extract_contigs

March 14, 2018

1 Extracting target contigs

In order to extract the contigs representing your target sequences (the sequences that were being captured during the sequence capture process), you need to provide a fasta file containing the reference sequences for all loci of interest. Ususally all sequences of interest should be present in the file that was used to design the RNA baits for sequence capture. If you are using some standarad RNA bait library that was not specifically designed for your organism group/project (e.g. Ultraconserved Elements - UCEs), you can usually find the reference library on the webpage of the developer or in the respective publication. If all else fails, you can try to extract sequences of the same loci that you captured, for organisms that are closely related to your taxa, e.g. from NCBI GenBank.

The reference library should be in simple fasta format, containign one sequence per locus of interest. Here is an example of a reference library that was used in our example dataset of palms, extracted from Heyduk et al., 2016:

Citation: *Heyduk, K., Trapnell, D. W., Barrett, C. F., & Leebens-Mack, J. (2016). Phylogenomic analyses of species relationships in the genus Sabal (Arecaceae) using targeted sequence capture. Biological Journal of the Linnean Society, 117, 106–120.*

```
In [3]: %%bash
        head ../../data/raw/palm_reference_sequences.fasta
```

```
>Elaeis_1007_0
TGGGAGTCGCCGGGCATTTCTGGGATCTCCTCAAGCCCTACGCCCGGAACGAGGGCGTCGACTTCCTCCGGAACAAGCGCGTC
>Elaeis_1007_1
TCGAAGATGGGGGCGTTCCCGGTGTTCGTCGTTGACGGCGAGCCATCGCCGTTGAAGACGCAGGCAAGGATGGAGCGCTTCTT
>Elaeis_1007_2
GAACTCCTCGAAATCCTAGGGATGCCAGTTCTAAGAGCATGTGGTGAGGCTGAAGCCCTGTGTGCACAGTTAAATAGTGAAGC
>Elaeis_1007_3
GACCCATTTGAGTGCTACAACATATCAGATGTTGAAGCTGGTCTTGGTTTGAAGAGAAAACAAATGGTAGCCATTGCTCTTCT
>Elaeis_1007_4
GGTTATGTGAGGTTGGTAAAGGGGTTTTCCCTTTTTCAGAGGGAAGCATCAGTTTGGCCATGGATCCCCACATGCCTATTTCA
```

1.1 Find and extract all target contigs

Once you got your reference fasta files ready you are good to start with extracting the contigs of interest. For this purpose we want to create an overview over which contigs represent which reference locus in each sample. At the same time we also have to be somewhat selective and discard potential duplicates that match several loci. Let's check the function that helps you do this:

```
In [3]: %%bash
        source activate secapr_env
        secapr find_target_contigs -h
```

```
usage: secapr find_target_contigs [-h] --contigs CONTIGS --reference REFERENCE
                                  --output OUTPUT
                                  [--min-coverage MIN_COVERAGE]
                                  [--min-identity MIN_IDENTITY]
                                  [--regex REGEX] [--keep-duplicates]

Extract the contigs that match the reference database

optional arguments:
  -h, --help            show this help message and exit
  --contigs CONTIGS     The directory containing the assembled contigs in
                        fasta format.
  --reference REFERENCE
                        The fasta-file containign the reference sequences
                        (probe-order-file).
  --output OUTPUT       The directory in which to store the extracted target
                        contigs and lastz results.
  --min-coverage MIN_COVERAGE
                        The minimum percent coverage required for a match
                        [default=80].
  --min-identity MIN_IDENTITY
                        The minimum percent identity required for a match
                        [default=80].
  --regex REGEX         A regular expression to apply to the reference
                        sequence names as tags in the output table.
  --keep-duplicates     Use this flag in case you want to keep those contigs
                        that span across multiple exons.
```

Before running the script, you should take a look at the fasta headers in your reference fasta file. The script will use the fasta headers to extract the locus names. By default it takes the complete fasta header as the locus name. However, in many cases there is a lot of information in the fasta headers which you may not want to keep and translate as locus names (e.g. **>RPB2_intron23 Geonoma weberbaueri RNA polymerase II second largest subunit** should preferably translate into the locus name **RPB2_intron23** and discard all the rest of the header). If your fasta sequences are named consistently you can define a regular expression, using the `--regex` flag in order to only use the part of the string you are interested in. You can only define a single regex for the whole fasta file, which will be applied to all fasta headers in the same way.

Further, you can choose to add the `--keep-duplicates` flag, in order to also keep contigs which span across multiple loci. These will be extracted independently for each contig thhey match and may hence be present in several copies in the FASTA file containing your extracted contigs. If this flag is used a txt file with the duplicate informaiton is being printed into the output directory.

The sensitivity of the blast algorithm (LASTZ) can be altered with the flags `--min-coverage` and `--min-identity`. High values mean conservative matching requirements, while low values will return more matches but also possibly non-orthologous sequences.

Now let's run the script.

```
secapr find_target_contigs --contigs ../../data/processed/contigs/ --reference ../.
```

To get a first idea of the resulting matches, you can have a look at the file `match_table.txt` in the output folder.

```
In [1]: import pandas as pd
        table = pd.read_csv('../../data/processed/target_contigs/match_table.txt',
        table.head()
```

```
Out[1]:                   sample_1061  sample_1063  sample_1064  sample_1065  \
        Elaeis_1007_0             1            1            1            1
        Elaeis_1007_1             1            1            1            1
        Elaeis_1007_2             1            1            1            1
        Elaeis_1007_3             1            1            1            1
        Elaeis_1007_4             1            1            0            1

                          sample_1068  sample_1070  sample_1073  sample_1074  \
        Elaeis_1007_0             1            1            1            1
        Elaeis_1007_1             1            1            1            1
        Elaeis_1007_2             1            1            1            1
        Elaeis_1007_3             1            1            1            1
        Elaeis_1007_4             1            1            1            1

                          sample_1079  sample_1080  sample_1082  sample_1083  \
        Elaeis_1007_0             1            1            1            1
        Elaeis_1007_1             1            1            1            1
        Elaeis_1007_2             1            1            1            1
        Elaeis_1007_3             1            1            1            1
        Elaeis_1007_4             1            1            1            1

                          sample_1085  sample_1086  sample_1087  sample_1140  sample_1
        Elaeis_1007_0             1            1            1            1
        Elaeis_1007_1             1            1            1            1
        Elaeis_1007_2             1            1            1            1
        Elaeis_1007_3             1            1            1            1
        Elaeis_1007_4             1            1            1            1
```

Those fields containing a '1' indicate that a unique match was extracted from the contig sequences for the respective exon and sample. If the output reveals a very low harvest of target sequences, you can try to reduce the values for the flags `--min-coverage` and `--min-identity` in order to be more generous in the matching step. If on the other hand your output turns out to capture a lot of non-homologous sequences between the different samples (can be identified after the alignment step), you may want to turn up the values for these flags in order to be more conservative in your search.

The script also prints out summary stats in a textfile in the output folder:

```
In [59]: %%bash
         cat ../../data/processed/target_contigs/summary_stats.txt
```

```
Total number of samples: 17
Total number of targeted exons: 837

120 exons are shared by all samples.

sample_1061: 545 extracted contigs
sample_1063: 525 extracted contigs
sample_1064: 543 extracted contigs
sample_1065: 544 extracted contigs
sample_1068: 563 extracted contigs
sample_1070: 539 extracted contigs
sample_1073: 529 extracted contigs
sample_1074: 531 extracted contigs
sample_1079: 550 extracted contigs
sample_1080: 531 extracted contigs
sample_1082: 556 extracted contigs
sample_1083: 534 extracted contigs
sample_1085: 512 extracted contigs
sample_1086: 516 extracted contigs
sample_1087: 562 extracted contigs
sample_1140: 469 extracted contigs
sample_1166: 544 extracted contigs
mean: 534.882353 stdev: 21.605779
```

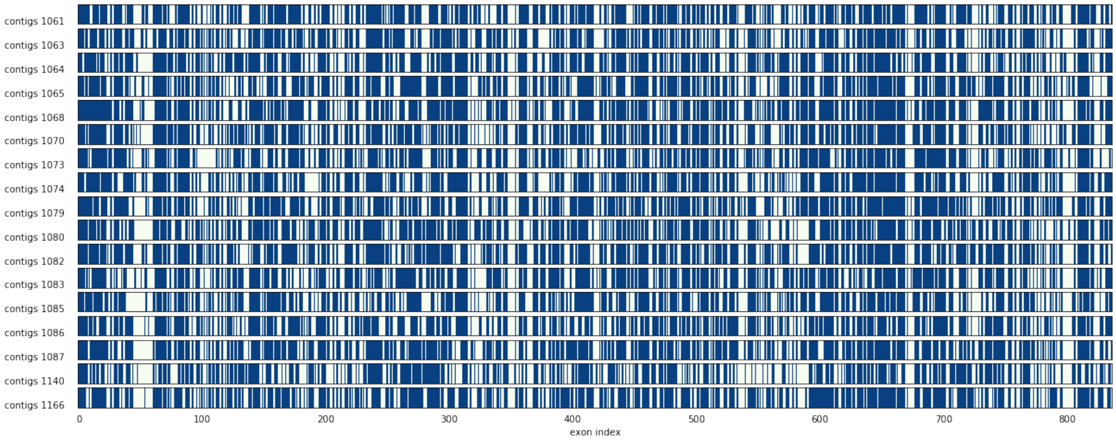
Let's plot the matrix for a better overview of our contig data:

```
In [1]: import sys
        sys.path.append("../../src")
        import plot_contig_data_function as secapr_plot

        match_table_path = '../../data/processed/target_contigs/match_table.txt'

        contig_yield = secapr_plot.plot_contig_yield(match_table_path)
        contig_yield
```

```
Out[1]:
```



Blue means presence and white absence of the respective sequence in the final contig file. SECAPR also prints a separate text file with an overview which exon index coressponds to which locus. It appears that most loci were recovered for most samples. Further there are gaps in some places for all samples, which could indicate that the bait sequences for that locus are not very suitable for our sample data of the genus *Geonoma*.

If you are satisfied with your contig yield you are ready to continue to the alignment step.

Previous page | Next page