Data Analysis of UCE intersects

Cody Raul Cardenas

02.2023

Using same pipeline as before, but using JUST pterostichus melenarus

We need to import the intersect data from our data. Here is what our data looks like:

- Scaffold: the scaffold or chromosome queery matched too
- qstart: query start
- qend: query endsw
- type: genomic feature type
- seqname: scaffold or chromosome of reference
- seqtart: reference feature start
- seqerence: reference feature end
- attribute: if exonic, exonic feature given. Mainly concerned about transcript ID here.

```
scaffold    qstart    qend    query   type    seqname    seqstart   seqend  attribute
---
1   39310   39334   uce-127282_p11  intron  1   39281   39334
1   39334   39419   uce-127282_p11  exon    1   39335   39463   Parent=transcript:ENSMPTT00005003008;co
...
1   108104  108225  uce-146693_p5   intergenic  1   72314   146673
...
```

(how intersect was created can be found here: https://github.com/crcardenas/Adephaga_UCE/blob/main/workflow.md)

Load Library

```
library(tidyr) # data clean up
library(dplyr) # data cleanup
library(readr) # for importing
library(ggplot2) # for plotting
library(scales) # additional package for plotting
library(ggtext) # additional package for plotting
#library(psych) # for statistics, if necessary
# library(GenomicFeatures) # my not actually need this since we are doing our own thing
```

Load data

We have two files because of different GFF attribute fields for genes and exons. These can be joined later for manipulation, but may not need to be. There is no header information so we will need to add the header info I described above

```
d.intro_exon <- read_tsv(file="../Adephaga2.9-pterMadi2.introns-exons.out.intersect", col_names = F, na
```

```
## Rows: 6226 Columns: 9
## -- Column specification --------------------------------------------------------
## Delimiter: "\t"
## chr (5): X1, X4, X5, X6, X9
## dbl (4): X2, X3, X7, X8
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
d.inter_gene <- read_tsv(file="../Adephaga2.9-pterMadi2.intergenic-genentic.out.intersect", col_names =
```

```
## Rows: 4913 Columns: 9
## -- Column specification --------------------------------------------------------
## Delimiter: "\t"
## chr (5): X1, X4, X5, X6, X9
## dbl (4): X2, X3, X7, X8
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
colnames(d.intro_exon) <- c("scaffold","qstart","qend","query","type","seqname","seqstart","seqend","at
colnames(d.inter_gene) <- c("scaffold","qstart","qend","query","type","seqname","seqstart","seqend","at
d.intro_exon
```

```
## # A tibble: 6,226 x 9
##    scaffold  qstart    qend query          type   seqname seqst~1 seqend attri~2
##    <chr>      <dbl>   <dbl> <chr>          <chr>  <chr>     <dbl>  <dbl> <chr>
##  1 1          39310   39334 uce-127282_p11 intron 1         39280 3.93e4 <NA>
##  2 1          39334   39419 uce-127282_p11 exon   1         39335 3.95e4 Parent~
##  3 1          39334   39419 uce-127282_p11 exon   1         39335 3.95e4 Parent~
##  4 1          39339   39459 uce-127282_p12 exon   1         39335 3.95e4 Parent~
##  5 1          39339   39459 uce-127282_p12 exon   1         39335 3.95e4 Parent~
##  6 1          59597   59717 uce-258245_p11 exon   1         58920 5.98e4 Parent~
##  7 1          59637   59757 uce-258245_p12 exon   1         58920 5.98e4 Parent~
##  8 1        1011172 1011292 uce-71245_p11  exon   1       1010689 1.01e6 Parent~
##  9 1        1011172 1011292 uce-71245_p11  exon   1       1010709 1.01e6 Parent~
## 10 1        1011212 1011329 uce-71245_p12  exon   1       1010689 1.01e6 Parent~
## # ... with 6,216 more rows, and abbreviated variable names 1: seqstart,
## #   2: attribute
```

```
d.inter_gene
```

```
## # A tibble: 4,913 x 9
##    scaffold  qstart    qend query          type   seqname seqst~1 seqend attri~2
##    <chr>      <dbl>   <dbl> <chr>          <chr>  <chr>     <dbl>  <dbl> <chr>
##  1 1          39310   39419 uce-127282_p11 gene   1         34013 3.95e4 ID=gen~
##  2 1          39339   39459 uce-127282_p12 gene   1         34013 3.95e4 ID=gen~
##  3 1          59597   59717 uce-258245_p11 gene   1         58920 6.67e4 ID=gen~
##  4 1          59637   59757 uce-258245_p12 gene   1         58920 6.67e4 ID=gen~
```

```
##  5 1         108107  108227 uce-146693_p11 inter~ 1         72314 1.47e5 <NA>
##  6 1         108147  108256 uce-146693_p12 inter~ 1         72314 1.47e5 <NA>
##  7 1        1011172 1011292 uce-71245_p11  gene   1        982581 1.01e6 ID=gen~
##  8 1        1011212 1011329 uce-71245_p12  gene   1        982581 1.01e6 ID=gen~
##  9 1        1018435 1018555 uce-267689_p11 gene   1       1018430 1.04e6 ID=gen~
## 10 1        1018475 1018574 uce-267689_p12 gene   1       1018430 1.04e6 ID=gen~
## # ... with 4,903 more rows, and abbreviated variable names 1: seqstart,
## #   2: attribute
```

add new columns for each file

1) split out query column, one for UCE and one for UCE probe (UCE#_p##)

- query
- uce
- uce_probe

2) split out GFF attribute column if present:

- attribute
- transcript
- exon-id

```
df.intro_exon <- d.intro_exon %>% separate_wider_delim(cols = query,
                           delim = "_",
                           names = c("uce","probe")) %>%
  separate_wider_delim(cols = attribute,
                       delim = ";",
                       names = c("parent","constitutive","ID","rank","version")) %>%
  separate_wider_delim(cols = parent,
                       delim = ":",
                       names = c("parent","transcript")) %>%
  separate_wider_delim(cols = ID,
                       delim = "=",
                       names = c("gff_attribute","exon_id")) %>%
  mutate(query=paste(uce,probe, sep="_")) %>%
  select(scaffold, qstart, qend, query, uce, probe, type, seqname, seqstart, seqend, transcript, exon_i
df.intro_exon
```

```
## # A tibble: 6,226 x 12
##    scaffold qstart   qend query uce   probe type  seqname seqst~1 seqend trans~2
##    <chr>     <dbl>  <dbl> <chr> <chr> <chr> <chr> <chr>     <dbl>  <dbl> <chr>
##  1 1        3.93e4 3.93e4 uce-~ uce-~ p11   intr~ 1         39280 3.93e4 <NA>
##  2 1        3.93e4 3.94e4 uce-~ uce-~ p11   exon  1         39335 3.95e4 ENSMPT~
##  3 1        3.93e4 3.94e4 uce-~ uce-~ p11   exon  1         39335 3.95e4 ENSMPT~
##  4 1        3.93e4 3.95e4 uce-~ uce-~ p12   exon  1         39335 3.95e4 ENSMPT~
##  5 1        3.93e4 3.95e4 uce-~ uce-~ p12   exon  1         39335 3.95e4 ENSMPT~
##  6 1        5.96e4 5.97e4 uce-~ uce-~ p11   exon  1         58920 5.98e4 ENSMPT~
##  7 1        5.96e4 5.98e4 uce-~ uce-~ p12   exon  1         58920 5.98e4 ENSMPT~
##  8 1        1.01e6 1.01e6 uce-~ uce-~ p11   exon  1       1010689 1.01e6 ENSMPT~
##  9 1        1.01e6 1.01e6 uce-~ uce-~ p11   exon  1       1010709 1.01e6 ENSMPT~
## 10 1        1.01e6 1.01e6 uce-~ uce-~ p12   exon  1       1010689 1.01e6 ENSMPT~
```

```
## # ... with 6,216 more rows, 1 more variable: exon_id <chr>, and abbreviated
## #   variable names 1: seqstart, 2: transcript
```

```
df.inter_gene <- d.inter_gene %>% separate_wider_delim(cols = query,
                           delim = "_",
                           names = c("uce","probe")) %>%
  separate_wider_delim(cols = attribute,
                       delim = ";",
                       names = c("ID","biotype","geneID","version")) %>%
  separate_wider_delim(cols = biotype,
                       delim = "=",
                       names = c("gff_attribute1","biotype")) %>%
  separate_wider_delim(cols = ID,
                       delim = ":",
                       names = c("gff_attribute2","gene_id")) %>%
  mutate(query=paste(uce,probe, sep="_")) %>%
  select(scaffold, qstart, qend, query, uce, probe, type, seqname, seqstart, seqend, biotype, gene_id)
df.inter_gene
```

```
## # A tibble: 4,913 x 12
##    scaffold qstart   qend query uce   probe type  seqname seqst~1 seqend biotype
##    <chr>     <dbl>  <dbl> <chr> <chr> <chr> <chr> <chr>     <dbl>  <dbl> <chr>
##  1 1        3.93e4 3.94e4 uce-~ uce-~ p11   gene  1         34013 3.95e4 GTF2H1
##  2 1        3.93e4 3.95e4 uce-~ uce-~ p12   gene  1         34013 3.95e4 GTF2H1
##  3 1        5.96e4 5.97e4 uce-~ uce-~ p11   gene  1         58920 6.67e4 protei~
##  4 1        5.96e4 5.98e4 uce-~ uce-~ p12   gene  1         58920 6.67e4 protei~
##  5 1        1.08e5 1.08e5 uce-~ uce-~ p11   inte~ 1         72314 1.47e5 <NA>
##  6 1        1.08e5 1.08e5 uce-~ uce-~ p12   inte~ 1         72314 1.47e5 <NA>
##  7 1        1.01e6 1.01e6 uce-~ uce-~ p11   gene  1        982581 1.01e6 protei~
##  8 1        1.01e6 1.01e6 uce-~ uce-~ p12   gene  1        982581 1.01e6 protei~
##  9 1        1.02e6 1.02e6 uce-~ uce-~ p11   gene  1       1018430 1.04e6 protei~
## 10 1        1.02e6 1.02e6 uce-~ uce-~ p12   gene  1       1018430 1.04e6 protei~
## # ... with 4,903 more rows, 1 more variable: gene_id <chr>, and abbreviated
## #   variable name 1: seqstart
```

Genetic & intergenic

we need to make one more category that best characterizes as genetic, intergenic, or both

```
  scaffold  qstart    qend query         uce        probe type       seqname seqstart  seqend biotype
  <chr>      <dbl>   <dbl> <chr>         <chr>      <chr> <chr>      <chr>      <dbl>   <dbl> <chr>
1 1        1011217 1011282 uce-71245_p7  uce-71245  p7    gene       1         982581 1011637 protein_c
2 1        1011217 1011329 uce-71245_p8  uce-71245  p8    gene       1         982581 1011637 protein_c
3 1        1018428 1018429 uce-267689_p7 uce-267689 p7    intergenic 1        1011637 1018429 NA
4 1        1018429 1018545 uce-267689_p7 uce-267689 p7    gene       1        1018430 1035525 protein_c
```

see second UCE here, it is both intergenic and genetic... these are the only variables we are worried about right now.

1) first group by UCEs

2) then create a new column called category with mutate

- if the type column has more than one distinct type (e.g., both intergenic and genetic) give the column "intergenic_genetic"

3) ungroup the UCEs
4) now that we have this new category, we can use distinct to get the counts of the UCE features

```
category.inter_gene <- df.inter_gene %>%
  group_by(uce) %>%
  mutate(category = if (n_distinct(type) > 1) 'intergenic_genetic' else unique(type)) %>%
  ungroup() %>%
  select(scaffold, uce, probe, type, category)
category.inter_gene.all <- category.inter_gene %>% distinct(uce, .keep_all = T) %>% select(category) %>%
category.inter_gene.all
```

```
## # A tibble: 3 x 2
##   .                     n
##   <chr>             <int>
## 1 gene               1730
## 2 intergenic          633
## 3 intergenic_genetic   112
```

```
category.inter_gene.bychromosome <- category.inter_gene %>% group_by(scaffold) %>% distinct(uce, .keep_
```

```
## Adding missing grouping variables: `scaffold`
```

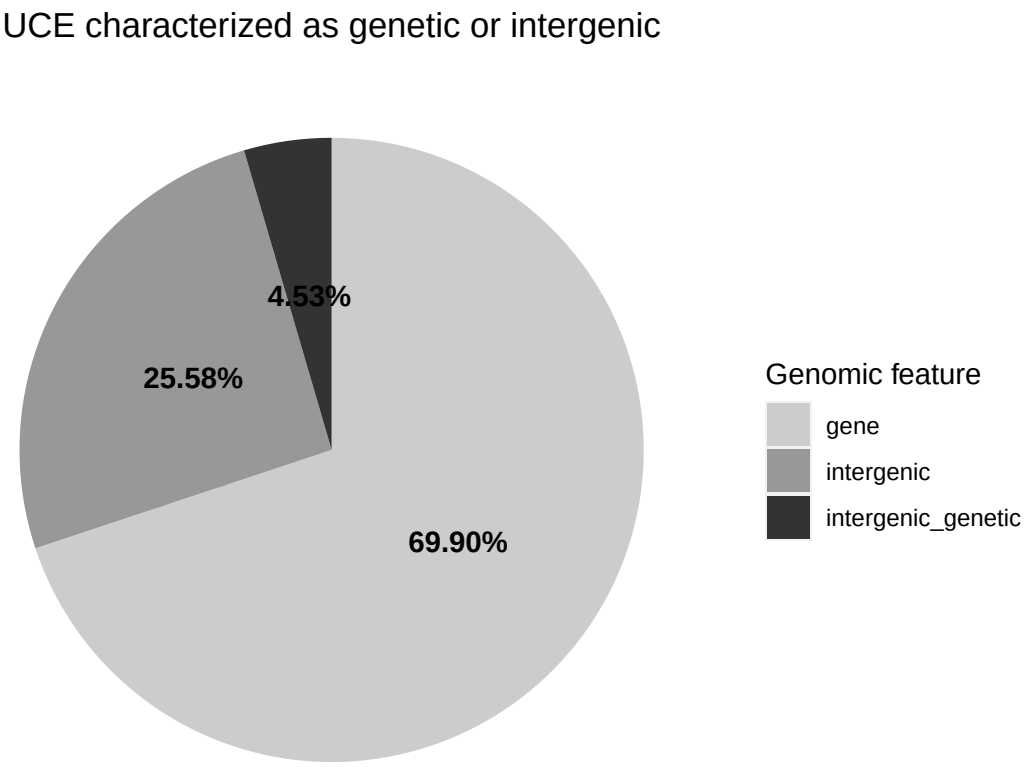
```
category.inter_gene.bychromosome
```

```
##          category
## scaffold gene intergenic intergenic_genetic
##       1    99         38                 9
##       10  105         27                 4
##       11  105         44                 4
##       12  144         35                 8
##       13   51         11                 3
##       14   52          9                 3
##       15   14          2                 1
##       16   26          8                 3
##       17   51         16                 4
##       18    7          1                 2
##       2   165         81                12
##       3   131         39                 8
##       4    76         34                 4
##       5   109         54                10
##       6   119         28                 9
##       7   147         52                 5
##       8    92         20                 6
##       9   118        104                11
##       X   119         30                 7
```

```
#create our dataframe
category.df.inter_gene.all <- data.frame(
  genomic_feature= category.inter_gene.all$.,
  uce_count = category.inter_gene.all$n) %>%
  mutate( proportion = round(uce_count / sum(uce_count), 4))
category.df.inter_gene.all
```

```
##      genomic_feature uce_count proportion
## 1               gene      1730     0.6990
## 2         intergenic       633     0.2558
## 3 intergenic_genetic       112     0.0453
```

```
# create a pie chart
pb.category.inter_gene.all <- category.df.inter_gene.all %>%
  ggplot(aes(x="", y=proportion, fill=reorder(genomic_feature,proportion))) +
  geom_bar(width = 1, stat = "identity") +
  coord_polar("y", start = 0) +
  scale_fill_grey() +
  theme(axis.title.y = element_blank(),
        axis.title.x = element_blank(),
        panel.border = element_blank(),
        panel.grid = element_blank(),
        axis.ticks = element_blank(),
        axis.text.x = element_blank(),
        panel.background = element_blank(),
        plot.title = element_text(hjust=0.5)) +
  geom_text(aes(label = percent(proportion, accuracy = 0.01), fontface=2),
            position = position_stack(vjust=0.5)) +
  ggtitle("UCE characterized as genetic or intergenic") +
  labs(fill="Genomic feature") +
  guides(fill=guide_legend(reverse=T))
pb.category.inter_gene.all
```



UCEs that map to gene features

get categorical information and summary like before

```
category.intro_exon <- df.intro_exon %>%
  group_by(uce) %>%
  mutate(category = if (n_distinct(type) > 1) 'intron_exon' else unique(type)) %>%
  ungroup() %>%
  select(scaffold, uce, probe, type, category)
category.intro_exon.all <- category.intro_exon %>% distinct(uce, .keep_all = T) %>% select(category) %>%
category.intro_exon.all
```

```
## # A tibble: 3 x 2
##   .               n
##   <chr>       <int>
## 1 exon         1019
## 2 intron         95
## 3 intron_exon   748
```

```
category.intro_exon.bychromosome <- category.intro_exon %>% group_by(scaffold) %>% distinct(uce, .keep_
```

```
## Adding missing grouping variables: `scaffold`
```

```
category.intro_exon.bychromosome
```

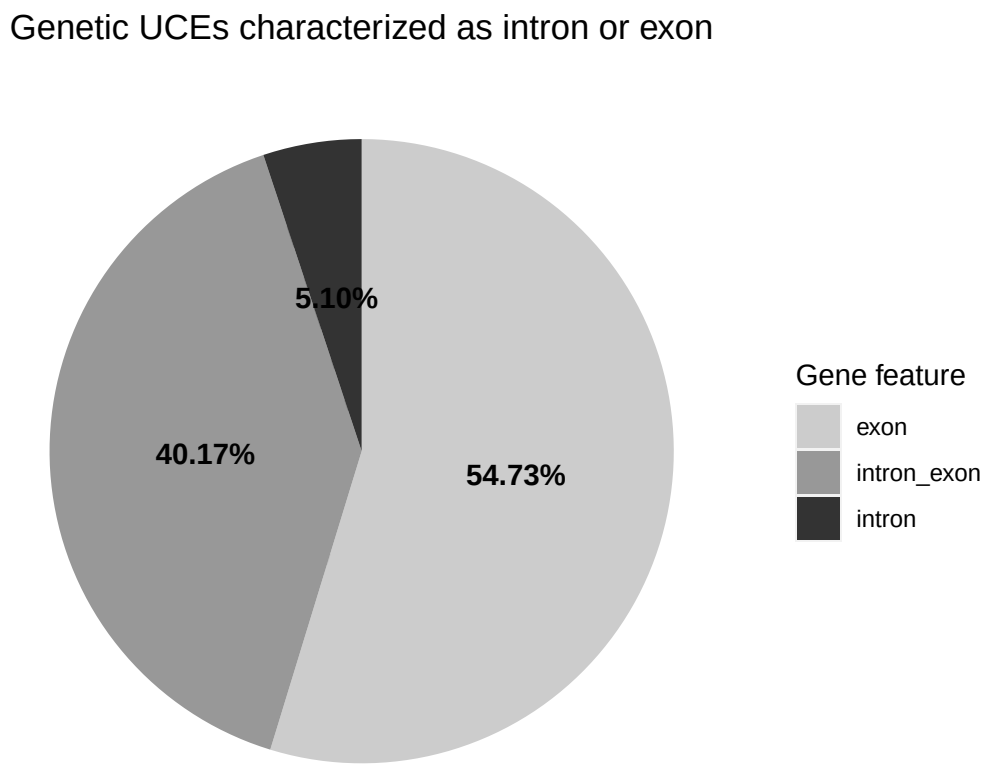
```
##          category
## scaffold exon intron intron_exon
##       1    58      5          45
##       10   61      3          45
##       11   60      4          47
##       12   94      6          53
##       13   29      2          23
##       14   28      2          26
##       15    7      0           8
##       16   19      1           9
##       17   28      4          23
##       18    6      0           3
##       2   102     14          64
##       3    79      6          57
##       4    41      3          37
##       5    69      9          44
##       6    64      5          59
##       7   103      3          47
##       8    55      3          42
##       9    64     11          55
##       X    52     14          61
```

```
#create our dataframe
category.df.intro_exon.all <- data.frame(
  genomic_feature= category.intro_exon.all$.,
  uce_count = category.intro_exon.all$n) %>%
  mutate( proportion = round(uce_count / sum(uce_count), 4)) %>%
  arrange(desc(proportion))
category.df.intro_exon.all
```

```
##   genomic_feature uce_count proportion
## 1            exon      1019     0.5473
## 2     intron_exon       748     0.4017
## 3          intron        95     0.0510
```

```
# create a pie chart
pb.category.intro_exon.all <- category.df.intro_exon.all %>%
  ggplot(aes(x="", y=proportion, fill=reorder(genomic_feature,uce_count))) +
  geom_bar(width = 1, stat = "identity") +
  coord_polar("y", start = 0) +
  scale_fill_grey() +
  theme(axis.title.y = element_blank(),
        axis.title.x = element_blank(),
        panel.border = element_blank(),
        panel.grid = element_blank(),
        axis.ticks = element_blank(),
        axis.text.x = element_blank(),
        panel.background = element_blank(),
        plot.title = element_text(hjust=0.5)) +
  geom_text(aes(label = percent(proportion, accuracy = 0.01), fontface=2),
```

```
            position = position_stack(vjust=0.5)) +
  ggtitle("Genetic UCEs characterized as intron or exon")  +
  labs(fill="Gene feature") +
  guides(fill=guide_legend(reverse=T))
pb.category.intro_exon.all
```



multi-UCE genes

next we need to characterize the number of UCEs per gene. Here is an example from the df.inter_gene with three different UCEs mapping to the same gene

scaffold	qstart	qend	query	uce	probe	type	seqname	seqstart	seqend	biotype
X	35120719	35120839	uce-123899_p11	uce-123899	p11	gene	X	35120216	35135278	protein_codi
X	35120759	35120879	uce-123899_p12	uce-123899	p12	gene	X	35120216	35135278	protein_codi
X	35129993	35130087	uce-17802_p12	uce-17802	p12	gene	X	35120216	35135278	protein_codi
X	35130007	35130127	uce-17802_p11	uce-17802	p11	gene	X	35120216	35135278	protein_codi
X	35131551	35131671	uce-123823_p12	uce-123823	p12	gene	X	35120216	35135278	protein_codi
X	35131591	35131711	uce-123823_p11	uce-123823	p11	gene	X	35120216	35135278	protein_codi

need to turn off warning, something weird happens when mutating the gene_id with the if logic

```
category.gene_uce <- df.inter_gene %>%
  group_by(gene_id) %>%
```

```
 mutate(gene_id =
          if (type == 'intergenic') 'intergenic' else gene_id) %>% # ignoring warning for now
  mutate(uce_count = case_when(n_distinct(uce) == 1 ~ 'n=1',
             n_distinct(uce) == 2 ~ 'n=2',
             n_distinct(uce) == 3 ~ 'n=3',
             n_distinct(uce) == 4 ~ 'n=4',
             n_distinct(uce) == 5 ~ 'n=5',
             n_distinct(uce) == 6 ~ 'n=6',
             n_distinct(uce) == 7 ~ 'n=7')) %>%
  ungroup() %>%
  select(scaffold, uce, probe, type, gene_id, uce_count)
category.gene_uce
```

```
## # A tibble: 4,913 x 6
##    scaffold uce        probe type       gene_id            uce_count
##    <chr>    <chr>      <chr> <chr>      <chr>              <chr>
##  1 1        uce-127282 p11   gene       ENSMPTG00005029508 n=1
##  2 1        uce-127282 p12   gene       ENSMPTG00005029508 n=1
##  3 1        uce-258245 p11   gene       ENSMPTG00005026164 n=1
##  4 1        uce-258245 p12   gene       ENSMPTG00005026164 n=1
##  5 1        uce-146693 p11   intergenic intergenic         <NA>
##  6 1        uce-146693 p12   intergenic intergenic         <NA>
##  7 1        uce-71245  p11   gene       ENSMPTG00005021500 n=1
##  8 1        uce-71245  p12   gene       ENSMPTG00005021500 n=1
##  9 1        uce-267689 p11   gene       ENSMPTG00005026854 n=2
## 10 1        uce-267689 p12   gene       ENSMPTG00005026854 n=2
## # ... with 4,903 more rows
```

```
category.gene_uce.all <- category.gene_uce %>% distinct(gene_id, .keep_all = T) %>% select(uce_count) %
category.gene_uce.all
```

```
## # A tibble: 5 x 2
##   .         n
##   <chr> <int>
## 1 n=1    1346
## 2 n=2     202
## 3 n=3      38
## 4 n=4       4
## 5 n=5       1
```

```
category.gene_uce.bychromosome <- category.gene_uce %>% group_by(scaffold) %>% distinct(gene_id, .keep_
```

```
## Adding missing grouping variables: `scaffold`
```

```
category.gene_uce.bychromosome
```

```
##          uce_count
## scaffold n=1 n=2 n=3 n=4 n=5
##       1   82  10   2   0   0
##       10  91   9   1   0   0
##       11  80   9   1   2   0
```

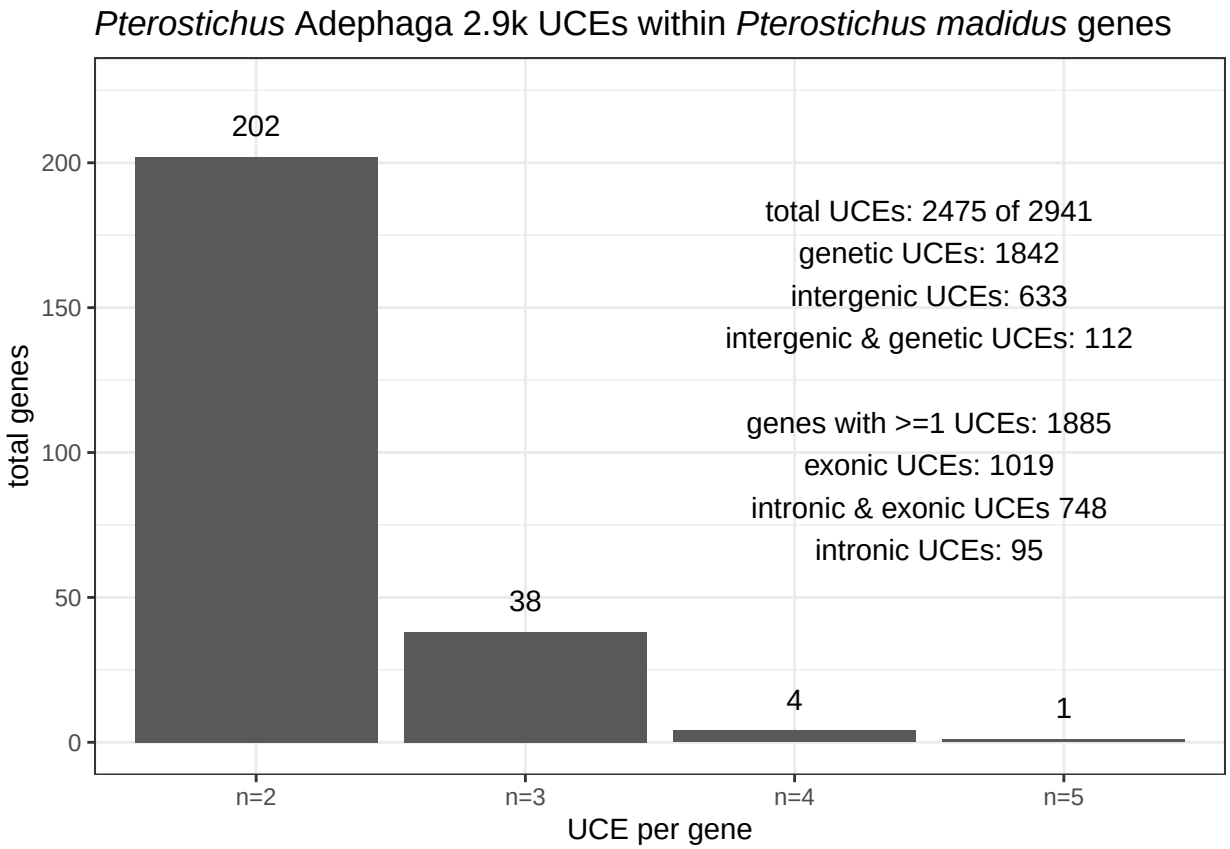
```
##       12 108  24   0   0   0
##       13  50   3   0   0   0
##       14  46   4   1   0   0
##       15  13   1   0   0   0
##       16  25   2   0   0   0
##       17  40   5   2   0   0
##       18   7   1   0   0   0
##       2  116  24   5   1   0
##       3  107  15   2   0   0
##       4   52  12   2   0   0
##       5   75  18   4   0   0
##       6   88  12   6   0   0
##       7  105  15   5   1   0
##       8   71  13   1   0   1
##       9   94  16   2   0   0
##       X   96   9   4   0   0
```

Next we create a data frame and plot our results, only plot N > 1 UCEs per gene

```
# create our data frame from our object class table
category.df.gene_uce.all <- data.frame(
  uce_per_gene = category.gene_uce.all$.,
  genes_total = category.gene_uce.all$n) %>%
  filter(uce_per_gene!='n=1')
category.df.gene_uce.all
```

```
##   uce_per_gene genes_total
## 1           n=2         202
## 2           n=3          38
## 3           n=4           4
## 4           n=5           1
```

```
# create barplot
bp.gene_uce.all <- category.df.gene_uce.all %>%
  ggplot(aes(x=uce_per_gene,y=genes_total)) +
  geom_bar(stat="identity") +
  annotate("text",x=3.5, y=125,
           label=
"total UCEs: 2475 of 2941\ngenetic UCEs: 1842\nintergenic UCEs: 633\nintergenic & genetic UCEs: 112\n
genes with >=1 UCEs: 1885\nexonic UCEs: 1019\nintronic & exonic UCEs 748\nintronic UCEs: 95") +
  geom_text( aes(label=genes_total), vjust=-1) +
  scale_y_continuous(limits=c(0,225)) +
  labs(x="UCE per gene",
       y="total genes",
       title= expression("*Pterostichus* Adephaga 2.9k UCEs within *Pterostichus madidus* genes")) +
  theme_bw() +
  theme(plot.title = ggtext::element_markdown())
bp.gene_uce.all
```



Something is off in the counting, things arent adding up. I think it is because of probes overlapping or duplicate matches...

```
df.inter_gene.duplicate_query <- df.inter_gene %>% janitor::get_dupes(query) # get duplicate probes fro

df.inter_gene.duplicate_query.genes <- df.inter_gene.duplicate_query %>% distinct(gene_id, .keep_all = 


df.inter_gene.duplicate_query.genes %>%
  group_by(query) %>%
  filter(length(query) == 1) %>%
  ungroup() %>%
  distinct(uce, .keep_all = T) # %>%
```

```
## # A tibble: 111 x 13
##    query  dupe_~1 scaff~2 qstart   qend uce   probe type  seqname seqst~3 seqend
##    <chr>    <int> <chr>    <dbl>  <dbl> <chr> <chr> <chr> <chr>     <dbl>  <dbl>
##  1 uce-1~       2 7       1.72e7 1.72e7 uce-~ p11   gene  7        1.72e7 1.72e7
##  2 uce-1~       2 X       3.62e7 3.62e7 uce-~ p11   gene  X        3.62e7 3.62e7
##  3 uce-1~       2 6       8.63e5 8.64e5 uce-~ p12   gene  6        8.21e5 8.64e5
##  4 uce-1~       2 1       1.34e6 1.34e6 uce-~ p11   gene  1        1.34e6 1.34e6
##  5 uce-1~       2 X       2.19e7 2.19e7 uce-~ p11   gene  X        2.19e7 2.19e7
##  6 uce-1~       2 11      2.88e7 2.88e7 uce-~ p11   gene  11       2.87e7 2.88e7
##  7 uce-1~       2 8       3.77e6 3.77e6 uce-~ p12   gene  8        3.75e6 3.77e6
##  8 uce-1~       2 8       8.85e5 8.85e5 uce-~ p11   gene  8        8.85e5 8.95e5
##  9 uce-1~       2 17      1.99e6 1.99e6 uce-~ p11   gene  17       1.99e6 1.99e6
```

```
## 10 uce-1~       2 14      4.35e6 4.35e6 uce-~ p11   gene  14       4.35e6 4.35e6
## # ... with 101 more rows, 2 more variables: biotype <chr>, gene_id <chr>, and
## #   abbreviated variable names 1: dupe_count, 2: scaffold, 3: seqstart
```

```
#  filter(type != 'intergenic')
```

OK, this makes sense, (so far) * 43 UCEs, represented by 2 probes, means there are 86 probes matching to the *SAME* area *

```
# uces_in_genes <- uce_count_by_gene2 %>% ggplot(aes(x=uce_count)) +
#   geom_bar(stat="count") +
#     annotate("text",
#              x=4.5, y=150,
#              label="intergenic UCEs: 770\ngenes with UCEs: 1934\n>1 UCE per gene: 257\nUCEs in exons: 1
#   geom_text(stat= 'count',
#             aes(label=..count..),
#             vjust=-1) +
#   scale_y_continuous(limits=c(0,225)) +
# #  theme_update(plot.title = element_text(hjust = 0.5)) +
#   labs(x="UCE per gene",
#        y="total genes",
#        title= expression("Adephaga 2.9k UCEs present in *Pterostichus madidus* genome")) +
#   theme_bw() +
#   theme(plot.title = ggtext::element_markdown())
# uces_in_genes
```

Get UCE biotype

```
df.inter_gene.biotype <- df.inter_gene %>%
  filter(biotype != 'protein_coding') %>%
  select(scaffold, qstart, qend, uce, probe, seqstart, seqend, biotype, gene_id)
df.inter_gene.biotype
```

```
## # A tibble: 1,628 x 9
##    scaffold  qstart    qend uce        probe seqstart  seqend biotype  gene_id
##    <chr>      <dbl>   <dbl> <chr>      <chr>    <dbl>   <dbl> <chr>    <chr>
##  1 1          39310   39419 uce-127282 p11      34013   39463 GTF2H1   ENSMPTG0~
##  2 1          39339   39459 uce-127282 p12      34013   39463 GTF2H1   ENSMPTG0~
##  3 1        1167005 1167103 uce-52790  p11    1154370 1167549 UHRF1BP1 ENSMPTG0~
##  4 1        1167023 1167138 uce-52790  p12    1154370 1167549 UHRF1BP1 ENSMPTG0~
##  5 1        1449153 1449273 uce-190318 p12    1437678 1526851 AGAP1    ENSMPTG0~
##  6 1        1449193 1449313 uce-190318 p11    1437678 1526851 AGAP1    ENSMPTG0~
##  7 1        2301076 2301196 uce-209397 p11    2297009 2304731 QSER1    ENSMPTG0~
##  8 1        2301116 2301236 uce-209397 p12    2297009 2304731 QSER1    ENSMPTG0~
##  9 1        3498674 3498794 uce-26138  p11    3497151 3499824 SLC35F6  ENSMPTG0~
## 10 1        3498714 3498834 uce-26138  p12    3497151 3499824 SLC35F6  ENSMPTG0~
## # ... with 1,618 more rows
```

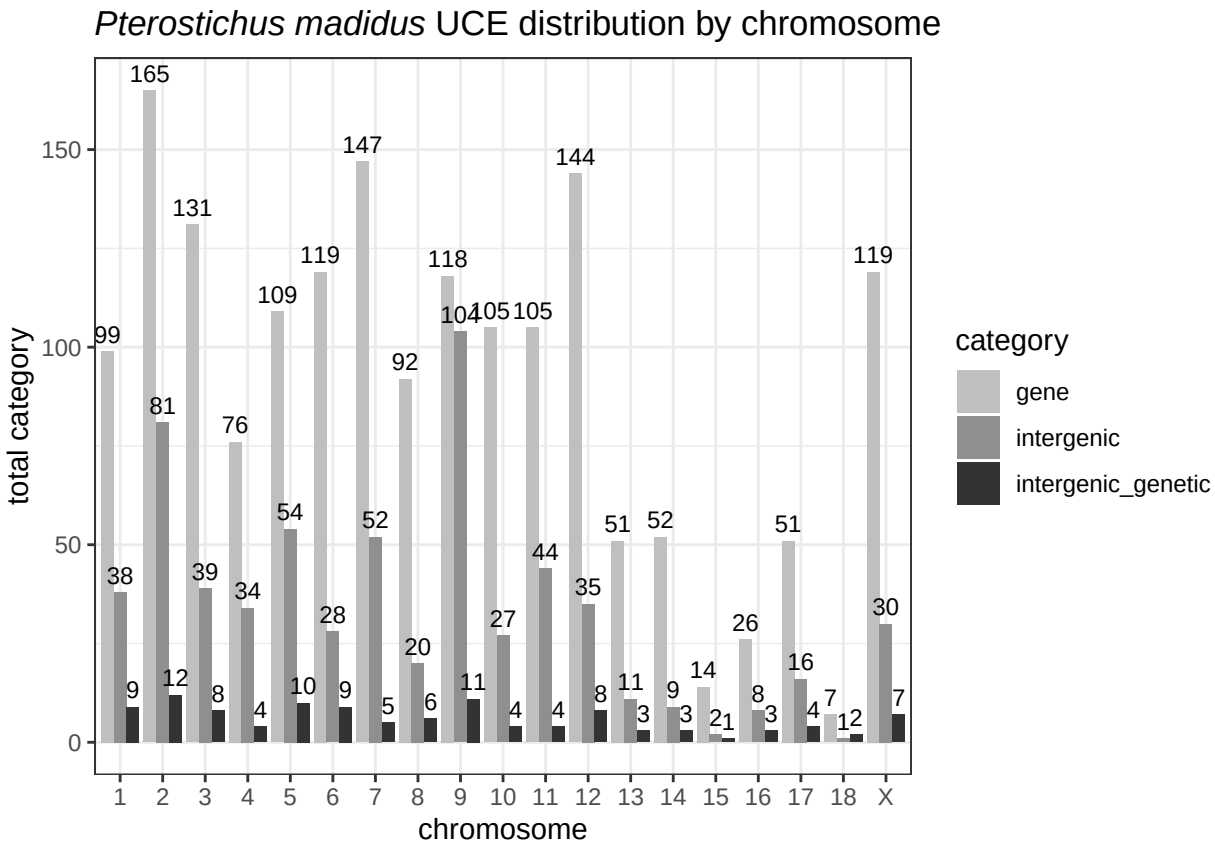
```
pterMadi2.UCE_biotype <- df.inter_gene.biotype %>% distinct(biotype, .keep_all = T) %>%
  select(uce, biotype)
pterMadi2.UCE_biotype
```

```
## # A tibble: 669 x 2
##    uce        biotype
##    <chr>      <chr>
##  1 uce-127282 GTF2H1
##  2 uce-52790  UHRF1BP1
##  3 uce-190318 AGAP1
##  4 uce-209397 QSER1
##  5 uce-26138  SLC35F6
##  6 uce-22907  ARFGAP3
##  7 uce-195490 MAP7
##  8 uce-6110   FKBP4
##  9 uce-63342  BEST3
## 10 uce-232826 CPSF6
## # ... with 659 more rows
```

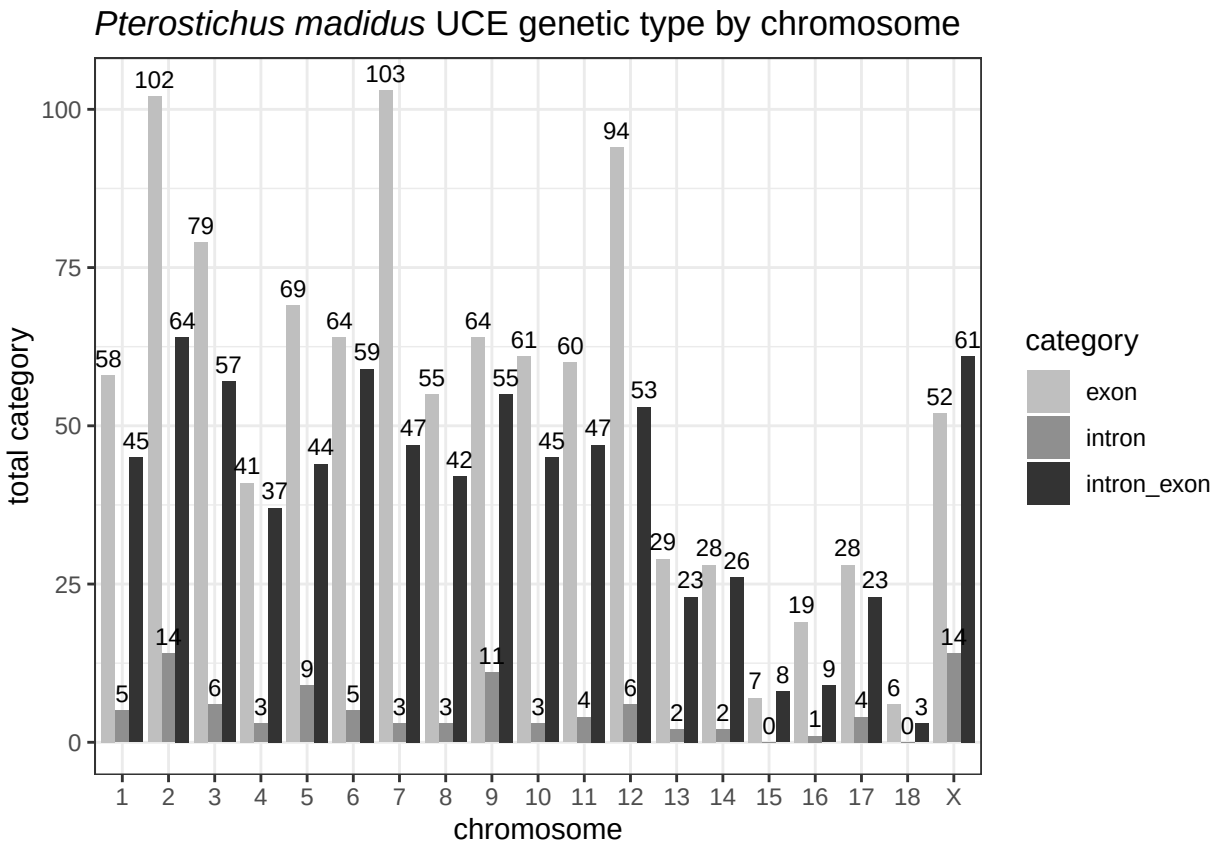
By chromosome plots

last thing for now is to get plots of distribution of UCEs by the chromosme

```
intergene.bychrom <- as.data.frame(category.inter_gene.bychromosome)
intergene.bychrom.plt <-  intergene.bychrom %>%
  ggplot(aes(x = factor(scaffold,
                        level = c('1','2', '3','4','5','6','7','8','9',
                                  '10','11','12','13','14','15','16','17','18','X')),
                        y = Freq, fill = category))+
  geom_col(position = "dodge",
           orientation = "x") +
  scale_fill_grey(start = 0.75, end = 0.2) +
  labs(x="chromosome",
       y="total category",
       title= expression("*Pterostichus madidus* UCE distribution by chromosome            ")) +
  theme_bw() +
  theme(plot.title = ggtext::element_markdown()) +
  geom_text( aes(label=Freq),
             position = position_dodge(width = 0.9),
             size = 3,
             vjust=-0.5)
intergene.bychrom.plt
```




```
introexon.bychrom <- as.data.frame(category.intro_exon.bychromosome)
introexon.bychrom.plt <- introexon.bychrom %>%
  ggplot(aes(x = factor(scaffold,
                        level = c('1','2', '3','4','5','6','7','8','9',
                                  '10','11','12','13','14','15','16','17','18','X')),
                        y = Freq, fill = category))+
  geom_col(position = "dodge",
           orientation = "x") +
  scale_fill_grey(start = 0.75, end = 0.2) +
  labs(x="chromosome",
       y="total category",
       title= expression("*Pterostichus madidus* UCE genetic type by chromosome")) +
  theme_bw() +
  theme(plot.title = ggtext::element_markdown()) +
  geom_text( aes(label=Freq),
             position = position_dodge(width = 0.9),
             size = 3,
             vjust=-0.5)
introexon.bychrom.plt
```




```
#category.gene_uce.bychromosome
```

Export data for chromosome painting

```
df.inter_gene %>%
  write.table(., './just_pterostichus_probes_intergenic.tsv', col.names = T, quote = F, sep='\t', row.na
```

```
sessionInfo()
```

```
## R version 4.1.2 (2021-11-01)
## Platform: x86_64-pc-linux-gnu (64-bit)
## Running under: Ubuntu 22.04.2 LTS
##
## Matrix products: default
## BLAS:   /usr/lib/x86_64-linux-gnu/blas/libblas.so.3.10.0
## LAPACK: /usr/lib/x86_64-linux-gnu/lapack/liblapack.so.3.10.0
##
## locale:
##  [1] LC_CTYPE=en_US.UTF-8    LC_NUMERIC=C            LC_TIME=en_GB.UTF-8
##  [4] LC_COLLATE=en_US.UTF-8  LC_MONETARY=en_US.UTF-8 LC_MESSAGES=en_US.UTF-8
##  [7] LC_PAPER=en_US.UTF-8    LC_NAME=C               LC_ADDRESS=C
## [10] LC_TELEPHONE=C          LC_MEASUREMENT=C        LC_IDENTIFICATION=C
```

```
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
## [1] ggtext_0.1.2  scales_1.2.1  ggplot2_3.4.1 readr_2.1.4   dplyr_1.1.0
## [6] tidyr_1.3.0
##
## loaded via a namespace (and not attached):
##  [1] Rcpp_1.0.10      highr_0.10       pillar_1.8.1     compiler_4.1.2
##  [5] tools_4.1.2      bit_4.0.5        digest_0.6.31    lubridate_1.8.0
##  [9] evaluate_0.20    lifecycle_1.0.3  tibble_3.1.8     gtable_0.3.1
## [13] pkgconfig_2.0.3  rlang_1.0.6      cli_3.6.0        rstudioapi_0.14
## [17] commonmark_1.8.1 parallel_4.1.2   yaml_2.3.7       xfun_0.37
## [21] fastmap_1.1.0    janitor_2.2.0    stringr_1.5.0    withr_2.5.0
## [25] knitr_1.42       xml2_1.3.3       generics_0.1.3   vctrs_0.5.2
## [29] hms_1.1.2        bit64_4.0.5      grid_4.1.2       tidyselect_1.2.0
## [33] gridtext_0.1.5   snakecase_0.11.0 glue_1.6.2       R6_2.5.1
## [37] fansi_1.0.4      vroom_1.6.1      rmarkdown_2.20   farver_2.1.1
## [41] purrr_1.0.1      tzdb_0.3.0       magrittr_2.0.3   ellipsis_0.3.2
## [45] htmltools_0.5.4  colorspace_2.1-0 labeling_0.4.2   utf8_1.2.3
## [49] stringi_1.7.12   munsell_0.5.0    markdown_1.5     crayon_1.5.2
```