RIP_index

Introduction

RIP causes C-to-T mutations, usually with CpA dinucleotides as the prefered context. Thus, RIP affected sequences are often seen as having:

- releative decrease in CpA and TpG dinucleotides
- corresponding increase in TpA

These changes can be used to identify RIP affected repeats by measuring the ratios of pre-RIP and post-RIP dinucleotides within repeats. Here we will sdklfslkdj the RIP indices of consensus sequences for each TE familysjalskdj

RIP Indices by Margolin *et al*

- Index 1: TpA/ApT
- Index 2: (CpA + TpG)/(ApC + GpT)

Scripts - Python

```
from Bio import SeqIO
from collections import Counter
import argparse

parser = argparse.ArgumentParser(
    description = "makes two files with info on dinucs and a file with RIP indices"
    )

parser.add_argument("--seqs", dest = "seqs", help = "put consensus sequences here
    (fasta)", required = True)
parser.add_argument("--outstem", dest = "outstem", help = "name specified for output
    files. Indices file will automatically be '[outstem]_indices'")

def count_dinucs(record):
    """count dinucleotides"""
    all_dinucs = []
    for i in range (len(record.seq)-1):
        all_dinucs.append(str(record.seq[i:i+2]))
    dinuc_counted = Counter(all_dinucs)
    return(dinuc_counted)

def calc_RIP_idx(dinuc_counted):
    try:
        RIP_idx_1 = round(dinuc_counted['TA']/dinuc_counted['AT'],4)
    except ZeroDivisionError:
        RIP_idx_1 = "NaN"
    try:
        RIP_idx_2 = round((dinuc_counted['CA']+dinuc_counted['TG'])/(dinuc_counted['AC']+dinuc_counted[
    except ZeroDivisionError:
        RIP_idx_2 = "NaN"
    return([RIP_idx_1, RIP_idx_2])


if __name__ == "__main__":
    args = parser.parse_args()
    seqs = SeqIO.parse(args.seqs, "fasta")
    R_idx_handle = open(args.outstem + "_indices.tsv", "w")
    R_idx_handle.write("family\tTE_class\tRIP_idx_1\tRIP_idx_2\n")

    with open(args.outstem + ".tsv", "w") as out:

        for rec in seqs:
            dn = count_dinucs(rec)
            family,TE_class = rec.id.split("#")
            idx_1, idx_2 = calc_RIP_idx(dn)

            R_idx_handle.write("{}\t{}\t{}\t{}\n".format(family, TE_class, idx_1, idx_2))
            biglist = ['AA','AT','AG','AC','TA','TT','TG','TC','GA','GT','GG','GC','CA','CT','CG','CC']
            denom = sum([n for (d, n) in dn.items() if "N" not in d])
            for d in biglist:
                percentage = dn[d]/denom * 100
                out.write("{}\t{}\t{}\t{}\t{:.4f}\n".format(family, TE_class,
                d, dn[d], percentage))
```

Usage

The script above can be called from commandline

```
python RIP_dinucs.py --seqs seqs.fa --outstem outstem
```

where seqs.fa is the user-specified consensus sequences (fasta), and outstem is the dedicated output file name.

Scripts - R

```
library(tidyverse)
```

```
## -- Attaching packages ---------------------------------------------------------------------------

## v ggplot2 3.3.2     v purrr   0.3.4
## v tibble  3.0.3     v dplyr   1.0.0
## v tidyr   1.1.0     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.5.0

## -- Conflicts ------------------------------------------------------------------------------------
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(ggplot2)
library(tidyr)
```

```r
ecl_RIP_idx <- read.table("~/analyses/RIP_indices/outputs/Ecl_RIP_indices.tsv", header = TRUE)
epo_RIP_idx <- read.table("~/analyses/RIP_indices/outputs/Epo_RIP_indices.tsv", header = TRUE)
ety_RIP_idx <- read.table("~/analyses/RIP_indices/outputs/Ety_RIP_indices.tsv", header = TRUE)
```

```r
ecl_RIP_idx$species <- "ecl"
epo_RIP_idx$species <- "epo"
ety_RIP_idx$species <- "ety"

#example of data
knitr::kable(head(ecl_RIP_idx))
```

family	TE_class	RIP_idx_1	RIP_idx_2	species
rnd-1_family-94	Unknown	0.7143	1.2821	ecl
rnd-1_family-79	Unknown	1.6961	0.4912	ecl
rnd-1_family-43	Unknown	1.1176	0.9412	ecl
rnd-1_family-53	Unknown	2.6000	0.2308	ecl
rnd-1_family-26	Unknown	2.3000	0.8519	ecl
rnd-1_family-89	Unknown	1.5357	0.6452	ecl

Data frame

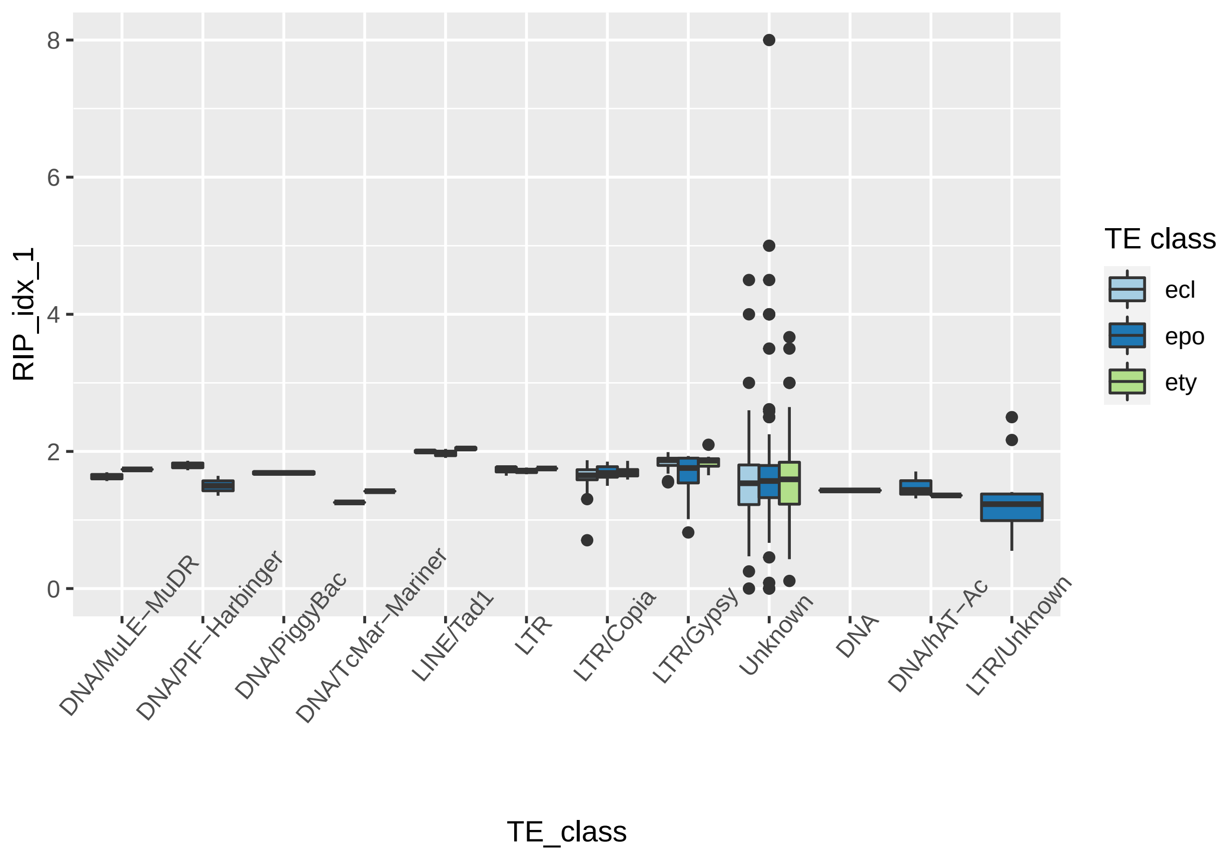
```r
idx_df <- rbind.data.frame(ecl_RIP_idx, epo_RIP_idx, ety_RIP_idx)
summary <- idx_df[,2:5]
summary <- summary %>% group_by(species, TE_class) %>% summarise(mean = mean(RIP_idx_1))
```

```
## `summarise()` regrouping output by 'species' (override with `.groups` argument)
```

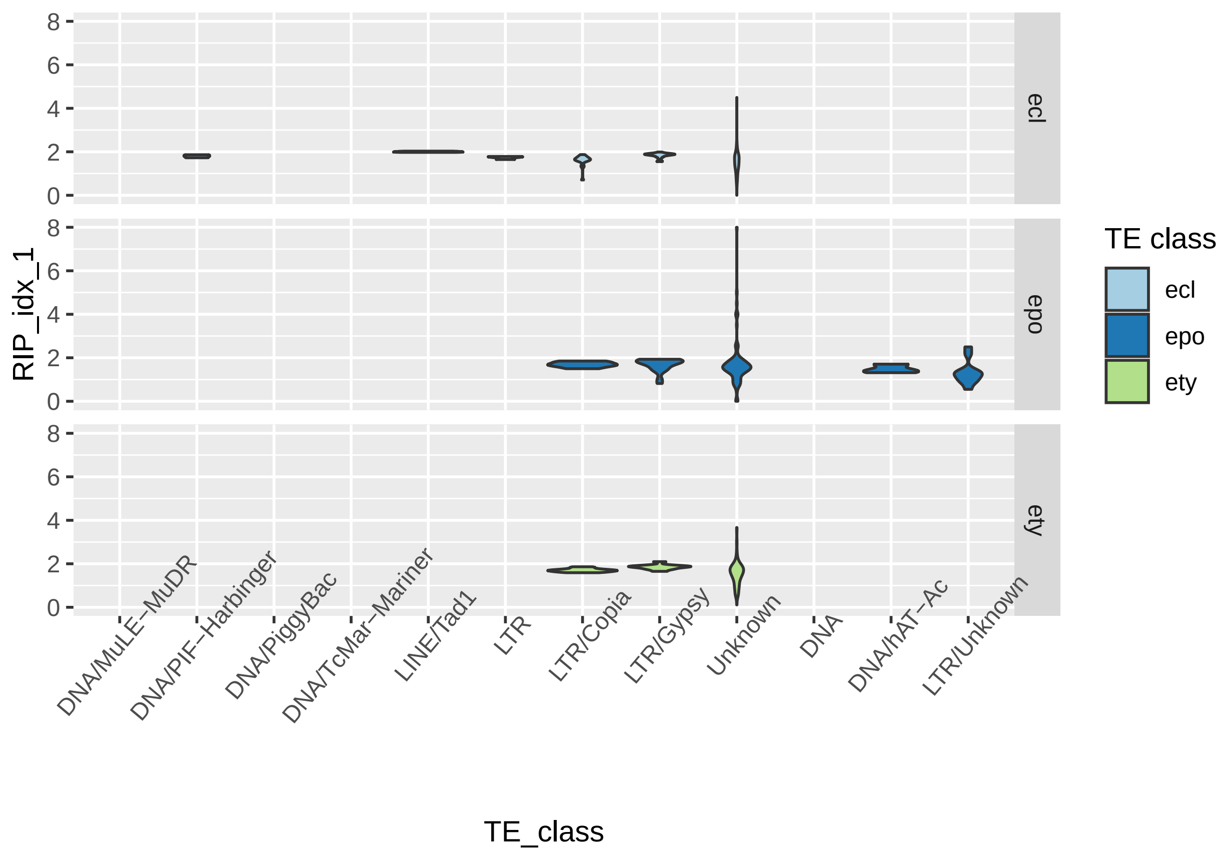
```r
idx_df = subset(idx_df, !(TE_class %in% c("Simple_repeat", "buffer", "rRNA", "Satellite")))
```

Visualisation

```r
ggplot(idx_df, aes(x= TE_class, y = RIP_idx_1, fill = species)) +
  geom_boxplot(position = "dodge") +  theme(axis.text.x = element_text(angle = 50)) +
  scale_fill_brewer("TE class", palette = "Paired")
```

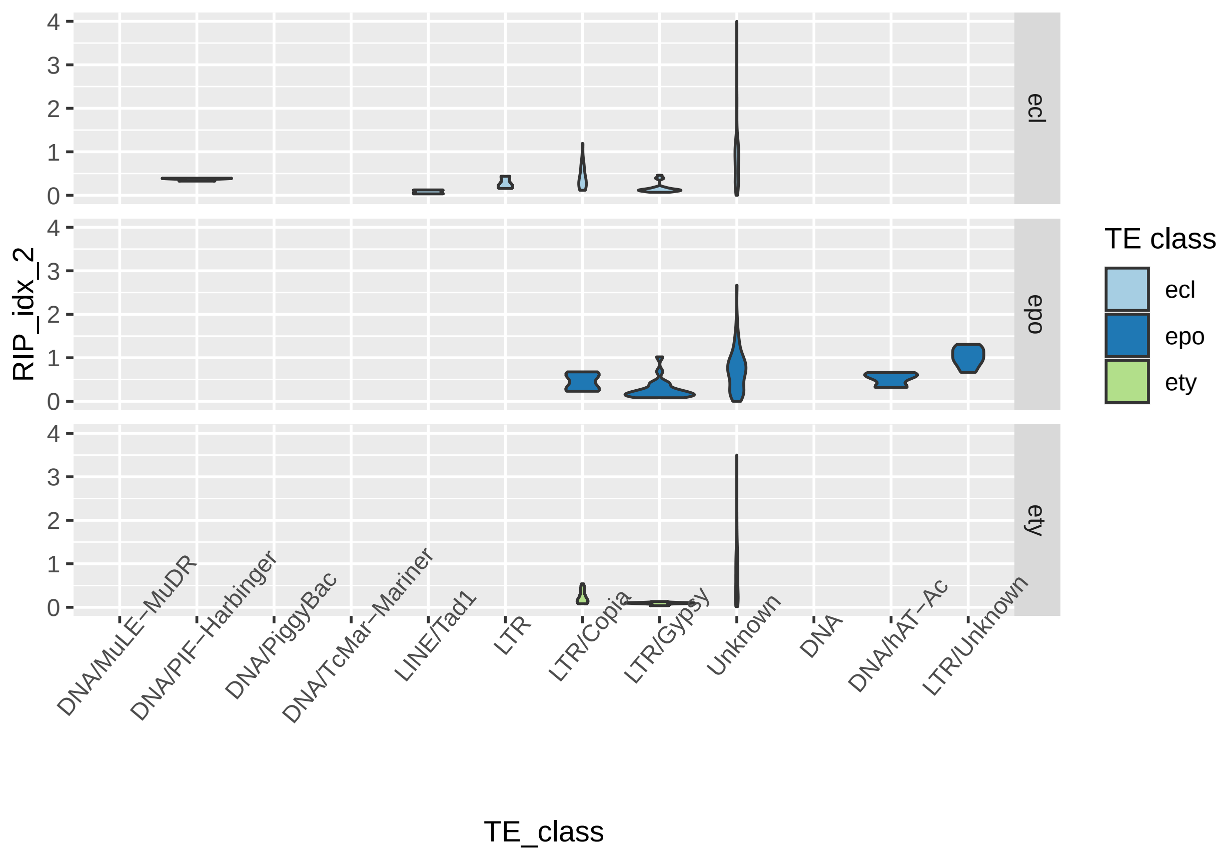


```
ggplot(idx_df, aes(x= TE_class, y = RIP_idx_1, fill = species)) +
  geom_violin() +  theme(axis.text.x = element_text(angle = 50)) + facet_grid(species ~ .) +
   scale_fill_brewer("TE class", palette = "Paired")
```



```
ggplot(idx_df, aes(x= TE_class, y = RIP_idx_2, fill = species)) +
  geom_violin() +  theme(axis.text.x = element_text(angle = 50)) + facet_grid(species ~ .) +
   scale_fill_brewer("TE class", palette = "Paired")
```

```
## Warning: Removed 1 rows containing non-finite values (stat_ydensity).
```



RIP_idx_2
TE_class
TE class
ecl
epo
ety
DNA/MuLE−MuDR
DNA/PIF−Harbinger
DNA/PiggyBac
DNA/TcMar−Mariner
LINE/Tad1
LTR
LTR/Copia
LTR/Gypsy
Unknown
DNA
DNA/hAT−Ac
LTR/Unknown