Annotation code description

1. Bakta annotation
2. Removal of bad COG annotations
3. Domain based annotation of anks and tprs
4. Annotation of some specific ankyrins
5. RAGE protein completeness checks
6. SpoT name processing
7. RAGE classification

Bakta annotation

Bakta annotation was chosen as fairly recent bacterial genome annotation package, which importantly does not require existing models for a species. It also integrates many different databases into it methodology.

This package enables the use of “expert proteins” to be provided as a .faa file and we make use of this for more accurate annotation of the O*rientia tsutsugamushi* genomes. To create this file all full-length genes annotated in the Giengkam et al (2023) paper were used, with the exception of genes with domain based names (e.g. ankyrin protiens, tetratricopeptide proteins and HATpase\_c domain containing protein). Additionally, some gene names were changed due to spelling mistakes or other naming issues (e.g. mmmA to mnmA and iepA to LepA).

This approach resulted in the correct annotation of most genes, however was insufficient for some proteins, often those with considerable sequence variability. Consequently, for a small group of proteins the required identity threshold was reduced from 90% to 80%. These are; scaA, scaC, tsa22 and tsa56, this approach was deemed reasonable for these as scaA and scaC are important host interaction proteins and likely subject to high selection pressure, and tsa22 and tsa56 are well established membrane proteins with lots of genetic diversity.

Another approach used to improve the annotation of cinA, scaA, scaB, scaD, scaE, secA, traA, traG, tsa22, tsa47 and tsa56 was to extract known sequences from NCBI and use these in the .faa file. These were extracted by searching for them and excluding partial or incomplete from the results (e.g. cinA AND txid784[Organism] NOT incomplete NOT partial), in some cases gene length was also used as a filter to remove the incomplete ones.

Adding new proteins

Modification of the .faa input file for bakta is the simplest way to add/change an annotation. To do this you need:

1. Full length sequences, the more the better especially for highly variable proteins. I would suggest using at least 8.
2. These sequences should be clustered to see the sequence identity between the genes. To be used in bakta normally this should be >90% and of similar lengths (80% similarity).
3. Ensure that there are no conflicting/contradictory gene names in the .faa file by performing (cd-hit) clustering. Closely related sequences should be checked as they may be other cases of the gene, and so the gene name should be changed in the file.

For genes with lower sequence identity, you can:

1. Modify the percentage identity in the .faa file to 80 (by including this in the fasta header; 80~~~80~~~80~~~).
2. Increase the number of sequences by identifying them in NCBI and including these in the .faa file. It is important these are complete copies.

However, this should be done with caution. Modification of the percentage identity should be performed very carefully and there should be good evidence that the gene is highly variable. It is also important to check that by doing this, genes are not mis-annotated.

Some genes cannot be handled like this if there is very low sequence identity (e.g. many of the ankyrin annotations in Giengkam et al (2023)). These should probably be annotated manually/separately.

COG gene removal

Bakta annotation occasionally mis-annotates some protein names but not the protein product. By examining the evidence for these mis-annotations it was determined they came from the COG database used in bakta.

To resolve this issue, a list of genes mis-named in the gene name (but not product) due to COG was created (all other databases were checked when generating this list). Then the COG identifier for these was extracted and a list was made in the `cog\_list.txt` file.

The script `1\_cog\_removal.py` is then used to remove gene names where the COG identifier is present. This is possible, as bakta only writes a COG identifier if it is used for the gene name or the product (e.g. it is not superseded by another database).

In the future if other genes are identified as problematic and as having COG issues, the COG identifier can be added to the `cog\_list.txt` file. Proceed with caution, these should be checked carefully first!

Domain based annotation of ANKs and TPRs

These domains are important in many host-pathogen interaction and so worth identifying. *Orientia* has a huge variety of these ank and tpr repeats in its genome. Consequently, we identify these repeats using a hmmer search and then annotate this information in the gbff files.

A threshold of -E 1E-10 was used, as (1) it is the threshold used in bakta and (2) it represents a high stringency threshold avoiding false positives.

The scripts over write any previous annotations for the hmmer hits, but skips proteins annotated as traG and bamD (which sometime contain tpr domains). In tests other annotations over written were mis-annotations, including pilW (*Orientia* doesn’t have a flagella).

The number of ankyrin repeats are included in the product description as a single hmmer model can be used for these proteins. However, for tetratricopeptides, there are many different models and so calculating the number of repeats becomes complicated.

Annotation of some specific ankyrins

Giengkam et al (2023) identified many *Orientia* anks. Clustering experiments showed some of these have high sequence identity and are conserved across the 8 genomes (investigated in the paper). These were the ones targeted for annotation.

For this a blast database was created and genes were tested against them. Ankyrin proteins covered in this are: Ank03, Ank08, Ank10, Ank11, Ank12, Ank20 and Ank24.

There is so much diversity of ankyrin proteins in *Orientia*, I think additional annotations should probably be avoided until someone clusters more genomes together.

RAGE protein completeness checks

To identify complete RAGEs, full-length; traA, traB, traC, traD, traE, traF, traG, traH, traI, traK1, traK2, traL, traN, traU, traV, traW and integrase must be present. Consequently, through the use of a blast script we check the overlap between the query and the subject, if this is >=95% then the sequence is considered complete and annotated as such in the gene product (e.g. complete traN) without overwriting the existing information.

This step is essential as there are many truncated copies of these genes in the genome (from degraded RAGEs).