
        axis.text =  element_text(size=24, colour="black"),
        plot.title = element_text(size=30, face = "bold"),
        plot.margin = ggplot2::margin(10,30,10,10),
        legend.position="none",
        axis.title.y = element_blank(),
        axis.title.x = element_text(margin = ggplot2::margin(t = 10, r = 0, b = 0,
l = 0)),
        panel.grid.major = element_blank(),panel.grid.minor = element_blank())

prop_eux.Node.plot <- ggplot(Fepy.anox.rf.import, aes(x = Variable, y = IncNodePur
```

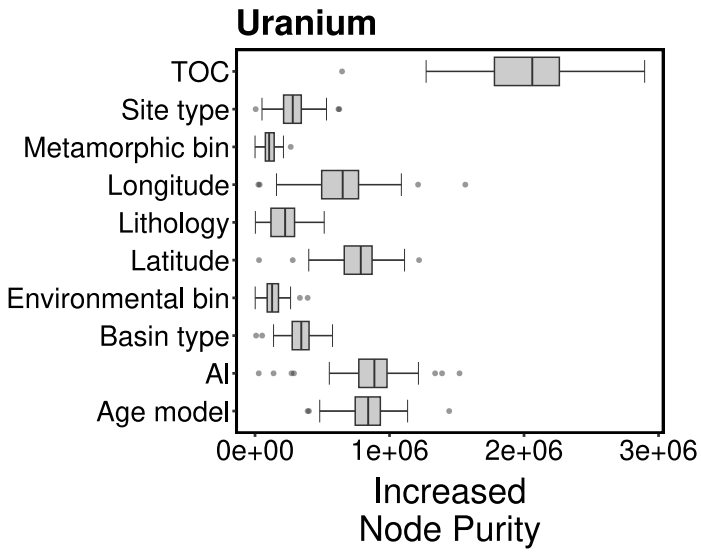
```
ity))+stat_boxplot(geom ='errorbar', width = 0.6, size=0.6, color="grey20", lwd=.1
)+ geom_boxplot(outlier.alpha = .5, outlier.shape=16, outlier.size=2, outlier.colo
r="grey20", color="grey20", lwd=0.6, fill="grey80", fatten = 1.3)+coord_flip()+
  theme_bw()+
  ylab("Increased\nNode Purity")+
  ggtitle("Proportion Euxinic")+
  theme(panel.border = element_rect(fill=NA,color="black", size=2,linetype="solid"
),
        axis.ticks = element_line(size=1.1),
        axis.title = element_text(size=30),
        axis.text =  element_text(size=24, colour="black"),
        plot.title = element_text(size=30, face = "bold"),
        plot.margin = ggplot2::margin(10,30,10,10),
        legend.position="none",
        axis.title.y = element_blank(),
        axis.title.x = element_text(margin = ggplot2::margin(t = 10, r = 0, b = 0,
l = 0)),
        panel.grid.major = element_blank(),panel.grid.minor = element_blank())

TOC.MSE.plot <- ggplot(TOC.all.rf.import, aes(x = Variable, y = IncMSE))+stat_boxp
lot(geom ='errorbar', width = 0.6, size=0.6, color="grey20", lwd=.1)+ geom_boxplot
(outlier.alpha = .5, outlier.shape=16, outlier.size=2, outlier.color="grey20", col
or="grey20", lwd=0.6, fill="grey80", fatten = 1.3)+coord_flip()+
  theme_bw()+
  ylab("Increased Mean\nSquared Error (%)")+
  ggtitle("Total Organic Carbon")+
  theme(panel.border = element_rect(fill=NA,color="black", size=2,linetype="solid"
),
        axis.ticks = element_line(size=1.1),
        axis.title = element_text(size=30),
        axis.text =  element_text(size=24, colour="black"),
        plot.title = element_text(size=30, face = "bold"),
        plot.margin = ggplot2::margin(10,30,10,10),
        legend.position="none",
        axis.title.y = element_blank(),
        axis.title.x = element_text(margin = ggplot2::margin(t = 10, r = 0, b = 0,
l = 0)),
        panel.grid.major = element_blank(),panel.grid.minor = element_blank())

TOC.Node.plot <- ggplot(TOC.all.rf.import, aes(x = Variable, y = IncNodePurity))+s
tat_boxplot(geom ='errorbar', width = 0.6, size=0.6, color="grey20", lwd=.1)+ geom
_boxplot(outlier.alpha = .5, outlier.shape=16, outlier.size=2, outlier.color="grey
20", color="grey20", lwd=0.6, fill="grey80", fatten = 1.3)+coord_flip()+
  theme_bw()+
  ylab("Increased\nNode Purity")+
  ggtitle("Total Organic Carbon")+
  theme(panel.border = element_rect(fill=NA,color="black", size=2,linetype="solid"
),
        axis.ticks = element_line(size=1.1),
        axis.title = element_text(size=30),
        axis.text =  element_text(size=24, colour="black"),
        plot.title = element_text(size=30, face = "bold"),
        plot.margin = ggplot2::margin(10,30,10,10),
        legend.position="none",
```

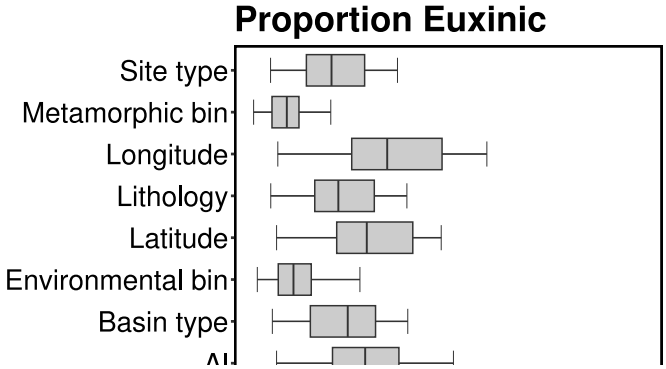
```
        axis.title.y = element_blank(),
        axis.title.x = element_text(margin = ggplot2::margin(t = 10, r = 0, b = 0,
l = 0)),
        panel.grid.major = element_blank(),panel.grid.minor = element_blank())

Imp.sum <- ggarrange2(Mo.MSE.plot,
           Mo.Node.plot,
           U.MSE.plot,
           U.Node.plot,
           prop_eux.MSE.plot,
           prop_eux.Node.plot,
           TOC.MSE.plot,
           TOC.Node.plot,
           ncol=2)
```

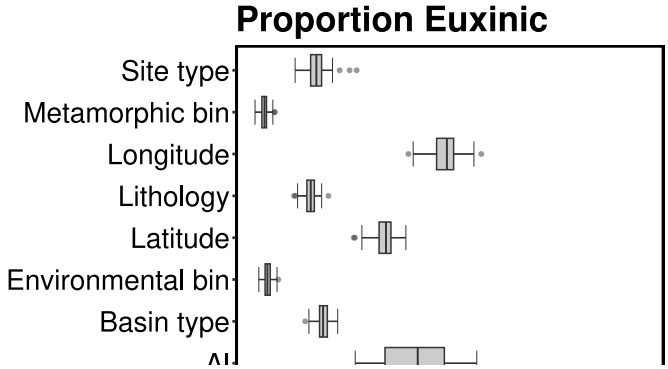


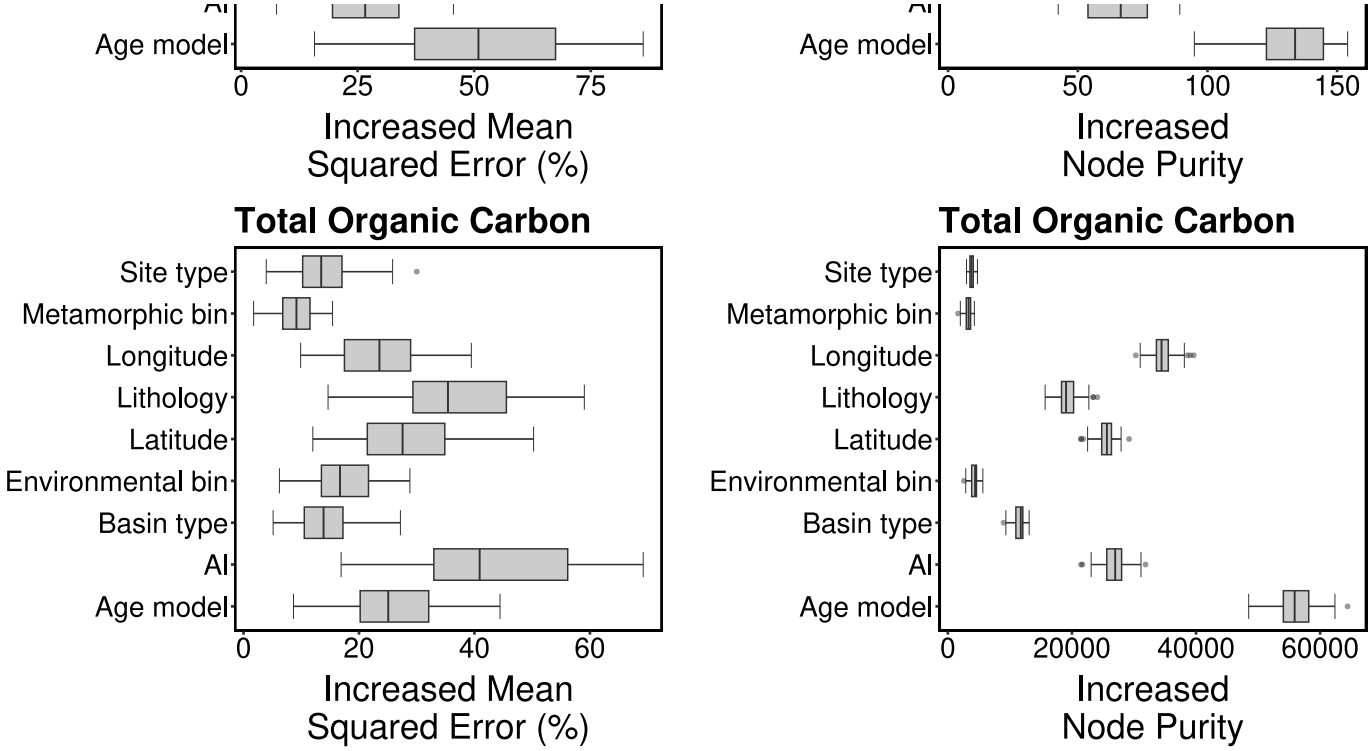













```
ggsave("Figure Sx Variable importance plots for Monte Carlo random forest 20240207
100 iterations.pdf", Imp.sum, height=27, width=17)
```

10. Model fit plots

We will summarize both model fit and the cross validated random forest variables used in our primary Monte Carlo random forest analyses, using box and whisker plots to illustrate the distributions generated by our Monte Carlo approach.

First, make grouped dataframe for all model fit.

```
Mo.anox.py.rf.model$proxy <- "Mo"
U.anox.rf.model$proxy <- "U"
Fepy.anox.rf.model$proxy <- "Fepy"
TOC.all.rf.model$proxy <- "TOC"


model.sum <- rbind(Mo.anox.py.rf.model,
                   U.anox.rf.model,
                   Fepy.anox.rf.model,
                   TOC.all.rf.model
                   )

# for just the MSE plots, separate ppm and wt (%) proxies
ppm.model.sum <- rbind(Mo.anox.py.rf.model,
                   U.anox.rf.model
                   )


perc.model.sum <- rbind(Fepy.anox.rf.model,
                   TOC.all.rf.model
                   )
```

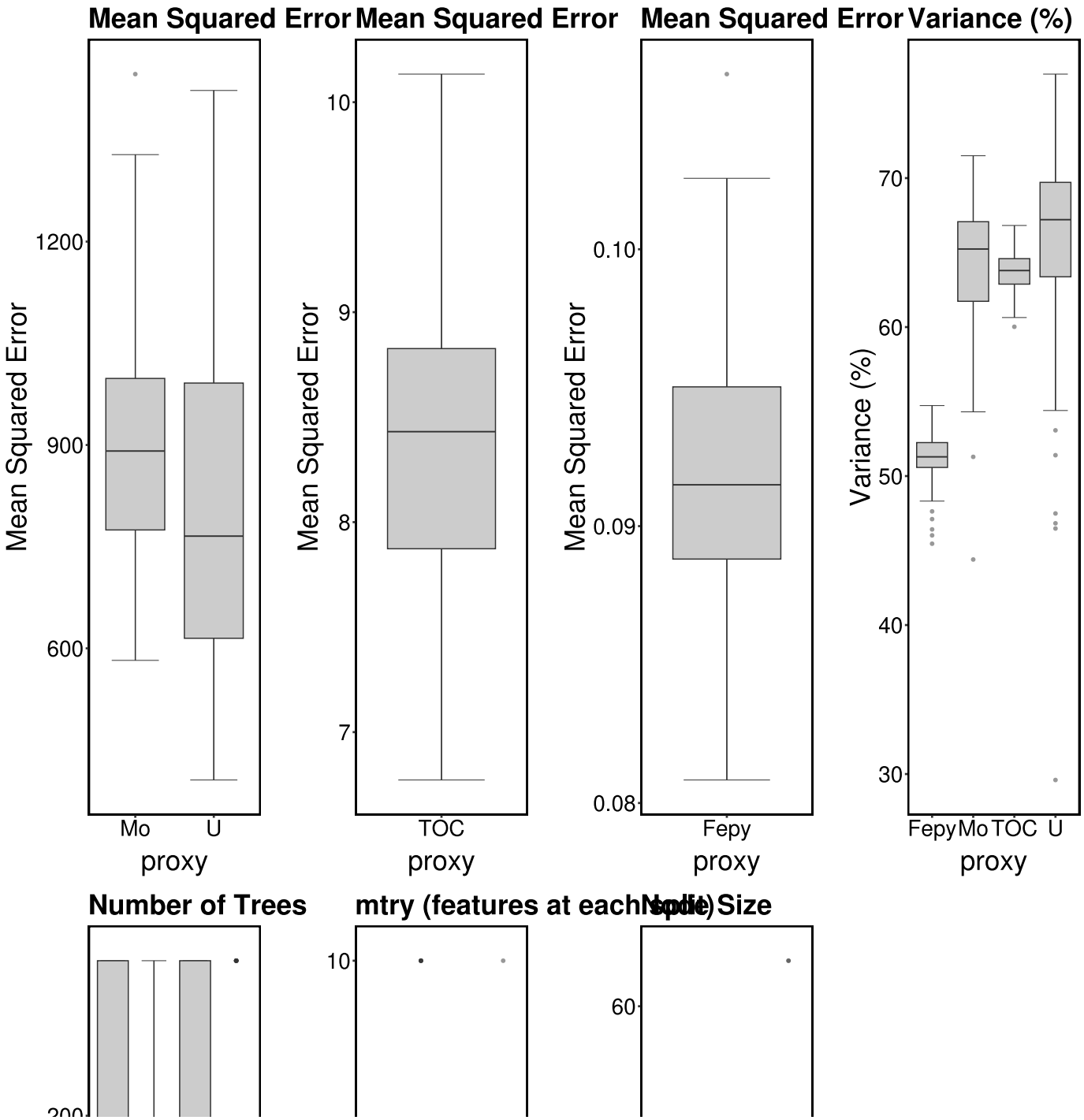
Then, we make a summary figure…

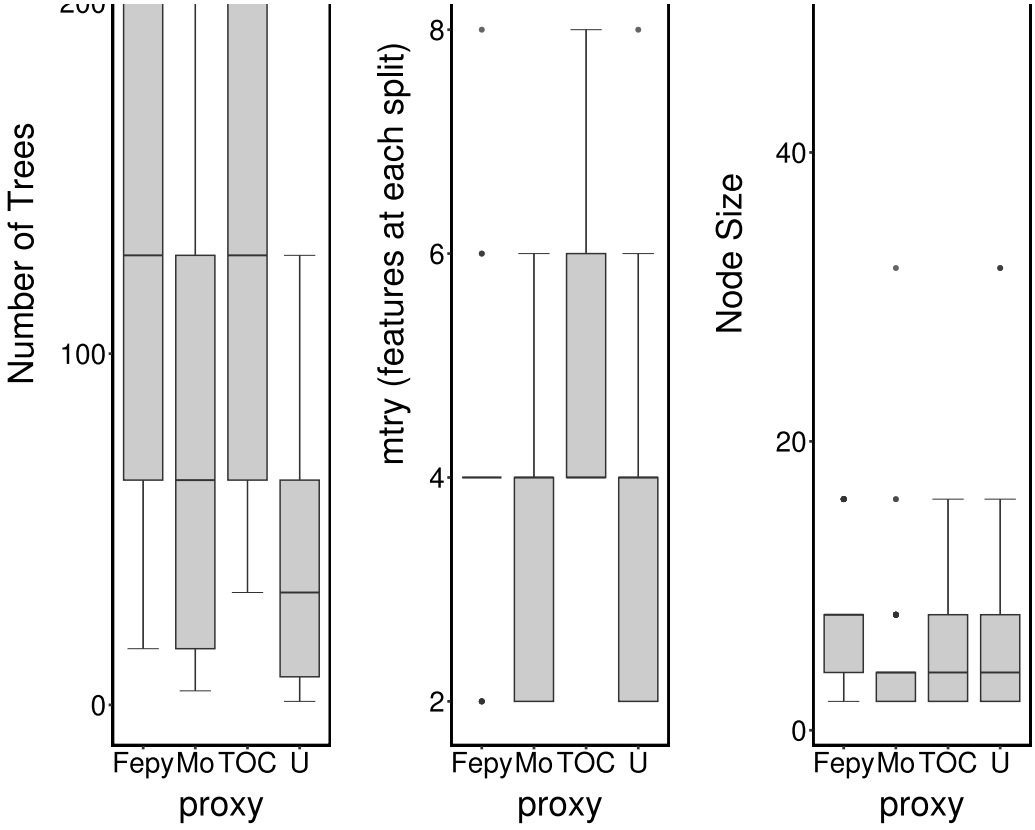
```
#names(model.fit.data) <- c("best.nodesize", "best.mtry", "best.ntree", "best.MSE"
, "best.var")


MSE.plot.ppm <- ggplot(ppm.model.sum, aes(x = proxy, y = best.MSE))+stat_boxplot(g
eom ='errorbar', width = 0.6, size=0.6, color="grey20", lwd=.1)+ geom_boxplot(outl
ier.alpha = .5, outlier.shape=16, outlier.size=2, outlier.color="grey20", color="g
rey20", lwd=0.6, fill="grey80", fatten = 1.3)+
  theme_bw()+
  ylab("Mean Squared Error")+
  ggtitle("Mean Squared Error")+
  theme(panel.border = element_rect(fill=NA,color="black", size=2,linetype="solid"
),
        axis.ticks = element_line(size=1.1),
        axis.title = element_text(size=30),
        axis.text =  element_text(size=24, colour="black"),
        plot.title = element_text(size=30, face = "bold"),
        plot.margin = ggplot2::margin(10,30,10,10),
        legend.position="none",
        #axis.title.y = element_blank(),
        axis.title.x = element_text(margin = ggplot2::margin(t = 10, r = 0, b = 0,
l = 0)),
        panel.grid.major = element_blank(),panel.grid.minor = element_blank())


MSE.plot.TOC <- ggplot(TOC.all.rf.model, aes(x = proxy, y = best.MSE))+stat_boxplo
t(geom ='errorbar', width = 0.6, size=0.6, color="grey20", lwd=.1)+ geom_boxplot(o
utlier.alpha = .5, outlier.shape=16, outlier.size=2, outlier.color="grey20", color
="grey20", lwd=0.6, fill="grey80", fatten = 1.3)+
```

```
  theme_bw()+
  ylab("Mean Squared Error")+
  ggtitle("Mean Squared Error")+
  theme(panel.border = element_rect(fill=NA,color="black", size=2,linetype="solid"
),
        axis.ticks = element_line(size=1.1),
        axis.title = element_text(size=30),
        axis.text =  element_text(size=24, colour="black"),
        plot.title = element_text(size=30, face = "bold"),
        plot.margin = ggplot2::margin(10,30,10,10),
        legend.position="none",
        #axis.title.y = element_blank(),
        axis.title.x = element_text(margin = ggplot2::margin(t = 10, r = 0, b = 0,
l = 0)),
        panel.grid.major = element_blank(),panel.grid.minor = element_blank())

MSE.plot.Fepy <- ggplot(Fepy.anox.rf.model, aes(x = proxy, y = best.MSE))+stat_box
plot(geom ='errorbar', width = 0.6, size=0.6, color="grey20", lwd=.1)+ geom_boxplo
t(outlier.alpha = .5, outlier.shape=16, outlier.size=2, outlier.color="grey20", co
lor="grey20", lwd=0.6, fill="grey80", fatten = 1.3)+
  theme_bw()+
  ylab("Mean Squared Error")+
  ggtitle("Mean Squared Error")+
  theme(panel.border = element_rect(fill=NA,color="black", size=2,linetype="solid"
),
        axis.ticks = element_line(size=1.1),
        axis.title = element_text(size=30),
        axis.text =  element_text(size=24, colour="black"),
        plot.title = element_text(size=30, face = "bold"),
        plot.margin = ggplot2::margin(10,30,10,10),
        legend.position="none",
        #axis.title.y = element_blank(),
        axis.title.x = element_text(margin = ggplot2::margin(t = 10, r = 0, b = 0,
l = 0)),
        panel.grid.major = element_blank(),panel.grid.minor = element_blank())

var.plot <- ggplot(model.sum, aes(x = proxy, y = best.var))+stat_boxplot(geom ='er
rorbar', width = 0.6, size=0.6, color="grey20", lwd=.1)+ geom_boxplot(outlier.alph
a = .5, outlier.shape=16, outlier.size=2, outlier.color="grey20", color="grey20",
lwd=0.6, fill="grey80", fatten = 1.3)+
  theme_bw()+
  ylab("Variance (%)")+
  ggtitle("Variance (%)")+
  theme(panel.border = element_rect(fill=NA,color="black", size=2,linetype="solid"
),
        axis.ticks = element_line(size=1.1),
        axis.title = element_text(size=30),
        axis.text =  element_text(size=24, colour="black"),
        plot.title = element_text(size=30, face = "bold"),
        plot.margin = ggplot2::margin(10,30,10,10),
        legend.position="none",
        #axis.title.y = element_blank(),
        axis.title.x = element_text(margin = ggplot2::margin(t = 10, r = 0, b = 0,
l = 0)),
```

```
        panel.grid.major = element_blank(),panel.grid.minor = element_blank())

ntree.plot <- ggplot(model.sum, aes(x = proxy, y = best.ntree))+stat_boxplot(geom
='errorbar', width = 0.6, size=0.6, color="grey20", lwd=.1)+ geom_boxplot(outlier.
alpha = .5, outlier.shape=16, outlier.size=2, outlier.color="grey20", color="grey2
0", lwd=0.6, fill="grey80", fatten = 1.3)+
  theme_bw()+
  ylab("Number of Trees")+
  ggtitle("Number of Trees")+
  theme(panel.border = element_rect(fill=NA,color="black", size=2,linetype="solid"
),
        axis.ticks = element_line(size=1.1),
        axis.title = element_text(size=30),
        axis.text =  element_text(size=24, colour="black"),
        plot.title = element_text(size=30, face = "bold"),
        plot.margin = ggplot2::margin(10,30,10,10),
        legend.position="none",
        #axis.title.y = element_blank(),
        axis.title.x = element_text(margin = ggplot2::margin(t = 10, r = 0, b = 0,
l = 0)),
        panel.grid.major = element_blank(),panel.grid.minor = element_blank())

mtry.plot <- ggplot(model.sum, aes(x = proxy, y = best.mtry))+stat_boxplot(geom ='
errorbar', width = 0.6, size=0.6, color="grey20", lwd=.1)+ geom_boxplot(outlier.al
pha = .5, outlier.shape=16, outlier.size=2, outlier.color="grey20", color="grey20"
, lwd=0.6, fill="grey80", fatten = 1.3)+
  theme_bw()+
  ylab("mtry (features at each split)")+
  ggtitle("mtry (features at each split)")+
  theme(panel.border = element_rect(fill=NA,color="black", size=2,linetype="solid"
),
        axis.ticks = element_line(size=1.1),
        axis.title = element_text(size=30),
        axis.text =  element_text(size=24, colour="black"),
        plot.title = element_text(size=30, face = "bold"),
        plot.margin = ggplot2::margin(10,30,10,10),
        legend.position="none",
        #axis.title.y = element_blank(),
        axis.title.x = element_text(margin = ggplot2::margin(t = 10, r = 0, b = 0,
l = 0)),
        panel.grid.major = element_blank(),panel.grid.minor = element_blank())

nodesize.plot <- ggplot(model.sum, aes(x = proxy, y = best.nodesize))+stat_boxplot
(geom ='errorbar', width = 0.6, size=0.6, color="grey20", lwd=.1)+ geom_boxplot(ou
tlier.alpha = .5, outlier.shape=16, outlier.size=2, outlier.color="grey20", color=
"grey20", lwd=0.6, fill="grey80", fatten = 1.3)+
  theme_bw()+
  ylab("Node Size")+
  ggtitle("Node Size")+
  theme(panel.border = element_rect(fill=NA,color="black", size=2,linetype="solid"
),
        axis.ticks = element_line(size=1.1),
        axis.title = element_text(size=30),
        axis.text =  element_text(size=24, colour="black"),
```

```
        plot.title = element_text(size=30, face = "bold"),
        plot.margin = ggplot2::margin(10,30,10,10),
        legend.position="none",
        #axis.title.y = element_blank(),
        axis.title.x = element_text(margin = ggplot2::margin(t = 10, r = 0, b = 0,
l = 0)),
        panel.grid.major = element_blank(),panel.grid.minor = element_blank())

model.sum <- ggarrange2(MSE.plot.ppm,
          MSE.plot.TOC,
          MSE.plot.Fepy,
           var.plot,
           ntree.plot,
           mtry.plot,
           nodesize.plot,
           ncol=4)
```






```
ggsave("Figure Sx Model fit and parameter plots for Monte Carlo random forest 2024
0207 100 iterations.pdf", model.sum, height=15, width=32)
```

11. Appendix - Three phase plot evolution for talks

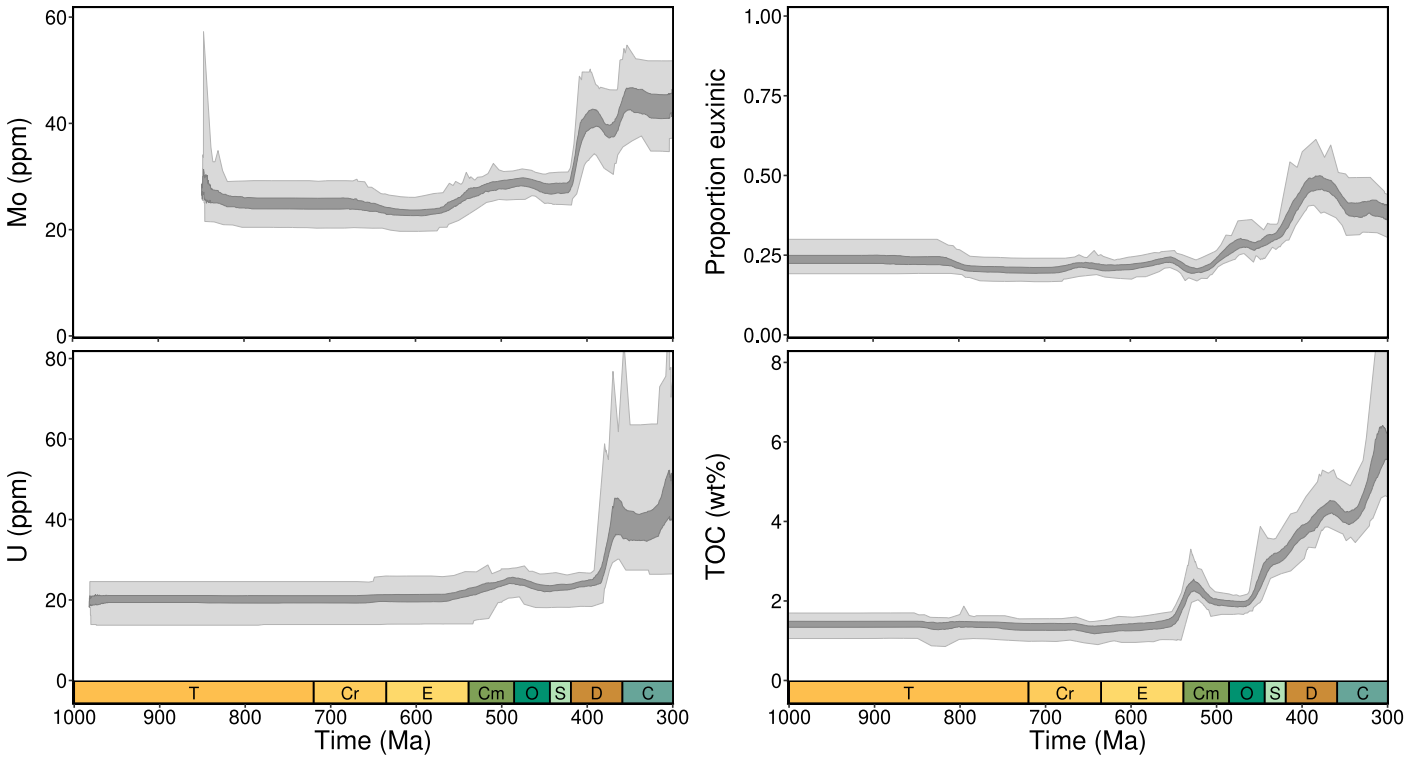
No shading.

```
Mo.plot.for.sum.no.shade <- ggplot(mo.pdp, aes(x=Age))+
  #annotate(geom="rect", xmax=Inf, xmin=541, ymin=-Inf, ymax=Inf, fill="#C9DEE8",
alpha=0.2)+
  #annotate(geom="rect", xmax=541, xmin=418, ymin=-Inf, ymax=Inf, fill="#9DC2D6",
alpha=0.4)+
  #annotate(geom="rect", xmax=418, xmin=-Inf, ymin=-Inf, ymax=Inf, fill="#619FBB",
alpha=0.4)+
  geom_ribbon(aes(ymin = min, ymax = max), alpha=1, fill="grey85", color="grey70",
size=.3)+
  geom_ribbon(aes(ymin = perc.25, ymax = perc.75), alpha=1, fill="grey60", color="
grey50", size=.5)+
  theme_bw()+
  coord_cartesian(xlim=rev(c(298.9,1000)), ylim=c(-.4,62),expand=FALSE)+
  scale_x_reverse(breaks=c(300,400,500,600,700,800,900,1000))+
  ylab("Mo (ppm)")+xlab("Time (Ma)")+
  theme(plot.margin = ggplot2::margin(.1,1,.3,1,"cm"),panel.border = element_rect(
fill=NA,color="black", size=2,linetype="solid"),
        axis.ticks = element_line(size=1),
        axis.line = element_line(lineend = 'square'),
        axis.title = element_text(size=34),
```

```
        axis.text = element_text( size=26, color="black"),
        legend.title = element_text(size=20),
        legend.text = element_text( size=16),
        axis.ticks.length = unit(5, "points"),
        legend.position="top",         legend.justification = c(0, 0),
        axis.title.x = element_blank(),
        axis.text.x = element_blank(),
        axis.title.y = element_text(vjust=3),
        panel.grid.major = element_blank(),panel.grid.minor = element_blank())

U.plot.for.sum.no.shade <- ggplot(u.pdp, aes(x=Age))+
  #annotate(geom="rect", xmax=Inf, xmin=541, ymin=-Inf, ymax=Inf, fill="#C9DEE8",
alpha=0.2)+
  #annotate(geom="rect", xmax=541, xmin=418, ymin=-Inf, ymax=Inf, fill="#9DC2D6",
alpha=0.4)+
  #annotate(geom="rect", xmax=418, xmin=-Inf, ymin=-Inf, ymax=Inf, fill="#619FBB",
alpha=0.4)+
  geom_ribbon(aes(ymin = min, ymax = max), alpha=1, fill="grey85", color="grey70",
size=.3)+
  geom_ribbon(aes(ymin = perc.25, ymax = perc.75), alpha=1, fill="grey60", color="
grey50", size=.5)+
  theme_bw()+
  coord_geo(xlim=rev(c(298.9,1002)), expand=FALSE, ylim=c(-.4,82),
            pos = as.list(rep("bottom", 1)),
            abbrv=list(TRUE),
            dat = list(periods.edit),
            height = list(unit(2, "lines")),
            bord=list(c("left", "bottom", "right")), lwd=as.list(c(1)), size=8)+
  scale_x_reverse(breaks=c(300,400,500,600,700,800,900,1000))+
  ylab("U (ppm)")+xlab("Time (Ma)")+
  theme(plot.margin = ggplot2::margin(.1,1,.3,1,"cm"),panel.border = element_rect(
fill=NA,color="black", size=2,linetype="solid"),
        axis.ticks = element_line(size=1),
        axis.line = element_line(lineend = 'square'),
        axis.title = element_text(size=34),
        axis.text = element_text( size=26, color="black"),
        legend.title = element_text(size=20),
        legend.text = element_text( size=16),
        axis.ticks.length = unit(5, "points"),
        legend.position="top",
        legend.justification = c(0, 0),
        axis.title.y = element_text(vjust=3),
        panel.grid.major = element_blank(),panel.grid.minor = element_blank())

prop_eux.plot.for.sum.no.shade <- ggplot(prop_eux.pdp, aes(x=Age))+
  #annotate(geom="rect", xmax=Inf, xmin=541, ymin=-Inf, ymax=Inf, fill="#C9DEE8",
alpha=0.2)+
  #annotate(geom="rect", xmax=541, xmin=418, ymin=-Inf, ymax=Inf, fill="#9DC2D6",
alpha=0.4)+
  #annotate(geom="rect", xmax=418, xmin=-Inf, ymin=-Inf, ymax=Inf, fill="#619FBB",
alpha=0.4)+
  geom_ribbon(aes(ymin = min, ymax = max), alpha=1, fill="grey85", color="grey70",
size=.3)+
  geom_ribbon(aes(ymin = perc.25, ymax = perc.75), alpha=1, fill="grey60", color="
```

```
grey50", size=.5)+
  theme_bw()+
  coord_cartesian(xlim=rev(c(298.9,1000)), ylim=c(-.01,1.03),expand=FALSE)+
  scale_x_reverse(breaks=c(300,400,500,600,700,800,900,1000))+
  ylab("Proportion euxinic")+xlab("Time (Ma)")+
  theme(plot.margin = ggplot2::margin(.1,1,.3,1,"cm"),panel.border = element_rect(
fill=NA,color="black", size=2,linetype="solid"),
        axis.ticks = element_line(size=1),
        axis.line = element_line(lineend = 'square'),
        axis.title = element_text(size=34),
        axis.text = element_text( size=26, color="black"),
        legend.title = element_text(size=20),
        legend.text = element_text( size=16),
        axis.ticks.length = unit(5, "points"),
        legend.position="top",         legend.justification = c(0, 0),
        axis.title.x = element_blank(),
        axis.text.x = element_blank(),
        axis.title.y = element_text(vjust=3),
        panel.grid.major = element_blank(),panel.grid.minor = element_blank())

TOC.plot.for.sum.no.shade <- ggplot(TOC.pdp, aes(x=Age))+
  #annotate(geom="rect", xmax=Inf, xmin=541, ymin=-Inf, ymax=Inf, fill="#C9DEE8",
alpha=0.2)+
  #annotate(geom="rect", xmax=541, xmin=418, ymin=-Inf, ymax=Inf, fill="#9DC2D6",
alpha=0.4)+
  #annotate(geom="rect", xmax=418, xmin=-Inf, ymin=-Inf, ymax=Inf, fill="#619FBB",
alpha=0.4)+
  geom_ribbon(aes(ymin = min, ymax = max), alpha=1, fill="grey85", color="grey70",
size=.3)+
  geom_ribbon(aes(ymin = perc.25, ymax = perc.75), alpha=1, fill="grey60", color="
grey50", size=.5)+
  theme_bw()+
  coord_geo(xlim=rev(c(298.9,1002)), expand=FALSE, ylim=c(-0.04,8.3),
            pos = as.list(rep("bottom", 1)),
            abbrv=list(TRUE),
            dat = list(periods.edit),
            height = list(unit(2, "lines")),
            bord=list(c("left", "bottom", "right")), lwd=as.list(c(1)), size=8)+
  scale_x_reverse(breaks=c(300,400,500,600,700,800,900,1000))+
  ylab("TOC (wt%)")+xlab("Time (Ma)")+
  theme(plot.margin = ggplot2::margin(.1,1,.3,1,"cm"),panel.border = element_rect(
fill=NA,color="black", size=2,linetype="solid"),
        axis.ticks = element_line(size=1),
        axis.line = element_line(lineend = 'square'),
        axis.title = element_text(size=34),
        axis.text = element_text( size=26, color="black"),
        legend.title = element_text(size=20),
        legend.text = element_text( size=16),
        axis.ticks.length = unit(5, "points"),
        legend.position="top",
        legend.justification = c(0, 0),
        axis.title.y = element_text(vjust=3),
        panel.grid.major = element_blank(),panel.grid.minor = element_blank())
```

```
summary.plot.no.shade <- ggarrange2(Mo.plot.for.sum.no.shade, prop_eux.plot.for.su
m.no.shade, U.plot.for.sum.no.shade, TOC.plot.for.sum.no.shade, ncol=2, heights=c(
1,1))
```



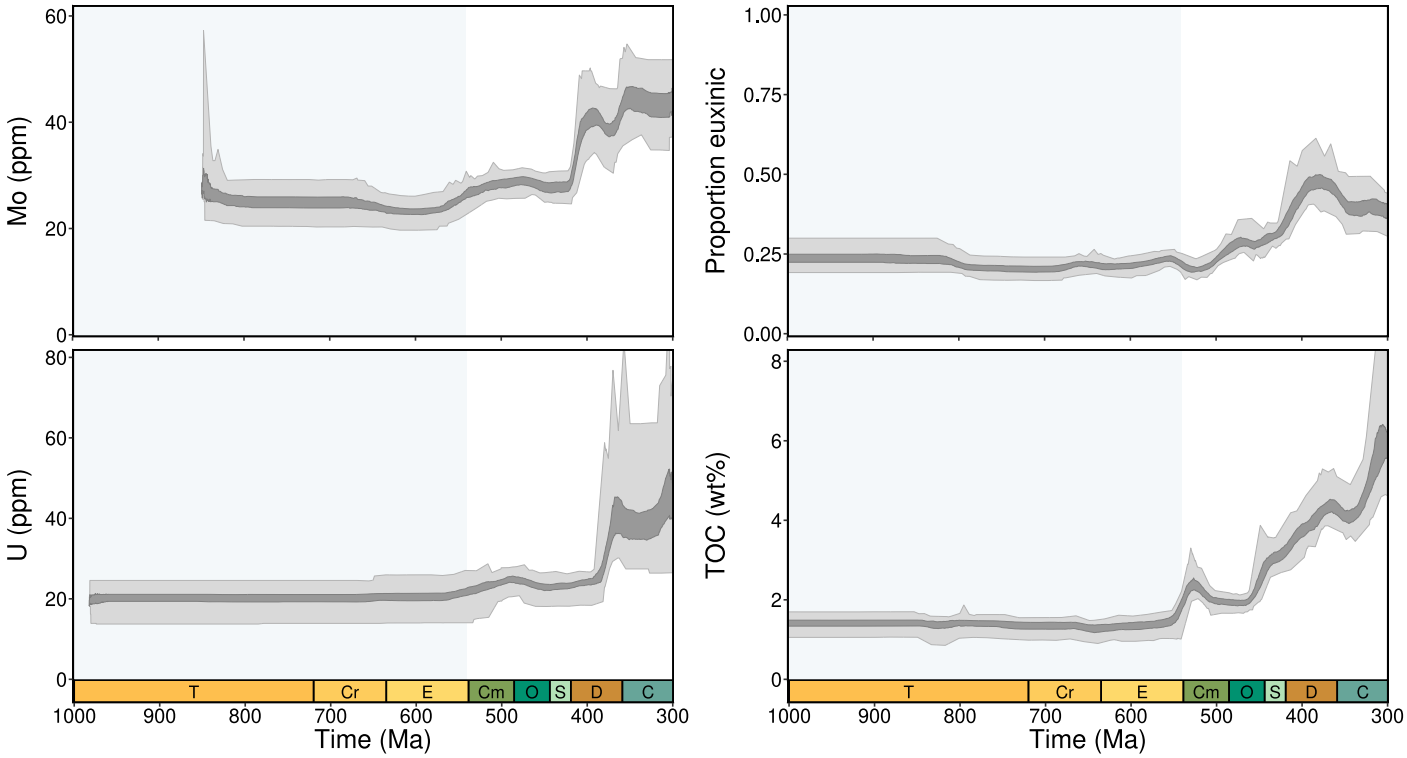

```
ggsave("Figure 2 Partial Dependence Plot no shading 20240207 100 iterations.pdf",
summary.plot.no.shade, height=14, width=26)
```

Neoproterozoic shading.

```
Mo.plot.for.sum.shade.1 <- ggplot(mo.pdp, aes(x=Age))+
  annotate(geom="rect", xmax=Inf, xmin=541, ymin=-Inf, ymax=Inf, fill="#C9DEE8", a
lpha=0.2)+
  #annotate(geom="rect", xmax=541, xmin=418, ymin=-Inf, ymax=Inf, fill="#9DC2D6",
alpha=0.4)+
  #annotate(geom="rect", xmax=418, xmin=-Inf, ymin=-Inf, ymax=Inf, fill="#619FBB",
alpha=0.4)+
  geom_ribbon(aes(ymin = min, ymax = max), alpha=1, fill="grey85", color="grey70",
size=.3)+
  geom_ribbon(aes(ymin = perc.25, ymax = perc.75), alpha=1, fill="grey60", color="
grey50", size=.5)+
  theme_bw()+
  coord_cartesian(xlim=rev(c(298.9,1000)), ylim=c(-.4,62),expand=FALSE)+
  scale_x_reverse(breaks=c(300,400,500,600,700,800,900,1000))+
  ylab("Mo (ppm)")+xlab("Time (Ma)")+
  theme(plot.margin = ggplot2::margin(.1,1,.3,1,"cm"),panel.border = element_rect(
fill=NA,color="black", size=2,linetype="solid"),
        axis.ticks = element_line(size=1),
        axis.line = element_line(lineend = 'square'),
        axis.title = element_text(size=34),
        axis.text = element_text( size=26, color="black"),
        legend.title = element_text(size=20),
        legend.text = element_text( size=16),
```

```
        axis.ticks.length = unit(5, "points"),
        legend.position="top",         legend.justification = c(0, 0),
        axis.title.x = element_blank(),
        axis.text.x = element_blank(),
        axis.title.y = element_text(vjust=3),
        panel.grid.major = element_blank(),panel.grid.minor = element_blank())

U.plot.for.sum.shade.1 <- ggplot(u.pdp, aes(x=Age))+
  annotate(geom="rect", xmax=Inf, xmin=541, ymin=-Inf, ymax=Inf, fill="#C9DEE8", a
lpha=0.2)+
  #annotate(geom="rect", xmax=541, xmin=418, ymin=-Inf, ymax=Inf, fill="#9DC2D6",
alpha=0.4)+
  #annotate(geom="rect", xmax=418, xmin=-Inf, ymin=-Inf, ymax=Inf, fill="#619FBB",
alpha=0.4)+
  geom_ribbon(aes(ymin = min, ymax = max), alpha=1, fill="grey85", color="grey70",
size=.3)+
  geom_ribbon(aes(ymin = perc.25, ymax = perc.75), alpha=1, fill="grey60", color="
grey50", size=.5)+
  theme_bw()+
  coord_geo(xlim=rev(c(298.9,1002)), expand=FALSE, ylim=c(-.4,82),
            pos = as.list(rep("bottom", 1)),
            abbrv=list(TRUE),
            dat = list(periods.edit),
            height = list(unit(2, "lines")),
            bord=list(c("left", "bottom", "right")), lwd=as.list(c(1)), size=8)+
  scale_x_reverse(breaks=c(300,400,500,600,700,800,900,1000))+
  ylab("U (ppm)")+xlab("Time (Ma)")+
  theme(plot.margin = ggplot2::margin(.1,1,.3,1,"cm"),panel.border = element_rect(
fill=NA,color="black", size=2,linetype="solid"),
        axis.ticks = element_line(size=1),
        axis.line = element_line(lineend = 'square'),
        axis.title = element_text(size=34),
        axis.text = element_text( size=26, color="black"),
        legend.title = element_text(size=20),
        legend.text = element_text( size=16),
        axis.ticks.length = unit(5, "points"),
        legend.position="top",
        legend.justification = c(0, 0),
        axis.title.y = element_text(vjust=3),
        panel.grid.major = element_blank(),panel.grid.minor = element_blank())

prop_eux.plot.for.sum.shade.1 <- ggplot(prop_eux.pdp, aes(x=Age))+
  annotate(geom="rect", xmax=Inf, xmin=541, ymin=-Inf, ymax=Inf, fill="#C9DEE8", a
lpha=0.2)+
  #annotate(geom="rect", xmax=541, xmin=418, ymin=-Inf, ymax=Inf, fill="#9DC2D6",
alpha=0.4)+
  #annotate(geom="rect", xmax=418, xmin=-Inf, ymin=-Inf, ymax=Inf, fill="#619FBB",
alpha=0.4)+
  geom_ribbon(aes(ymin = min, ymax = max), alpha=1, fill="grey85", color="grey70",
size=.3)+
  geom_ribbon(aes(ymin = perc.25, ymax = perc.75), alpha=1, fill="grey60", color="
grey50", size=.5)+
  theme_bw()+
  coord_cartesian(xlim=rev(c(298.9,1000)), ylim=c(-.01,1.03),expand=FALSE)+
```

```
  scale_x_reverse(breaks=c(300,400,500,600,700,800,900,1000))+
  ylab("Proportion euxinic")+xlab("Time (Ma)")+
  theme(plot.margin = ggplot2::margin(.1,1,.3,1,"cm"),panel.border = element_rect(
fill=NA,color="black", size=2,linetype="solid"),
        axis.ticks = element_line(size=1),
        axis.line = element_line(lineend = 'square'),
        axis.title = element_text(size=34),
        axis.text = element_text( size=26, color="black"),
        legend.title = element_text(size=20),
        legend.text = element_text( size=16),
        axis.ticks.length = unit(5, "points"),
        legend.position="top",         legend.justification = c(0, 0),
        axis.title.x = element_blank(),
        axis.text.x = element_blank(),
        axis.title.y = element_text(vjust=3),
        panel.grid.major = element_blank(),panel.grid.minor = element_blank())

TOC.plot.for.sum.shade.1 <- ggplot(TOC.pdp, aes(x=Age))+
  annotate(geom="rect", xmax=Inf, xmin=541, ymin=-Inf, ymax=Inf, fill="#C9DEE8", a
lpha=0.2)+
  #annotate(geom="rect", xmax=541, xmin=418, ymin=-Inf, ymax=Inf, fill="#9DC2D6",
alpha=0.4)+
  #annotate(geom="rect", xmax=418, xmin=-Inf, ymin=-Inf, ymax=Inf, fill="#619FBB",
alpha=0.4)+
  geom_ribbon(aes(ymin = min, ymax = max), alpha=1, fill="grey85", color="grey70",
size=.3)+
  geom_ribbon(aes(ymin = perc.25, ymax = perc.75), alpha=1, fill="grey60", color="
grey50", size=.5)+
  theme_bw()+
  coord_geo(xlim=rev(c(298.9,1002)), expand=FALSE, ylim=c(-0.04,8.3),
            pos = as.list(rep("bottom", 1)),
            abbrv=list(TRUE),
            dat = list(periods.edit),
            height = list(unit(2, "lines")),
            bord=list(c("left", "bottom", "right")), lwd=as.list(c(1)), size=8)+
  scale_x_reverse(breaks=c(300,400,500,600,700,800,900,1000))+
  ylab("TOC (wt%)")+xlab("Time (Ma)")+
  theme(plot.margin = ggplot2::margin(.1,1,.3,1,"cm"),panel.border = element_rect(
fill=NA,color="black", size=2,linetype="solid"),
        axis.ticks = element_line(size=1),
        axis.line = element_line(lineend = 'square'),
        axis.title = element_text(size=34),
        axis.text = element_text( size=26, color="black"),
        legend.title = element_text(size=20),
        legend.text = element_text( size=16),
        axis.ticks.length = unit(5, "points"),
        legend.position="top",
        legend.justification = c(0, 0),
        axis.title.y = element_text(vjust=3),
        panel.grid.major = element_blank(),panel.grid.minor = element_blank())

summary.plot.shade.1<- ggarrange2(Mo.plot.for.sum.shade.1, prop_eux.plot.for.sum.s
hade.1, U.plot.for.sum.shade.1, TOC.plot.for.sum.shade.1, ncol=2, heights=c(1,1))
```



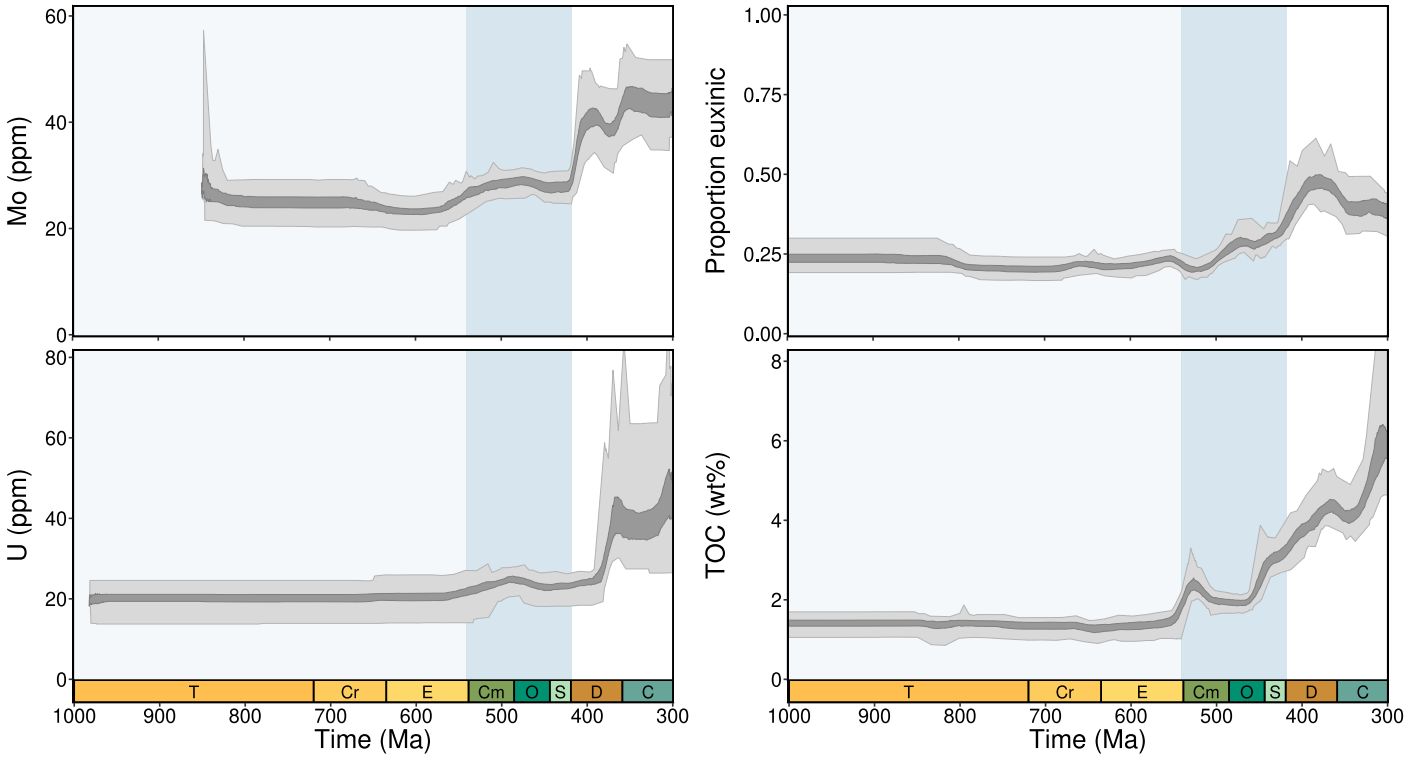

```
ggsave("Figure 2 Partial Dependence Plot with shading part 1 20240207 100 iteratio
ns.pdf", summary.plot.shade.1, height=14, width=26)
```

Neoproterozoic and early Paleozoic shading.

```
Mo.plot.for.sum.shade.2 <- ggplot(mo.pdp, aes(x=Age))+
  annotate(geom="rect", xmax=Inf, xmin=541, ymin=-Inf, ymax=Inf, fill="#C9DEE8", a
lpha=0.2)+
  annotate(geom="rect", xmax=541, xmin=418, ymin=-Inf, ymax=Inf, fill="#9DC2D6", a
lpha=0.4)+
  #annotate(geom="rect", xmax=418, xmin=-Inf, ymin=-Inf, ymax=Inf, fill="#619FBB",
alpha=0.4)+
  geom_ribbon(aes(ymin = min, ymax = max), alpha=1, fill="grey85", color="grey70",
size=.3)+
  geom_ribbon(aes(ymin = perc.25, ymax = perc.75), alpha=1, fill="grey60", color="
grey50", size=.5)+
  theme_bw()+
  coord_cartesian(xlim=rev(c(298.9,1000)), ylim=c(-.4,62),expand=FALSE)+
  scale_x_reverse(breaks=c(300,400,500,600,700,800,900,1000))+
  ylab("Mo (ppm)")+xlab("Time (Ma)")+
  theme(plot.margin = ggplot2::margin(.1,1,.3,1,"cm"),panel.border = element_rect(
fill=NA,color="black", size=2,linetype="solid"),
        axis.ticks = element_line(size=1),
        axis.line = element_line(lineend = 'square'),
        axis.title = element_text(size=34),
        axis.text = element_text( size=26, color="black"),
        legend.title = element_text(size=20),
        legend.text = element_text( size=16),
        axis.ticks.length = unit(5, "points"),
        legend.position="top",         legend.justification = c(0, 0),
        axis.title.x = element_blank(),
        axis.text.x = element_blank(),
```

```
        axis.title.y = element_text(vjust=3),
        panel.grid.major = element_blank(),panel.grid.minor = element_blank())

U.plot.for.sum.shade.2 <- ggplot(u.pdp, aes(x=Age))+
  annotate(geom="rect", xmax=Inf, xmin=541, ymin=-Inf, ymax=Inf, fill="#C9DEE8", a
lpha=0.2)+
  annotate(geom="rect", xmax=541, xmin=418, ymin=-Inf, ymax=Inf, fill="#9DC2D6", a
lpha=0.4)+
  #annotate(geom="rect", xmax=418, xmin=-Inf, ymin=-Inf, ymax=Inf, fill="#619FBB",
alpha=0.4)+
  geom_ribbon(aes(ymin = min, ymax = max), alpha=1, fill="grey85", color="grey70",
size=.3)+
  geom_ribbon(aes(ymin = perc.25, ymax = perc.75), alpha=1, fill="grey60", color="
grey50", size=.5)+
  theme_bw()+
  coord_geo(xlim=rev(c(298.9,1002)), expand=FALSE, ylim=c(-.4,82),
            pos = as.list(rep("bottom", 1)),
            abbrv=list(TRUE),
            dat = list(periods.edit),
            height = list(unit(2, "lines")),
            bord=list(c("left", "bottom", "right")), lwd=as.list(c(1)), size=8)+
  scale_x_reverse(breaks=c(300,400,500,600,700,800,900,1000))+
  ylab("U (ppm)")+xlab("Time (Ma)")+
  theme(plot.margin = ggplot2::margin(.1,1,.3,1,"cm"),panel.border = element_rect(
fill=NA,color="black", size=2,linetype="solid"),
        axis.ticks = element_line(size=1),
        axis.line = element_line(lineend = 'square'),
        axis.title = element_text(size=34),
        axis.text = element_text( size=26, color="black"),
        legend.title = element_text(size=20),
        legend.text = element_text( size=16),
        axis.ticks.length = unit(5, "points"),
        legend.position="top",
        legend.justification = c(0, 0),
        axis.title.y = element_text(vjust=3),
        panel.grid.major = element_blank(),panel.grid.minor = element_blank())

prop_eux.plot.for.sum.shade.2 <- ggplot(prop_eux.pdp, aes(x=Age))+
  annotate(geom="rect", xmax=Inf, xmin=541, ymin=-Inf, ymax=Inf, fill="#C9DEE8", a
lpha=0.2)+
  annotate(geom="rect", xmax=541, xmin=418, ymin=-Inf, ymax=Inf, fill="#9DC2D6", a
lpha=0.4)+
  #annotate(geom="rect", xmax=418, xmin=-Inf, ymin=-Inf, ymax=Inf, fill="#619FBB",
alpha=0.4)+
  geom_ribbon(aes(ymin = min, ymax = max), alpha=1, fill="grey85", color="grey70",
size=.3)+
  geom_ribbon(aes(ymin = perc.25, ymax = perc.75), alpha=1, fill="grey60", color="
grey50", size=.5)+
  theme_bw()+
  coord_cartesian(xlim=rev(c(298.9,1000)), ylim=c(-.01,1.03),expand=FALSE)+
  scale_x_reverse(breaks=c(300,400,500,600,700,800,900,1000))+
  ylab("Proportion euxinic")+xlab("Time (Ma)")+
  theme(plot.margin = ggplot2::margin(.1,1,.3,1,"cm"),panel.border = element_rect(
fill=NA,color="black", size=2,linetype="solid"),
```

```
        axis.ticks = element_line(size=1),
        axis.line = element_line(lineend = 'square'),
        axis.title = element_text(size=34),
        axis.text = element_text( size=26, color="black"),
        legend.title = element_text(size=20),
        legend.text = element_text( size=16),
        axis.ticks.length = unit(5, "points"),
        legend.position="top",         legend.justification = c(0, 0),
        axis.title.x = element_blank(),
        axis.text.x = element_blank(),
        axis.title.y = element_text(vjust=3),
        panel.grid.major = element_blank(),panel.grid.minor = element_blank())

TOC.plot.for.sum.shade.2 <- ggplot(TOC.pdp, aes(x=Age))+
  annotate(geom="rect", xmax=Inf, xmin=541, ymin=-Inf, ymax=Inf, fill="#C9DEE8", a
lpha=0.2)+
  annotate(geom="rect", xmax=541, xmin=418, ymin=-Inf, ymax=Inf, fill="#9DC2D6", a
lpha=0.4)+
  #annotate(geom="rect", xmax=418, xmin=-Inf, ymin=-Inf, ymax=Inf, fill="#619FBB",
alpha=0.4)+
  geom_ribbon(aes(ymin = min, ymax = max), alpha=1, fill="grey85", color="grey70",
size=.3)+
  geom_ribbon(aes(ymin = perc.25, ymax = perc.75), alpha=1, fill="grey60", color="
grey50", size=.5)+
  theme_bw()+
  coord_geo(xlim=rev(c(298.9,1002)), expand=FALSE, ylim=c(-0.04,8.3),
            pos = as.list(rep("bottom", 1)),
            abbrv=list(TRUE),
            dat = list(periods.edit),
            height = list(unit(2, "lines")),
            bord=list(c("left", "bottom", "right")), lwd=as.list(c(1)), size=8)+
  scale_x_reverse(breaks=c(300,400,500,600,700,800,900,1000))+
  ylab("TOC (wt%)")+xlab("Time (Ma)")+
  theme(plot.margin = ggplot2::margin(.1,1,.3,1,"cm"),panel.border = element_rect(
fill=NA,color="black", size=2,linetype="solid"),
        axis.ticks = element_line(size=1),
        axis.line = element_line(lineend = 'square'),
        axis.title = element_text(size=34),
        axis.text = element_text( size=26, color="black"),
        legend.title = element_text(size=20),
        legend.text = element_text( size=16),
        axis.ticks.length = unit(5, "points"),
        legend.position="top",
        legend.justification = c(0, 0),
        axis.title.y = element_text(vjust=3),
        panel.grid.major = element_blank(),panel.grid.minor = element_blank())

summary.plot.shade.2 <- ggarrange2(Mo.plot.for.sum.shade.2, prop_eux.plot.for.sum.
shade.2, U.plot.for.sum.shade.2, TOC.plot.for.sum.shade.2, ncol=2, heights=c(1,1))
```




```
ggsave("Figure 2 Partial Dependence Plot with shading part 2 20240207 100 iteratio
ns.pdf", summary.plot.shade.2, height=14, width=26)
```