# Synteny Plot quality control with SyntenyQC

Authors: Timothy Kirkwood, Eriko Takano and Rainer Breitling

# Supplementary Methods

## *S1: Collecting a BGC test data set and processing with SyntenyQC*

The MIBIG database (Zdouc et al., 2025) of verified Biosynthetic Gene Clusters (BGCs) was searched for entries from *Streptomyces coelicolor*. Each BGC protein set was used as a query in a cblaster (Gilchrist et al., 2021) search to identify putative BGC homologs. All hit neighbourhoods were required to have N unique hits to the BGC query, where N was either 5 or half the number of proteins in the query (whichever was larger), and all core biosynthetic proteins were required. For queries with 5 or fewer proteins, N was the number of proteins encoded by the BGC, and no specific proteins were required. Hits were restricted to those within Actinomycete genomes.

The commands used for cblaster, SyntenyQC Collect, and SyntenyQC Sieve are given in **Supplementary Data 1**. The fasta-format query files used for cblaster are given in **Supplementary Data 2**. The binary files output by cblaster are available at **Supplementary Data 3**. The GenBank files generated by SyntenyQC Collect and Sieve are available upon request. All supplementary files are available at [Tim-Kirkwood/SyntenyQC\_application\_note.](https://github.com/Tim-Kirkwood/SyntenyQC_application_note)

# Supplementary Figures

A black background with arrows and text

AI-generated content may be incorrect.

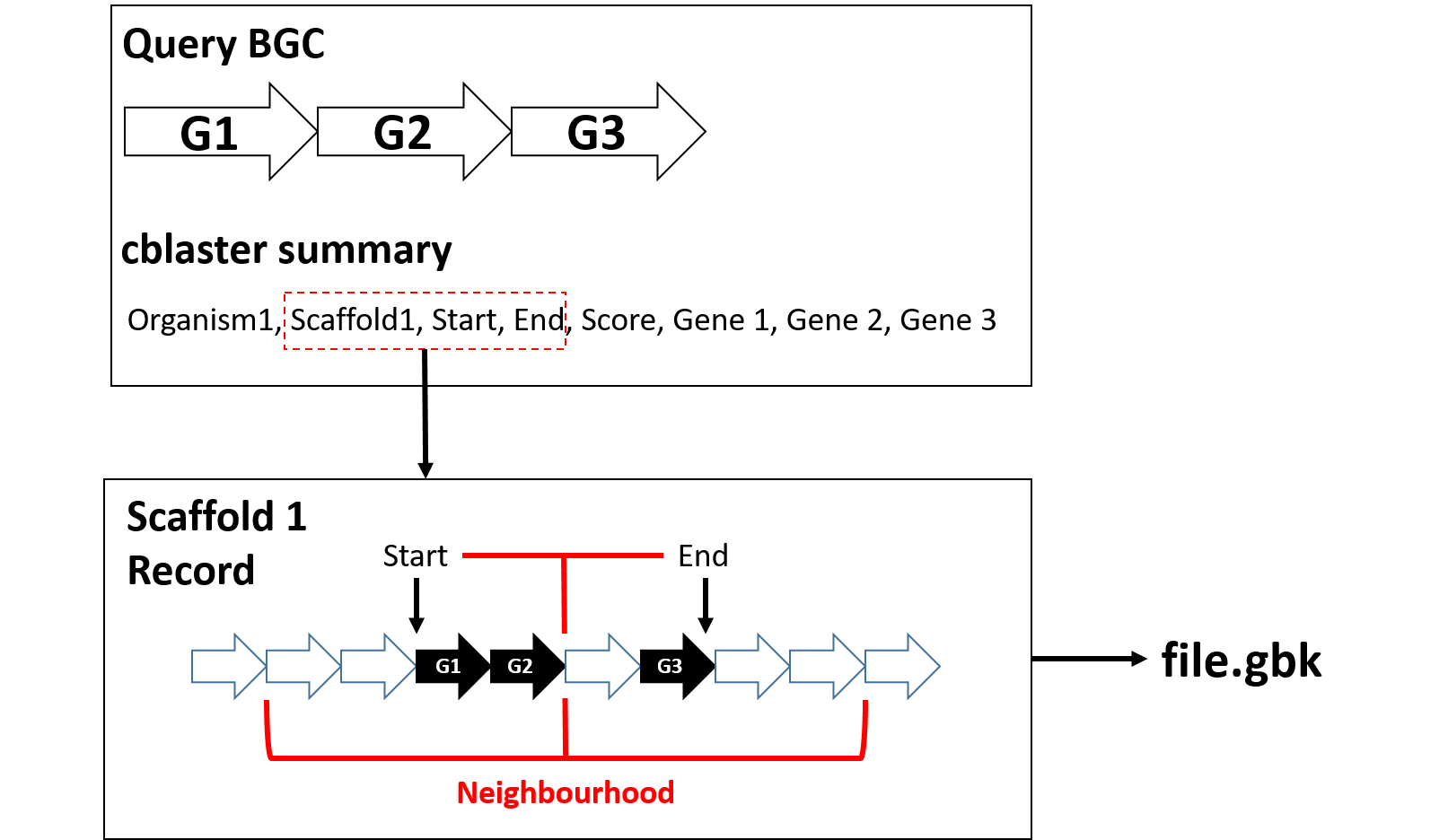
**Supplementary Figure 1: Integrating SyntenyQC into a synteny plot creation workflow.** First, cblaster is used to find records (i.e. genomes/contigs) containing a user-specified number of clustered hits to a user-supplied query. The loci of these homologs in each hit record are used to define a neighbourhood for each record, which is extracted using the SyntenyQC Collect subcommand. Files are named according to accession or organism name, facilitating automated annotation of the final synteny. Similar neighbourhoods (black braces) are filtered to remove redundant neighbourhoods, with what constitutes ‘similar’ specified by the user in the form of a similarity threshold. Finally, the collected, sieved neighbourhoods are fed into clinker to create the final synteny plot.

A screen shot of a computer screen

AI-generated content may be incorrect.

**Supplementary Figure 2: A synteny plot for the actinorhodin BGC.** Each track represents a separate genomic neighbourhood. The bottom track shows the genes within actinorhodin MIBIG entry BGC000194, which is used as the BGC reference in this plot. Coloured links indicate genes encoding proteins which fall within the same homology group. Black links indicate genes that encode proteins within the same homology group as SCO5087 and SCO5088, the only genes annotated as core biosynthetic within MIBIG entry BGC0000194. These genes encode the actinorhodin polyketide beta-ketoacyl synthase alpha and beta subunits respectively. The annotation “pBGC” (putative BGC) highlight neighbourhoods in different strains and/or genera, which share a clear set of genes with conserved homology and synteny.

(A) (B)

 A diagram of a diagram

AI-generated content may be incorrect.

**Supplementary Figure 3: The Collect subcommand. (A) The workflow.** A neighbourhood is downloaded from NCBI using the accession number supplied in the cblaster result file. The loci of the first (Gene 1) and last (Gene 3) cluster gene homologs in this neighbourhood are used to define a neighbourhood of a user-specified size, with a mid-point that lies between the loci. This neighbourhood is then written to a local GenBank file. **(B) Rejected records.** The neighbourhood is rejected if (TOP) its accession is not recognised, (MIDDLE) making a neighbourhood of a user-specified size would involve extending the neighbourhood beyond a contig edge – note this is optional, or (BOTTOM) if the hits identified by cblaster do not fall within a neighbourhood of user-specified size.

A diagram of arrows and arrows

AI-generated content may be incorrect.

**Supplementary Figure 4: Region similarity is non-transitive.** Boxes indicate neighbourhoods, arrows indicate genes, colours indicate homolog groups. A/B and B/C are fairly similar in terms of homolog composition (66%), but A and C are much less similar (33%).

A black background with a triangle and a black background

AI-generated content may be incorrect.

**Supplementary Figure 5: The Sieve subcommand.** A collection of input regions are subjected to an all-vs-all BLASTP to identify the number of reciprocal best hits between each pair of input regions. The neighbourhoods are then represented as a graph, where an edge is drawn between two nodes if their respective regions have a proportion of RBHs that exceed a user-defined threshold. NRBH is the number of reciprocal best hits between two node regions (A and B), NA and NB are the number of proteins in regions A and B respectively. T is a threshold proportion that is set by the user (typically 0.5 to 0.7). This graph is then pruned according to Algorithm 1. Following pruning, the remaining neighbourhoods are returned to the user.

A map of the constellation of the constellation

AI-generated content may be incorrect.

**Supplementary Figure 6: The neighbourhood graph.** Shown is the neighbourhood graph described in Supplementary Figure 5, with neighbourhoods as nodes and edges indicating that two neighbourhoods have a similarity score that exceeds the user threshold. Prior to pruning via Supplementary Algorithm 1, the neighbourhood graph is written to a dynamic HTML file, with hover labels describing the details associated with each node and edge. Node colour indicates the degree of a given node, and edge thickness indicates the similarity score magnitude.

# Supplementary Algorithms

**Algorithm 1**

**Data:** RBH graph

**Result:** Nodes from pruned RBH graph

**Procedure:**

**while** **max**(*node degrees* in *RBH graph*) > 0:

*delete nodes* = []

**for** *node* in *RBH graph*:

if *node* *degree* = **max**(*node degrees* in *RBH graph*):

*delete nodes* + *node*

*delete node* = random node from *delete nodes*

*RBH graph* = *RBH graph* - *delete node*

**return** *nodes* in *RBH graph*

# Supplementary Bibliography

Gilchrist, C. L. M., Booth, T. J., Van Wersch, B., Van Grieken, L., Medema, M. H., & Chooi, Y. H. (2021). cblaster: a remote search tool for rapid identification and visualization of homologous gene clusters. *Bioinformatics Advances*, *1*(1). https://doi.org/10.1093/BIOADV/VBAB016

Zdouc, M. M., Blin, K., Louwen, N. L., Navarro, J., Loureiro, C., Bader, C. D., Bailey, C. B., Barra, L., Booth, T. J., Bozhueyuek, K. A., Cediel-Becerra, J. D., Charlop-Powers, Z., Chevrette, M. G., Chooi, Y. H., D’Agostino, P. M., de Rond, T., Pup, E. D., Duncan, K. R., Gu, W., … Dillen, J. (2025). MIBiG 4.0: advancing biosynthetic gene cluster curation through global collaboration. *Nucleic Acids Research*, *53*(D1), 678–690. https://doi.org/10.1093/nar/gkae1115