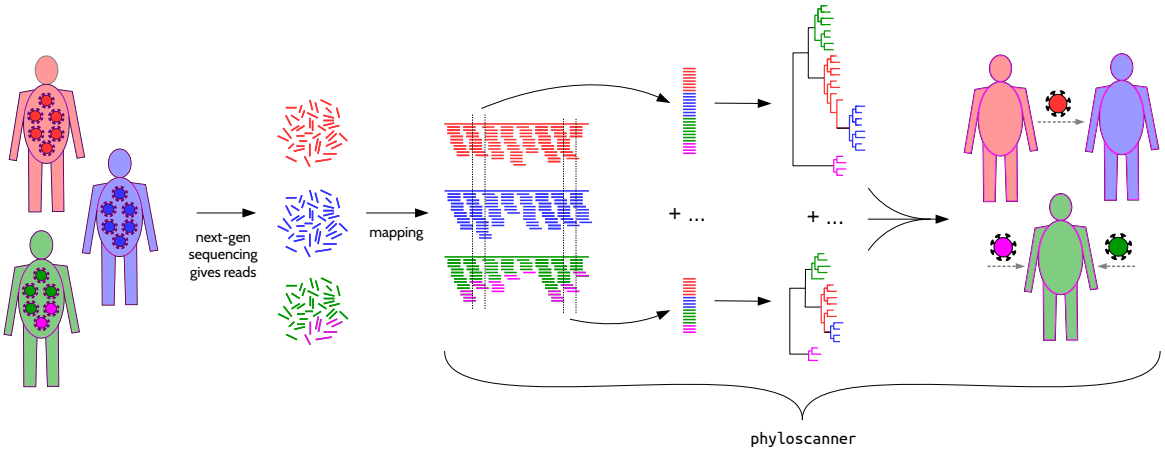
phyloscanner

Chris Wymant and Matthew Hall

Last updated June 2, 2017



`phyloscanner` analyses pathogen genetic diversity and relationships between and within hosts at once, in windows along the genome using mapped *reads* (fragments of DNA produced by next-generation sequencing), or using phylogenies with multiple sequences per host previously generated by the user via any method.

Part I
Generating within- and between-host phylogenies in windows along the genome, using mapped reads

Basic usage.
With the initial `$` traditionally indicating that we're on the command line, the basic command looks like

```
$ phyloscanner.py ListOfMyInputFiles.csv --windows 1,300,301,600,...
```

where

- `ListOfMyInputFiles.csv` is a plain-text, comma-separated-variable (csv) format file in which the first column is the bam files, the second column is the corresponding reference files, and an optional third column is *aliases* – things to rename each bam file to in `phyloscanner` output.
- the `--windows` option is used to specify an even number of comma-separated positive integers: these are the coordinates of the windows to analyse, interpreted pairwise, i.e. the first two are the left and right edges of the first window, the third and fourth are the left and right edges of the second window, ... i.e. in the above example we have windows 1-300, 301-600, ...

What windows should I choose for my own data?

I'm glad you asked. It's important. You might as well fully cover the genomic region you're interested in. That requires choosing where to start and where to end. If you're interested in the whole genome, the start is 1 and the end is the genome length, or more precisely the length of an alignment of the references in the bam files (this alignment is generated by `phyloscanner`, and it is with respect to this alignment that coordinates are interpreted by default; more on this later). You may know that your reads don't start right at the beginning of the genome. If this is the case, a good place to start your first window would be the genome position at which you start having reads. If your reads were generated by amplifying the sample using primers, the primers should have been trimmed from the reads as part of whatever bioinformatic pipeline produced your input bam files. (This can be done for example using `fastaq`, which is called as part of the `shiver` pipeline.) Then a sensible choice for the start of the first window would be the first position *after* the first primer, and a sensible choice for the end of the last window would be the last position *before* the last primer.

As well as choosing where your first window should start and where your last one should end, you need to choose how wide each single window is. If a window is very small, so little diversity is contained inside it (within or between samples) that the number of *unique* reads overlapping the window is small, hindering meaningful phylogenetics. If window width exceeds read length, then you will have no reads in the window, since we keep only reads fully overlapping the window. Somewhere between these two extremes therefore maximises the number of unique reads; to help you figure out what that is for your data, you can run `phyloscanner` with the `--explore-window-widths` option. This reports, for each in a list of window widths to try, how many unique reads are found for each bam and at each position along the genome. To summarise these read counts into a single value that varies with window width, you could use the mean, or the median; or you might be interested in a percentile lower than the 50th, if your concern is ensuring some minimal amount of diversity across all bams and all genomic positions. Up to you. (Note that some of the other options can affect how many reads you get in a window and so can affect what `--explore-window-widths` will tell you, namely the `--excision-coords`, `--merging-threshold`, `--min-read-count`, `--quality-trim-ends`, `--min-internal-quality`, and `--discard-improper-pairs` options. The first two can result in two or more unique reads being merged into one; the rest can simply discard some reads. You could choose values for the associated parameters immediately and then use `--explore-window-widths`, or else come back to `--explore-window-widths` later on once you've got the hang of `phyloscanner` and investigated the effect of those other options in your data.) NB power users might want to optimise their own measure of phylogenetic information as a function of window width; one of the first metrics to pop into your head might be the mean bootstrap of all nodes in the tree. That's not advised because within a sample there may be many very similar sequences, and the set of nodes connecting these may have poor boostrap support, but this is not something that ought to be penalised.

Also in theory you might be able to increase the window width until only a single read is found spanning the window in each patient; your bootstraps might then be great - between-host diversity is greater than that within-host - but you've thrown out all the within-host information.

How much should neighbouring windows overlap? A simple answer to this is zero, i.e. each window starts right after the previous one ends. e.g. 1-99, 100-199, 200-299, ... There's a longer discussion about this lower down, if you're interested.

NB wherever *read* and *read length* appeared in the discussion above, they should be substituted for *insert* and *insert size* if you have paired-read data AND the reads in a pair sometimes overlap AND you run `phyloscanner` with `--merge-paired-reads` to merge overlapping paired reads into a single longer read (see the cartoon below). A complication with this is that whereas read length is typically fixed within a sample, insert size has a distribution of different values. A window which is wider than twice the read length can never get any reads, because the reads in a pair need to overlap in order to be merged. So you have two choices. 1. Choose (read length) ¡ (window width) ¡ (twice the read length) Then you're restricted to the subset of read pairs that satisfy (window width) $\leq$ (insert size) $<$ (twice the read length) because only such pairs can overlap and fully span the window. The fraction of such reads in a sample is the integral of the unit-normalised insert size distribution between the two limits in the inequality above. 2. Choose (window width) $\leq$ (read length) Then you can have single reads contribute in addition to merged overlapping read pairs. But perhaps that window is too short; see the window-width discussion above.

Interpreting window coordinates

By default the references used for mapping (to produce the bam files), together with an extra set of references if specified with `--alignment-of-other-refs`, are all aligned together and window coordinates are interpreted with respect to the alignment (i.e. position n refers to the nth column of that alignment, which could be a gap for some of the sequences). This alignment can be found in the file `RefsAln.fasta` after running `phyloscanner`, should you want to inspect it and possibly run again. You can manually specify window coordinates with respect to this alignment, using the `--windows` option, or have windows automatically chosen using `--auto-window-params`, which attempts to minimise the affect of insertions and deletions in the references on your window width and overlap preferences. Alternatively, if you are using the `--alignment-of-other-refs` option to include extra references, you can use `--pairwise-align-to` to name one of these references to be a kind of *reference reference*: instead of aligning of all the bam file references to each other, they will be sequentially and separately pairwise-aligned to your named reference, and window coordinates are interpreted with respect to that named reference. Using the `--pairwise-align-to` option is expected to more stable than `--windows` or `--auto-window-params` if your bam file references are many and diverse, since pairwise alignment is easier than multiple sequence alignment. It also has the advantage that when running `phyloscanner` more than once with different bam files, the coordinates mean the same thing each time.

Part II

Analysing within- and between-host phylogenies