curate genome using ra2.py:

recommended usage:
$ ra2.py -i <genome.fa> -1 <forward_reads.fq> -2 <reverse_reads.fq> --add-Ns

* If doing additional curation afterwards, you may want to use --extend to extend the ends of scaffolds. However, since this can introduce redundancy, it is best not to use this unless doing additional curation to combine scaffolds with overlapping ends.

* Reads may be gzipped.

* See `$ ra2.py -h` for more info.

output will be in <genome>.curated directory:

* re_assembled.fa: curated genome

* re_assembled.report.txt: report (e = extended, n = error that was not fixed, f = error that was fixed, b = scaffold broken at error)

clean up ra2.py output:

* ra2.py produces fasta files with headers that are not compatible with some downstream applications.

clean fasta file:
$ fix_fasta.py <genome.curated/re_assembled.fa> | nr_fasta.py rename - > <genome.curated.fa>

filter mapping to look at stringently mapped reads:

* After running ra2.py you should re-map the reads to the genome sequence for visual inspection. Often it is helpful to remove poorly mapped reads from the mapping file.

filter mapping with same criteria used by ra2.py:
$ mapped.py -s <mapping.sam> -m 1 -p both -o <filtered_mapping.mm1-both.sam>

* This command allows for one mismatch in the read mapping. The “-p” option has to do with whether or not the mismatch requirement applies to one or both reads in a pair. In this case the requirement apples to both reads, unless one of the reads did not map.

* See `mapped.py -h` for more info.

collect mapped reads:

* For manual curation it is helpful to have all of the mapped reads.

get reads:
$ mapped.py -m False -p one -s <mapping.sam> -r > <mapped_reads.pe.fastq> 2><mapped_reads.se.fastq>

Notes:

* ra2.py saves mapped reads to memory and therefore can use a lot of memory when curating very large assemblies. I have had no problem running this on a single or even several genomes at one time. However, it should not be used on a metagenome assembly.

* Because this method relies on stringently mapped reads to identify assembly errors, low coverage genomes (<5x) may have an excessive number of predicted errors (i.e. false positives). In these cases, it may be better to not use the --add-Ns option and instead leave un-corrected errors as they originally were. This may also be the case for higher coverage genomes that have a lot of strain variation.