locus_selection

March 14, 2018

1 Locus selection

When inspecting the results of the reference-based assembly you may find that many loci are not covered by many or any reads. In that case you may want to make a selection of loci with good coverage and continue to only work with these loci for downstream analyses. `secapr` has a function that helps you to extract loci with good coverage called `locus_selection`.

```
In [1]: %%bash
        source activate secapr_env
        secapr locus_selection -h
```

```
usage: secapr locus_selection [-h] --input INPUT --output OUTPUT [--n N]
                              [--read_cov READ_COV]

Extract the n loci with the best read-coverage from you reference-based
assembly (bam-files)

optional arguments:
  -h, --help           show this help message and exit
  --input INPUT        The folder with the results of the reference based
                       assembly or the phasing results.
  --output OUTPUT      The output directory where results will be safed.
  --n N                The n loci that are best represented accross all
                       samples will be extracted.
  --read_cov READ_COV  The threshold for what average read coverage the
                       selected target loci should at least have.
```

This function will compile the average read-coverage for each locus and sample and will select the `n` loci with the best coverage accross all samples. You can run this function simply by providing the folder containing the remapping results (`--input`) and stating how many of the best loci you want to extract (`--n`). You can additionally use the `--read_cov` flag to provide a custom read coverage threshold (default=3). Only loci with an average read-coverage above this threshold in all samples are being extracted. If not n loci are found that fulfil this threshold for all samples, the script will print a warning to the screen but still extract all loci fulfilling the requirement.

```
secapr locus_selection --input ../../data/processed/remapped_reads --output ../../d
```

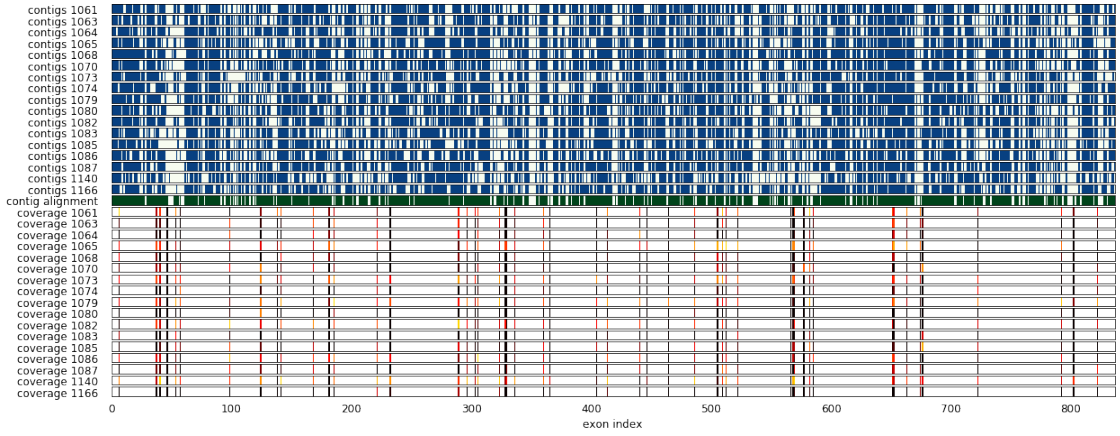
1.1 Which loci were selected?

Running the script as in the example command above will leave us with 50 loci that are present in all samples with an average read-coverage of more than 3. Below we visualize which loci were extracted.

```
In [2]: import sys
        sys.path.append("../../src")
        import plot_contig_data_function as secapr_plot

        contig_input_file = '../../data/processed/target_contigs/match_table.txt'
        alignment_folder = '../../data/processed/alignments/contig_alignments'
        read_cov_file_selected = '../../data/processed/selected_loci/overview_selec
        secapr_plot.plot_contigs_alignments_read_cov(contig_input_file,alignment_fo
```

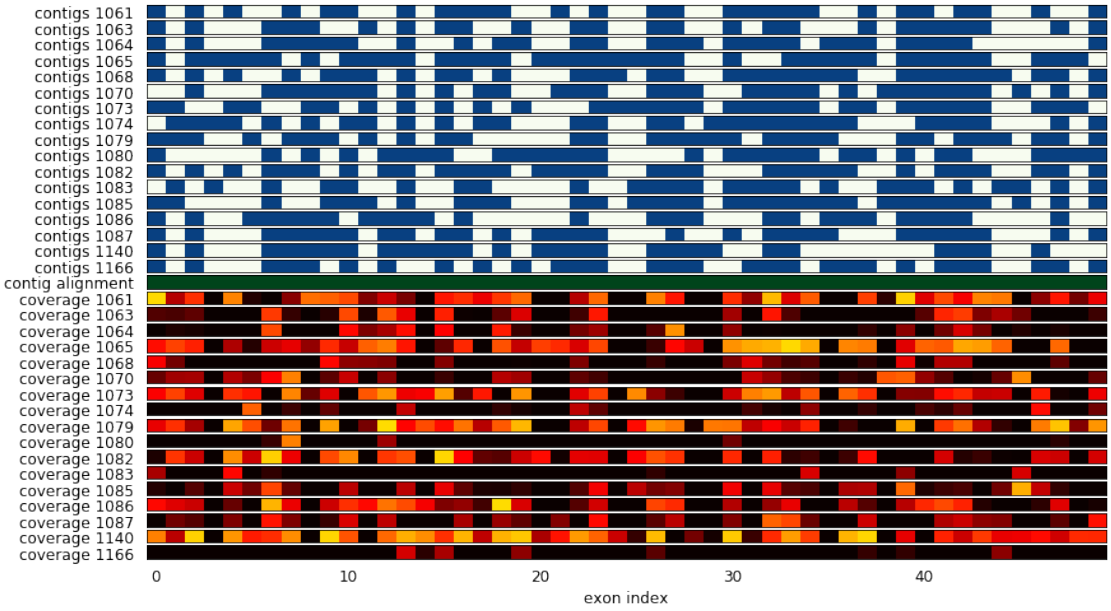
```
Out[2]:
```



We can also plot the results of the selection in a more detailed view, by showing the contig and read coverage of only the selected loci:

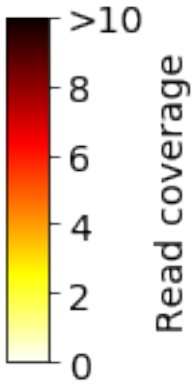
```
In [5]: selected_loci = secapr_plot.plot_contigs_alignments_read_cov(contig_input_f
        #selected_loci.savefig(os.path.join('/Users/tobias/GitHub/seqcap_processor/
```

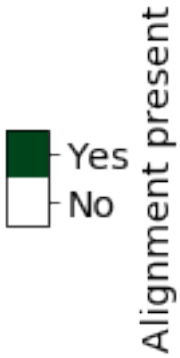
```
Reducing final matrix to selected loci.
```



And here are the corresponding legends:

```
In [3]: import matplotlib.pyplot as plt
        %matplotlib inline
        import warnings
        warnings.filterwarnings('ignore')
        legend = secapr_plot.plot_heatmap_legend(0,10,width=.75,height=2, font_size
        contig = secapr_plot.general_scale_bar(2,tick_labels=['No','Yes'],x0=.1,x1=
        align = secapr_plot.general_scale_bar(2,tick_labels=['No','Yes'],x0=.1,x1=.
        plt.show(contig)
        plt.show(align)
        plt.show(legend)
```





The plotting scripts also automatically output a text file that contains the corresponding locus name for each exon index in the plot. The text file is stored in the same folder as the input data, below we show the first lines of the file:

```
In [24]: import pandas as pd
         pd.read_csv('../../data/processed/selected_loci/locus_index_overview.txt',
```

Out[24]:

	0	1
0	0	Elaeis_1007_6
1	1	Elaeis_1052_0
2	2	Elaeis_1052_3
3	3	Elaeis_1119_1
4	4	Elaeis_1168_5
5	5	Elaeis_1171_3
6	6	Elaeis_136_2
7	7	Elaeis_150_1
8	8	Elaeis_164_6
9	9	Elaeis_1801_5

1.2 Aligning selected loci

Now after selecting the best loci we want to build Multiple Sequence Alignments (MSAs) from these loci. For this we can simply run the `secapr align_sequences` function:

```
secapr align_sequences --sequences ../../data/processed/selected_loci/joined_fastas
```

Just in case you are interested: Below you find the same data but plotted with a slightly different and more interactive plotting function.

```
In [7]: %%bash
        cd ../../
        python src/heatmap_plot.py
```

```
In [9]: %%html
        <div>
            <a href="https://plot.ly/~tobiashofmann/48/?share_key=wC4zjzzzXVpyZ4iR
            <script data-plotly="tobiashofmann:48" sharekey-plotly="wC4zjzzzXVpyZ4i
        </div>
```

```
<IPython.core.display.HTML object>
```

The read-coverage in the set of selected loci however is rather good for all/most samples:

```
In [10]: %%html
         <div>
             <a href="https://plot.ly/~tobiashofmann/50/?share_key=VZLFvEmzO1oJ3VGD
             <script data-plotly="tobiashofmann:50" sharekey-plotly="VZLFvEmzO1oJ3V
         </div>
```

```
<IPython.core.display.HTML object>
```

Previous page | Next page

```
In [ ]:
```