

## A practical look at Genome Analysis

### Introduction:

In the past, genes were located, sequenced and analysed individually. Nowadays the technology exists to allow whole genomes to be sequenced and analysed as a single project. Analytical methods have improved over the years, although many of the basic approaches have survived. The greatest changes are those of scale. Similar strategies maybe employed today to look for genes as were used 15 years ago, but it is now necessary to devise ways to employ those strategies to identify all the genes of a given organism rather than a single gene in a few thousand base pairs.

Here we will work through some of the the essential steps required to detect and annotate the genes of a newly sequenced bacteria genome.


### Setting up:

To start, we need a suitable bacteria genome with which to work. The **Sanger Institute** generates and annotates such sequences. It is from their resources that the data for the exercise was copied. Go to<sup>1</sup>:

<http://www.ebi.ac.uk/genomes/>

and click on the **Bacteria** link down the left hand side of the page. Move down the page until you get to the species **Buchnera aphidicola** (the list is alphabetical). You should see that there are strains of a number of subspecies with lyrical Latin names. We are going to analyse the raw sequence of one of these (**Cinara cedri**) using the annotated genomes of 3 of the others to compute a gene model.

To get started therefore, you need to download the annotated genomes of **Acyrtosiphon pisum**, **Baizongia pistaciae** and **Schizaphis graminum** along with the raw sequence of **Cinara cedri**. As this would be tedious and not a worthy way to spend our practical time, I have done this for you.

Open the graphical view of your **Home** directory (second button down on the left of your screen ) and click twice on the **GENOME\_ANALYSIS** icon within, you should see 3 files whose name extension is **.embl**. These are the annotated genomes. Click on at least one to view its contents. The file is in the format of the *EMBL DNA sequence library* (hence the extension). Note particularly the lines beginning with **FT** (for **Feature Table**). These define and describe the regions predicted as genes or **CDS** (CoDing Sequences) as well as other features of interest. The actual sequence is right at the bottom of each file. Also double click the file called **Cinara\_cedri.fasta**. Note that there is no annotation beyond a single sparse line at the top. This is the sequence we will analyse. Specifically, we seek to add annotation predicting the positions of genes in the sequence. When you have seen enough, close both files and click on the **Home** icon at the top of the window display to return to a view of your **Home** directory.

### Predicting genes with glimmer:

To initially speculate where the genes might be in the genome of **Cinara cedri** we will use the programs of a package called **glimmer**. The purpose of **glimmer** is to generate a **VERY** rough guess as to where the protein coding genes in a newly sequenced genome might be. The efficacy of **glimmer** depends rather on the number of clues that can be provided. Here we have some fully and carefully annotated genomes, very similar to the one under investigation, that can be used to build a “model” of what a gene might look like. **glimmer** can still predict genes if such a rich source of clues is not available, but clearly, not as reliably.

This exercise suggests that you run glimmer 3 times, as follows:

- Ignoring the evidence of the previously annotated genomes and not really trying that hard!
- Ignoring the evidence of the previously annotated genomes and trying a bit harder.
- Finally! Using all the clues available from the previously annotated genomes, and trying hard.

<sup>1</sup> If this page is not already in view, it is your home page for the **firefox** browser and should also be directly available as **Genomes Pages** | **EBI** from the top of your browser window.

Once **glimmer** has finished, you will use more manual tools. Clearly these must have sophisticated Graphical User Interfaces (GUIs), to allow experience annotators to decide where **glimmer** has either predicted a gene wrongly or missed something. Generally, **glimmer** provides a very good starting point for manual gene prediction but is far from being good enough to be believed without close supervision.

If we were doing this regularly, we would probably spend some time setting things up nicely so that the programs could be run with minimal human intervention. **glimmer** does not have a nice interface. It does not need one. Its function is fully define before it is started and so it requires no human intervention throughout its execution. Here, you need to manage things more directly. I provide you with full instruction, but ...

Running glimmer is trivial, not interesting to watch and is messy and error prone if you are not used to the UNIX command line. Accordingly, we will do just a subset of what follows together in order to get to the pretty bits more speedily. I have given a blueish background colour to the instructions I intend to dance over.

Make and move into a console window and type in:

```
pwd
```

This command simply reports (Prints) the current (Working) folder (Directory) which should be:

```
/home/participant
```

Next type in the command:

```
ls
```

to list the files of your home directory. Amongst other items, you should see a directory (folder if you prefer) called:

```
GENOME_ANALYSIS
```

which is where the files you have just been inspecting from the graphical file manager reside. Type in:

```
cd GENOME_ANALYSIS
```

to change directory to **GENOME\_ANALYSIS** and then type in:

```
pwd
```

again, to prove you really have arrived, as intended, in your working directory for this practical which is:

```
/home/participant/GENOME_ANALYSIS
```

Type in:

```
ls
```

once more to list the files in **GENOME\_ANALYSIS**. No surprises I hope? Now we need to predict some genes. For a first shot, let us take a very simple route. Type in:

```
g3-from-scratch.csh Cinara_cedri.fasta from-scratch
```

which runs a **glimmer** package script called **g3-from-scratch.csh** which analyses the sequence in the file **Cinara\_cedri.fasta** and generates various output files all of whose names begin with **from-scratch**. You can list the output files with the command:

```
ls -lrt from-scratch.*
```

which translates to list all the files that begin **from-scratch** (**from-scratch\***) in a single column (**-l**) in reverse time order (**-rt**, essentially in the order of their creation). You should see:

```
from-scratch.longorfs
from-scratch.train
from-scratch.icm
from-scratch.predict
from-scratch.detail
```

2 OK, I cheated. The script is so quick that the files are often dated identically and so difficult to list in the order of creation. I have fiddled it. My list is correct, yours maybe differently ordered.

Essentially, what the script **g3-from-scratch.csh** has done is to run a series of programs from the **glimmer** package (the details need not concern us) to:

- 1) Identify all the **Open Reading Frames (ORFs)** i.e. regions between adjacent stop codons) of the genome under analysis that are significantly longer than one would expect to happen by chance and to store their positions in the file **from-scratch.longorfs**. Have a quick look at this file from the graphical file manager.
- 2) Extract the regions of the genome listed in **from-scratch.longorfs** and store them in the file **from-scratch.train**. Have a quick look at this file from the graphical file manager.
- 3) Build a mathematical model representing the features of the gene sequences stored in **from-scratch.train** and store this model in the file **from-scratch.icm**. It is not possible to easily take a sensible look at the model file as it is not a text file, so just believe me this time.
- 4) Finally, use the computed gene model to predict where genes might be in our “new” genome. The position of the predicted genes are stored in the 2 output files **from-scratch.predict** and **from-scratch.detail**. These files contain essentially the same information, but, as its name suggests, **from-scratch.detail** has a little more detail. Have a quick look at these 2 files.

Fine, but we can do a little better by running a slightly more sophisticated **glimmer** script thus:

```
g3-iterated.csh Cinara_cedri.fasta iterated
```

As I am sure you can guess, the plan is not wildly dissimilar. This time the output files all begin with **iterated**. So, to view the results files, type:

```
ls -lrt iterated.*
```

and you should be rewarded with the list:

```
iterated.longorfs
iterated.train
iterated.icm
iterated.run1.predict
iterated.run1.detail
iterated.upstream
iterated.coords3
iterated.motif
iterated.predict
iterated.detail
```

**g3-iterated.csh** does everything that **g3-from-scratch.csh** did and then, guided by the predicted gene positions (stored in **iterated.run1.predict**), copies the first few bases<sup>4</sup> of each predicted gene into the file **iterated.upstream**<sup>5</sup>. A program is then run to identify ribosome binding sites within the stored upstream regions<sup>6</sup> and to compute a model to represent their conservation. This model is a simple weight matrix and is stored in the file **iterated.motif**<sup>7</sup>.

Finally the genes are predicted again, this time taking into account matches with the ribosome binding site model as well as the gene model stored in the file **iterated.icm**. The results of the second, final, gene prediction are stored in **iterated.predict** and **iterated.detail**.

The major “improvement” offered by **g3-iterated.csh** relative to **g3-from-scratch.csh** is to take into account that there should be a believable ribosome binding site at the start of each gene<sup>8</sup>.

<sup>3</sup> Just in case you were wondering, **iterated.coords** is exactly the same as **iterated.run1.predict** with the first line removed. It is only required as one of the programs run by **g3-iterated.csh** is ridiculously picky.

<sup>4</sup> By default the first 25 bases are used.

<sup>5</sup> So named as the stored regions are “upstream” of the start position of each predicted gene.

<sup>6</sup> To be exact, the most common motif of length 6 within all the “upstream” regions is computed. It is reasonable to suppose that this motif is the ribosome binding site.

<sup>7</sup> The model is both textual and really is very simple. take a look and see if you can work out what it means.

<sup>8</sup> Also, based upon the evidence of the first few gene predictions, the relative likelihoods of the start codons **atg**, **gtg** and **tgg** are computed. These are used to improve the predictions of the start of the genes the second time around.

This is fine, but both of the scripts we have used build a model of “what a gene looks like” from crudely selected (open read frames that are long) regions of the genome that is being investigated. Well, it works quite well, but as we have fully analysed genomes of very similar organisms to hand, maybe we can do a little better. Why not use the carefully predicted gene regions of the similar organisms to compute a model representing what a gene should look like, and then to proceed after the fashion of **g3-iterated.csh**? There is no pre constructed **glimmer** script to do this, so you would need to make one yourself. For today, I have done this for you, but it is not difficult to edit the scripts distributed with **glimmer** to fit particular user requirements.

In order to get our customised script flying, you need a file containing the names of the fully annotated genomes downloaded from the Sanger Institute site. This can be constructed **with the command**:

```
ls *.embl > Embl.List
```

which when translated means list all the files whose name ends in **.embl** and instead of showing the list on the display screen, send it to (>) a file called **Embl.List**. So you should have a new file called **Embl.List** now. Have a look at it and if the contents are not what you expected, ask for elucidation.

You are ready to run the customised script **with the command**:

```
bmb.csh Embl.List Cinara_cedri.fasta bmb
```

which does almost the same as **g3-iterated.csh** except that instead of using the long open reading frames from the **Cinara cedri** genome to form a training set for gene prediction, all the relatively reliably predicted genes in 3 similar genomes (stored in the files listed in **Embl.List**) are used. This must give us a better result? To see the files you generated with this final gene prediction script, **type in**:

```
ls -lrt bmb.*
```

which should generate a list similar to:

```
bmb.train
bmb.icm
bmb.run1.predict
bmb.run1.detail
bmb.upstream
bmb.coords
bmb.motif
bmb.predict
bmb.detail
```

The final step for this stage is to reformat the **bmb.predict** file. The intention is not to change the information in the file, but to just rearrange its message in the way that the annotation software to be used next understands. Happily, there is a ready made script to do this and all you have to do is **type in**:

```
glimmer3totab.pl bmb.predict > bmb.tab
```

**glimmer3totab.pl** converts the **glimmer** predictions stored in the file **bmb.predict** into EMBL format<sup>9</sup> and stores them into (>) a file called **bmb.tab**<sup>10</sup>.

<sup>9</sup> The same format used for the annotated genomes of the 3 similar organisms.

<sup>10</sup> For consistency, the file should really be called **bmb.embl**, however, the software that will read this file will recognize it more easily if we use the extension **.tab**.



## Annotating the genome with Artemis:

Open the Desktop folder called **Artemis Tools**. Therein, you will discover a link labelled **Artemis**. **Artemis** is the program with which you will annotate your “new genome”. To start up **Artemis** with your genome loaded, drag and drop the file icon for **Cinara\_cedri.fasta** onto the **Artemis** icon. Have patience, **Artemis** may take a little while to rumble forth and display your genome sequence.

The top half of the **Artemis** sequence display (the *Sequence View Panel*) shows the positions of the stop codons in all 6 reading frames. In the next section down (the *Translations Panel*) you are shown the translations of all 6 reading frames<sup>11</sup>, “\*” and “#” indicating stop codons<sup>12</sup>.



It is already very clear to see the longer open reading frames (ORFs) many of which will be genes in such a simple organism. To annotate these ORFs as **CoDing Sequences (CDS)** go to the **Create** pull down Menu and select **Mark Open Reading Frames**. Accept the default definition that an ORF is interesting if it be **100** bases pairs or wider and all qualifying regions will be painted a tasteful pale blue with built in arrow to indicate the strand. Note that in the *Features Panel*, right at the bottom of the **Artemis** window, the ORFs are listed as **CDS** features. These would be saved as part of your annotation decisions as things stand.



Clearly, it is too crude to imagine that all the biggish ORFs are coding, especially after we have spent so much effort in “sophisticated” gene prediction with **glimmer**. To match our predictions with the ORFs calculated by **Artemis**, select **Read an entry** from the File pull down menu. Select **bmb.tab** from the proffered list and click on the **Open** button. The **glimmer** predictions will be displayed in positively dark blue in your **Artemis** window. You will see that most **glimmer** predictions are superimposed upon the hesitantly pale ORFs. Scroll down the **CDS** features in the *Features Panel*. See that your **glimmer** gene predictions have been added after the ORF entries.

Switch the display of the **glimmer** predictions and/or the ORF features off and on using the buttons at the top of the **Artemis** window. Examine particularly the **glimmer** prediction **orf00007**. By manipulating the displays, you will see it is not supported by an ORF feature. **glimmer** has predicted a gene that is not evident by just looking for ORFs longer than **100** base pairs. See also that there are many regions in which ORFs have been detected on both strands although it is rare to find DNA that codes simultaneously on both strands. **glimmer** is far more constrained in this respect.

11 The outermost lines in each panel represents the first reading frame of each DNA strand. The first reading frame being that for which the first base pair of the genome is in codon position 1. To be sure which frame is frame 1 of the reverse strand, you will need to look at the very end of the display to see the first base of the reverse strand. Look at the ends of the *TranslationPanel* for this purpose.

12 By default, + for TAG, # for TAA and \* for TGA.

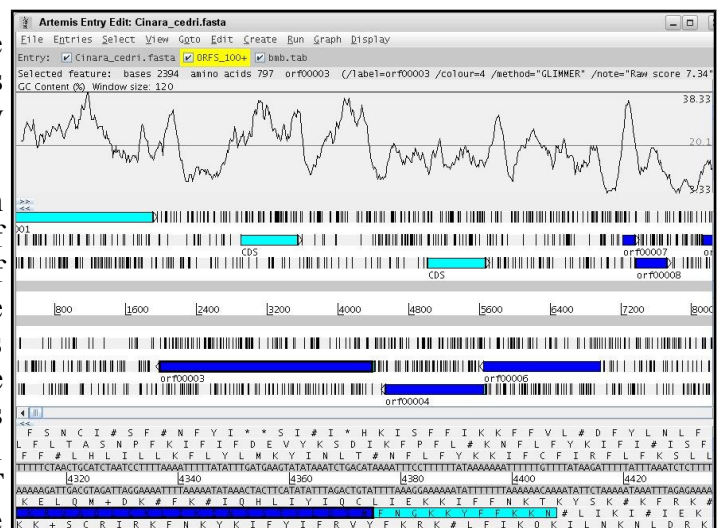
Note and try the scroll bars that allow you to meander whimsically back and forth sampling the delights of 6 lanes of stop laden reading frames in the *Sequence View Panel* and, independently, along their amino acid translations in the *Translations Panel*. Single click a feature with the left hand mouse button and it will become selected and its feature line will become highlighted. Double click a feature with the left hand mouse button and the *Translation Panel* will also align with the *Sequence View Panel*. Double click with your middle mouse button anywhere in either panel and you would see that the ORF (however small) you clicked on would become highlighted in various hues in both synchronised genome displays. If you selected an ORF feature or a **glimmer** prediction in this fashion, a window would appear boasting of all that is known of this feature (not a lot at this stage).

To discover a bit more about the genome features, an obvious move would be to compare all **glimmer** predicted genes (and possibly ORFs) with proteins from other bacteria. Good matches will suggest that the feature is indeed a gene. They may also provide a hint as to function. So, set some searches going<sup>13</sup>. First move back to the start of the genome and select (single click left mouse button) the glimmer prediction **orf00003**. Now go to the **Run** menu and select **Run fasta on selected features against** the database called **%uniprot\_bacteria**. Your search will be started up and run in the background. You will be alerted by the appearance of a new small window when your search is complete<sup>14</sup>. Set a similar **fasta** search going for the light blue ORF adjacent to **orf00003** but on the opposite strand. **glimmer** does not think this is a gene and it would be unusual to have a coding regions in the same place on both strands. It will be interesting to see what the evidence of the database search suggests.

Set a third and final **fasta** going for the small **glimmer** prediction **orf00007**. This is too small to register as an **Open Reading Frame** of 100 or more codons. It would be good to see if the database search supports the judgement of **glimmer**.

Your searches will take 1 or 2 minutes. While you are waiting, take a look at some of the **Graphs** representing the way some sequence properties vary over the length of the genome.

From the **Graph** menu, select **GC content**. An enlightening wiggly line will appear at the top of your **Artemis** display showing how the percentage of **Gs** and **Cs** varies over the length of the genome. The middle horizontal straight line of this plot indicates the average **G+C** content of the whole genome. The vertical range of the portion of the plot in view is shown at the right hand end of the graph. This graph is valuable for genomes that are either **GC** rich or **AT** rich. For such genomes, the base bias will be exaggerated in non-coding regions and lessen in coding regions<sup>15</sup>.



What is the average **G+C** content of the **Cinara cedri** genome? \_\_\_\_\_

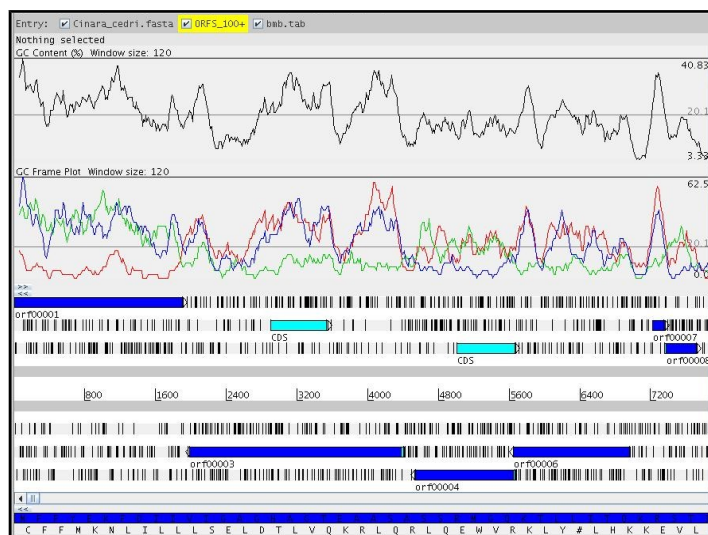
Would you judge the **GC content** graph useful for this genome? If so, why? \_\_\_\_\_

How would you interpret the **GC content** graph for the features that you gave to **fasta** to compare against other bacteria proteins? \_\_\_\_\_

13 In "Real Life", one might set searches going for all the regions of the genome that might be a gene. The searches are slow however, and this approach is therefore not possible in the context of this practical.

14 It will take a little time, so just carry on whilst you wait.

15 True as it is not possible to code for believable proteins with DNA that has either an extreme **AT** bias or an extreme **GC** bias. So, for example, in a very **GC** rich genome, the few **As** and **Ts** available must be concentrated in the coding regions to allow things to work.



Try also the **GC Frame Plot** option from the **Graph** pull down menu. 3 lines are generated. This graph is similar to the **GC content graph** but shows the **G+C** content of the first, second and third position in all codons simultaneously.

The green line shows the **G+C** content of positions 1,4,7,10 ... the blue line of positions 2,5,8,11 .... and the red line of positions 3,6,9,12 ...

So for a gene in reading frame 1, the green line shows the **G+C** in the first position of the codon (GC1), the blue shows the second codon position (GC2), and the red shows GC3. For a gene in frame 2, the blue shows GC1, the red GC2 and the green GC3, for a gene in frame 3 the red shows GC1, green GC2 and blue GC3. In summary:

Reading Frame 1			Reading Frame 2			Reading Frame 3		
Codon Position 1	Codon Position 2	Codon Position 3	Codon Position 1	Codon Position 2	Codon Position 3	Codon Position 1	Codon Position 2	Codon Position 3

For all organisms, in coding regions, GC2 tends to be higher than GC1, however, GC3 is dependant on the overall **G+C** of the genome; high **G+C** organism have very high GC3 and low **G+C** organisms have very low GC3. In genomes where there is no base bias, this plot is not very useful.

How do your expectations for this plot match the actuality for the regions of the top strand that you are analysing with **fasta**? \_\_\_\_\_

The **GC Frame Plot** can also be used to interpret the reverse strand of the genome as the proportions of **G+C** must be the same for either strand. With a bit of thought, and examination the end of the genome to see how the reading frames are arranged, you could compute how the colours of the plot relate to the codon positions of each reading frame of the reverse strand. To save you pain, I will do this for you. The colour scheme for the reverse strand can be summarised as follows:

Reading Frame 1			Reading Frame 2			Reading Frame 3		
Codon Position 1	Codon Position 2	Codon Position 3	Codon Position 1	Codon Position 2	Codon Position 3	Codon Position 1	Codon Position 2	Codon Position 3

How do your expectations for this plot match the actuality for the regions of the bottom strand that you are analysing with **fasta**? \_\_\_\_\_

Another way to judge where the genes might be is to use **codon usage**. This simple idea requires only that the observed **codon usage**<sup>16</sup> in each reading frame be compares to the “expected” **codon usage**. This requires a suitable model codon usage table which can be obtained in a number of ways, including computation from the previously annotated similar genomes or, in desperation, from the larger ORFs of the genome under investigation. A better approach would be to download a suitable precomputed model codon usage table from the comprehensive codon usage database in Japan. The tables stored here are carefully constructed from all available relevant sources. I have done this for you, as using this database is clumsy and absorbs more time than is justified in the context of this exercise. You will find the downloaded table in a file called:

**Buchnera.ctable**

<sup>16</sup> **Codon usage** is usually represented by **64** numbers showing the relative frequency of each of the **64** possible codons in a reading frame. The number can be percentages, or sometimes just raw counts.



I have also included the instructions (shaded as previously) for “doing it yourself” should you wish to.

To access this database, go to:

**<http://www.kazusa.or.jp/codon>**

and in the box labelled **QUERY Box** for search with **Latin name of organism** enter:

**Buchnera aphidicola**

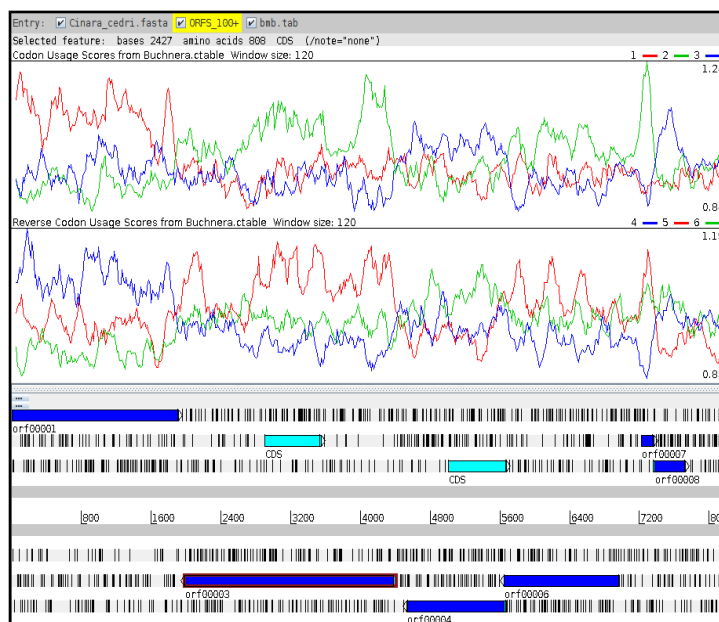
Click on the **Submit** button. You should see a list of links to codon usage tables for the 4 individual sub species of **Buchnera aphidicola** we are using, plus a link to a table for the species as a whole. The general table is the most appropriate as we would not want to cheat and use the table for the species we are pretending has yet to be fully analysed! So click on the **Buchnera aphidicola** link. A fine table, but curiously no easy way I can see to download? So, we have to copy and paste into a file.

UUU 38.5 ( 4544)	UCU 30.8 ( 3634)	UAU 27.8 ( 3287)	UGU 9.6 ( 1137)
UUC 2.9 ( 346)	UCC 3.0 ( 357)	UAC 4.0 ( 467)	UGC 2.4 ( 281)
UUA 72.7 ( 8589)	UCA 19.5 ( 2306)	UAA 2.7 ( 322)	UGA 0.5 ( 62)
UUG 9.7 ( 1146)	UCG 2.5 ( 293)	UAG 0.2 ( 18)	UGG 5.0 ( 592)
CUU 8.8 ( 1040)	CCU 13.2 ( 1559)	CAU 15.5 ( 1826)	CGU 13.4 ( 1586)
CUC 1.8 ( 210)	CCC 1.2 ( 138)	CAC 2.0 ( 232)	CGC 1.1 ( 133)
CUA 6.7 ( 795)	CCA 10.1 ( 1188)	CAA 28.9 ( 3408)	CGA 5.9 ( 697)
CUG 1.3 ( 156)	CCG 1.8 ( 213)	CAG 3.3 ( 385)	CGG 0.4 ( 51)
AUU 65.3 ( 7712)	ACU 20.7 ( 2448)	AUA 65.9 ( 7786)	AGU 13.6 ( 1608)
AUC 8.4 ( 994)	ACC 2.3 ( 273)	AAC 8.8 ( 1037)	AGC 2.1 ( 248)
AUA 39.6 ( 4675)	ACA 16.2 ( 1918)	AAA 92.7 ( 10946)	AGA 15.5 ( 1830)
AUG 26.3 ( 3107)	ACG 2.2 ( 257)	AAG 8.7 ( 1026)	AGG 1.0 ( 121)
GUU 23.1 ( 2723)	GCU 26.2 ( 3093)	GAU 38.7 ( 4575)	GGU 25.4 ( 2998)
GUC 1.9 ( 223)	GCC 2.4 ( 286)	GAC 4.3 ( 512)	GGC 3.1 ( 370)
GUA 22.8 ( 2694)	GCA 23.7 ( 2799)	GAA 55.6 ( 6588)	GGA 22.2 ( 2621)
GUG 3.9 ( 459)	GCG 2.6 ( 302)	GAG 4.1 ( 488)	GGG 3.4 ( 405)

To start a simple text editor, click once on the **gedit** icon on your **Desktop** ( middle of the panel on the left of the screen). **Copy** and **Paste** the region of your browser window shown in the illustration into the empty **gedit** window. **Save** your codon usage table as a file called:

**Buchnera.ctable**

in your **GENOME\_ANALYSIS** directory. **Quit gedit**.



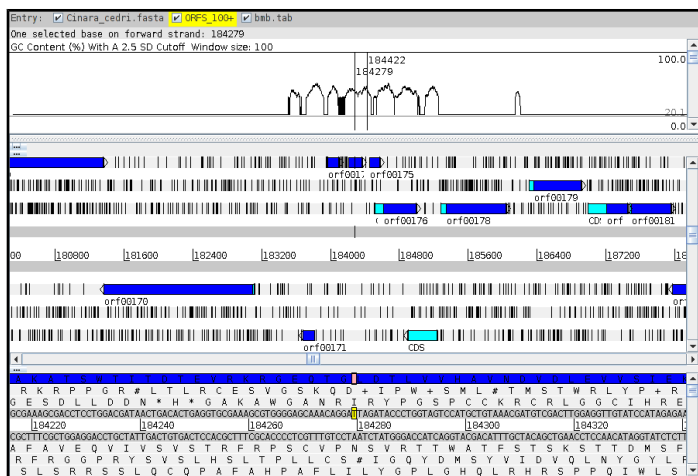
Back to Artemis. From the **Graph** menu select **Add Usage Plots** with the codon usage table in the file **Buchnera.ctable**. Note that the codon usage plots (6 coloured plots, 3 for the forward reading frames, 3 for the reverse) are of roughly the same shape and message as the **GC** analysis plots. **Tidy up** by turning off the **GC content** and **GC Frame Plots**.

How the colours relate to the reading frames is clearly indicated on the plots this time. There is a good prediction in both reading frames in most places. Even if you turned off the ORFs and the glimmer predictions, you should still be able to pick which frames are most likely to code by the absence of stop codons. Note particularly the really good support for **orf00007** which was too small to warrant an ORF feature. Check along the genome and see that the codon usage predictions consistently agree with the glimmer prediction. Change the proportion of the genome you have in view (the whole picture gets pretty jumbled if you view too much).

If you had only the evidence of this plot, which strand would you choose to be most likely to be coding around the region of the ORF **CDS 0001** (or glimmer prediction **orf00001**)?

Can you explain your decision?





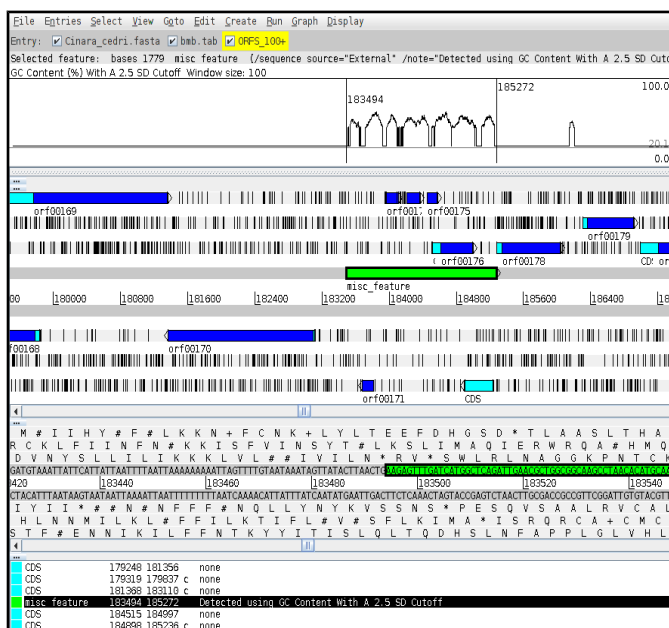
somewhere in the peak region to synchronise all the Artemis panels.

You have detected at least one region where the genome properties are very different to the norm. One explanation for these region(s) is that they might have been inserted from some external source(s). The lack of convincing gene predictions in these regions might support this theory.

Having found this interesting “feature“, it would be a good idea to record it. To do this click and drag from one end of the feature to the other in the plot window. A yellow region will appear in the genome display panels (associated with the forward strand, by default). From the **Create** menu, select the **Feature From Base Range** option.



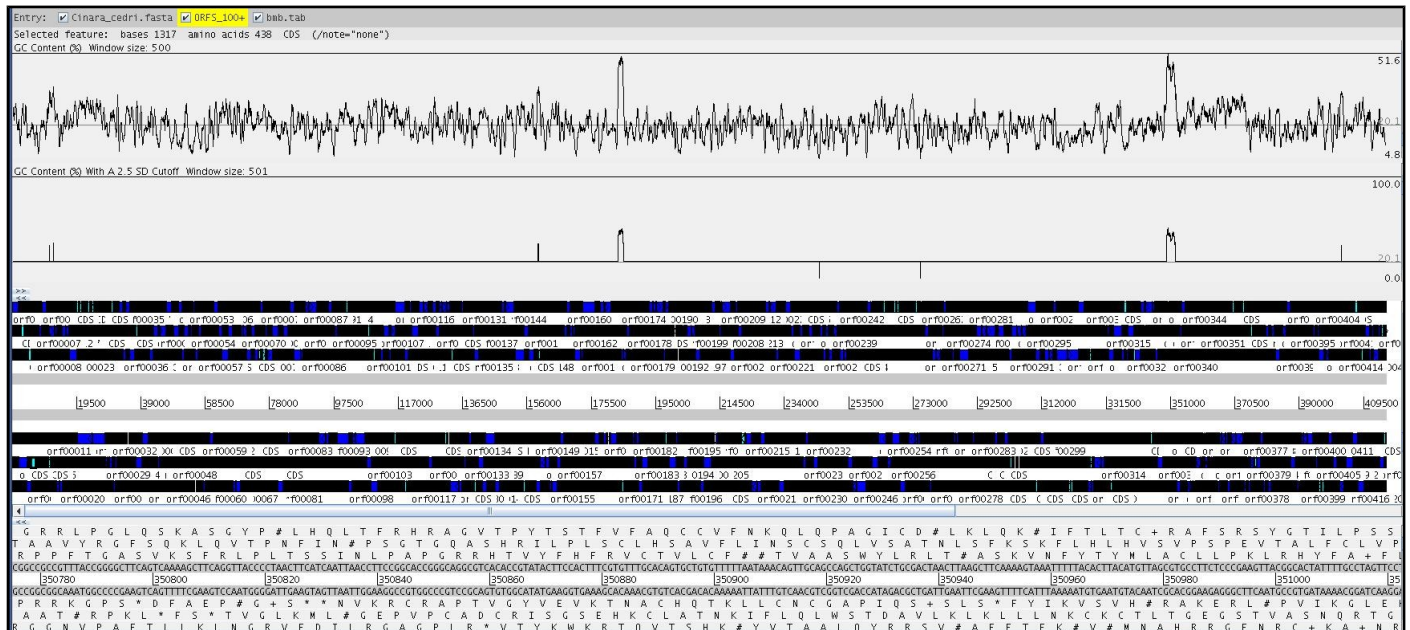
An **Artemis Feature Edit** window will appear representing the region you selected. Change the **Key** field from **CDS** to **misc\_feature**. Change the **Qualifier** to **sequence\_source** and click the **Add Qualifier** button. Speculate upon the **sequence\_source** as being **External**.



Add a **note** **Qualifier** to record the **Artemis** function that detected this feature. Add any other **Qualifiers** you see fit with the abandon afforded only to those removed from the harsh constraints of reality.

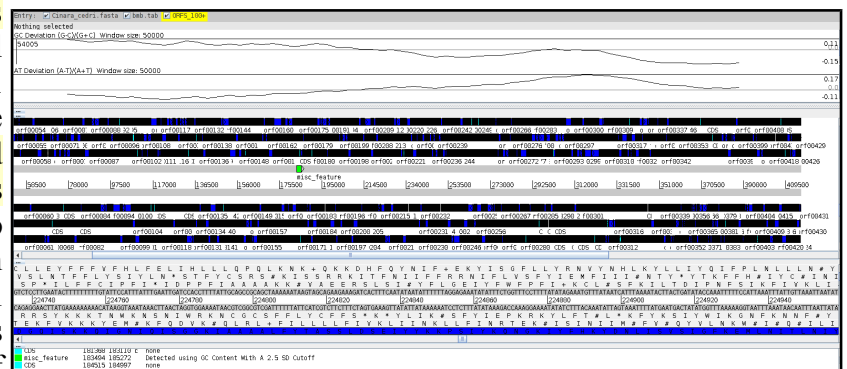
When your invention is sated, click with assertion upon the proffered **OK** button and you will be rewarded by a new feature of virulent green making its mark in all panels of your **Artemis** display.

Turn on the **GC content** plot once more, you should not be surprised to see that too picks out the feature you have just labelled. If you adjust your plots carefully to view the whole of your genome with both these plots “on”, you should agree that there are 2 regions worthy of note. Label the second region as you did the first.



Remove all the graphs you have made so far using the **Hide All Graphs** option in the **Graph** menu. Turn on the **GC Deviation (G-C)/(G+C)** and **AT Deviation (A-T)/(A+T)** graphs. These graphs plot the balance between Gs and Cs or As and Ts throughout the genome. For example, where the **GC Deviation (G-C)/(G+C)** plot meanders above the central horizontal line, there are more Gs than Cs in the forward strand of the genome, when below the middle level there are more Cs than Gs.

Make your Artemis window as wide as you can. Adjust the panel views so that all the genome is in view using the vertical scroll bars at the right hand side of the Artemis display. Now use your right hand mouse button in each of the graph windows to adjust the **Maximum Window Size** to **50000**. Finally set the window size as high as it will go (**50000** even) using the vertical scroll bar in each graph panel. The plots quite clearly indicate that the one half of the genome has more Gs and As than Cs and Ts. The other half has more Cs and As than Gs and Ts. This is true of bacteria genomes in general.



Can you explain why?

By this time, your **fasta** runs should have finished. As they terminated they will have thrust forth irritating undersized windows, dripping with boastful satisfaction and opining the completion of a job well done. Now is the time to make judgement upon the substance their complacent self assessment.

From the **Graph** menu, **Hide All Graphs** once more. Resize the sequence display back to something sane that allows you to clearly see the features at the start of the genome. Select the **glimmer** prediction **orf00003** (that you compared with the bacteria sequence database with **fasta**) with a single mouse click. From the **View** menu go to **Search Results** and then to **fasta results**. A new window should appear displaying your **fasta** results. Click on the **Send to Browser** button and your results will reappear in a browser window allowing you to easily retrieve the sequences matching **orf00003**.

Following a few opening remarks, **fasta** lists all the database sequences it found to match your query to a convincing degree. For each matching sequence, from left to right, **fasta** offers:

- A link to an alignment between query and database entry
- A one line description of the database entry
- The length of the database entry in brackets
- 3 evaluations of the degree to which the database entry matches the query sequence
- A link to the matching databases sequence itself

The third score, in the column approximately under the **E** heading, is the Expect score. It suggests the number of times that a "hit" of at least the quality of that to which it is associated, is likely to occur by chance. Hopefully you agree therefore that an **E** score of **0**, or only a tiny bit more than **0** is jolly good?

Based on their **Expect** scores, how convincing are the first few hits for this search?

Link to the first few alignments. Do they confirm your evaluation hit quality?

Would you conclude that **glimmer's** prediction for **orf00003** is accurate?

If **orf00003** is a gene, does the list give any clues as to what sort of gene?

Based upon your **fasta** results (and the evidence of previous analyses), if you think that **orf00003** is a real gene, then ensure the feature is selected, **Edit the Selected Features in Editor** to **Add Qualifier** of type **note** to express your deep satisfaction and possibly record what sort of gene **orf00003** might be. Otherwise .... think again!

Now, clear away your **fasta** results windows for **orf00003** and, in exactly the same fashion, investigate the search results for the CDS feature in the same region as **orf00003** but on the opposite strand.

Based on their **Expect** scores, how convincing are the first few hits for this search?

Link to the first few alignments. Do they confirm your evaluation hit quality?

What do you conclude about the likelihood of a gene in this ORF?

Based upon your **fasta** results (and the evidence of previous analyses), if you think that the CDS opposite **orf00003** is not a real gene, then remove its feature using your right hand mouse button on the feature and selecting **Edit**, then **Selected Feature(s)** and then **Delete**. After you confirm your terminal judgement, the CDS feature will retreat forever. If, on the other hand, you think this CDS is a real gene ... well ... you are

108 506 338:\* :==\*  
110 399 262:\* :==\*  
112 331 203:\* :==\*  
114 241 157:\* :==\*  
116 199 121:\* :==\*  
118 157 94:\* :==\*  
120 153 73:\* :==\*  
122 134 56:\* :==\*  
124 98 43:\* :==\*  
126 88 34:\* :==\*  
128 66 26:\* :==\*  
>130 7587 20:\*\*\*\*\* :=====

Statistics: Expectation n fit: rho(ln(x)) = 7.41404/-0.00018; mu = 5.6243/-0.009  
mean\_var=76.1818/-14.973; 0's: 15 Z-trim(117,0): 336 B-trim: 806 in 1/55  
Lambda= 0.146943  
Statistics sampled from 60000 (81714) to 635294 sequences  
Kolmogorov-Smirnov statistic: 0.0381 (N=24) at 56  
Algorithm: FASTA (3.7 Nov 2010) [optimized]  
Parameters: BL50 matrix (15:-5)x5, open/ext: -10/-2  
ktp: 3, E-join: 1 (0.482), E-opt: 0.2 (0.159), width: 16  
Scan time: 54.150

The best scores are:

Query	Database	Score	Expect
Q586G1_BUCC005861	Full-DNA gyrase subun	( 797)	5122 1096.9 0
GYRB_BUCCP_P29425	Full-DNA gyrase subunit	( 803)	3190 687.3 3.2e-195
GYRB_BUCCP_Q89637	Full-DNA gyrase subunit	( 804)	3151 679.1 9.9e-193
GYRB_BUCA1_P57126	Full-DNA gyrase subunit	( 803)	3079 663.8 3.9e-188
B5XT54_KLEPN_B5XT54	Full-DNA gyrase, B su	( 804)	3041 655.7 1e-185
Q7NA00_PHEL1_Q7NA00	Full-DNA gyrase subun	( 804)	3037 654.9 1.9e-185
A6TG01_KLEP7_A6TG01	Full-DNA gyrase subun	( 804)	3031 653.6 4.5e-185
B5EYB1_SALAA4_B5EYB1	Full-DNA gyrase, B su	( 804)	3028 653.0 7e-185
ABAC11_CITK9_ABAC11	Full-Putative unchara	( 804)	3025 652.3 1.1e-184
B5NDV7_SALET_B5NDV7	Full-DNA gyrase, B su	( 804)	3021 651.5 2e-184
AGMMW7_SALPB_AGMW7	Full-Putative unchara	( 804)	3021 651.5 2e-184
B5NDP9_SALET_B5NDP9	Full-DNA gyrase, B su	( 804)	3021 651.5 2e-184
GYRB_SALT1_POA213	Full-DNA gyrase subunit	( 804)	3021 651.5 2e-184
B5CA74_SALET_B5CA74	Full-DNA gyrase, B su	( 804)	3021 651.5 2e-184
B5MNP9_SALET_B5MNP9	Full-DNA gyrase, B su	( 804)	3021 651.5 2e-184
B5RFZ0_SALG2_B5RFZ0	Full-DNA gyrase subun	( 804)	3021 651.5 2e-184
B5P1A0_SALET_B5P1A0	Full-DNA gyrase, B su	( 804)	3021 651.5 2e-184
GYRB_SALT1_POA214	Full-DNA gyrase subunit	( 804)	3021 651.5 2e-184
B5C5A3_SALET_B5C5A3	Full-DNA gyrase, B su	( 804)	3021 651.5 2e-184
B5MGT0_SALET_B5MGT0	Full-DNA gyrase, B su	( 804)	3021 651.5 2e-184
B4A4U1_SALNE_B4A4U1	Full-DNA gyrase, B su	( 804)	3021 651.5 2e-184
B4TN04_SALSV_B4TN04	Full-DNA gyrase, B su	( 804)	3021 651.5 2e-184
B5Q666_SALV1_B5Q666	Full-DNA gyrase, B su	( 804)	3021 651.5 2e-184
B4SYA5_SALNS_B4SYA5	Full-DNA gyrase, B su	( 804)	3021 651.5 2e-184
B5FN07_SALDC_B5FN07	Full-DNA gyrase, B su	( 804)	3021 651.5 2e-184
B5QUP7_SALEP_B5QUP7	Full-DNA gyrase subun	( 804)	3021 651.5 2e-184
B5YENB_SALET_B5YENB	Full-DNA gyrase, B su	( 804)	3021 651.5 2e-184
B5PQV2_SALHA_B5PQV2	Full-DNA gyrase, B su	( 804)	3021 651.5 2e-184
A6GVF0_SALET_A6GVF0	Full-DNA gyrase subun	( 804)	3021 651.5 2e-184
B5PBH3_SALET_B5PBH3	Full-DNA gyrase, B su	( 804)	3021 651.5 2e-184
B4TAU6_SALHS_B4TAU6	Full-DNA gyrase, B su	( 804)	3021 651.5 2e-184
B5E0L1_SALPK_B5E0L1	Full-DNA gyrase subun	( 804)	3020 651.3 2.3e-184
Q5PKU9_SALPA_Q5PKU9	Full-DNA gyrase subun	( 804)	3020 651.3 2.3e-184
AGM3U3_SALAR_AGM3U3	Full-Putative unchara	( 804)	3018 650.9 3e-184
B3BDH0_EC057_B3BDH0	Full-DNA gyrase, B su	( 804)	3017 650.6 3.5e-184
B3A2E3_EC057_B3A2E3	Full-DNA gyrase, B su	( 804)	3017 650.6 3.5e-184
B3T5L0_EC0LX_B3T5L0	Full-DNA gyrase, B su	( 804)	3017 650.6 3.5e-184
A7ZTQ5_EC024_A7ZTQ5	Full-DNA gyrase, B su	( 804)	3017 650.6 3.5e-184
ABAG60_EC0H5_ABAG60	Full-DNA gyrase, B su	( 804)	3017 650.6 3.5e-184
Q2NV45_S00GH_Q2NV45	Full-DNA gyrase subun	( 804)	3017 650.6 3.5e-184


Close Send to browser

**Wrong!!!** Apart from any other consideration, it is not impossible for DNA to code on both strands in one place, but it is very unusual.

One more to look at. Clear up again and take a look at the search results for **orf00007**. This time, from the browser version of your search results, follow at least one of the links to the proteins that best match. You are retrieving the sequences from the European Bioinformatics Institute databases in Hinxton. From these sequence pages you could investigate many other sources holding information about the protein represented by **orf00007** (yes, this one has to be real surely!) and the family of proteins to which it belongs. All sadly beyond the scope of what we have time to tackle today.

Finally for **Artemis**, save all relevant features plus the entire genome sequence into a suitably formatted file for further investigation. If you were seriously annotating this entire genome, you would have checked and annotated every promising feature prediction properly and discarded all those that were improbable. As this is actually not so, you need to take a very crude short-cut to make something that will work ... approximately.

Note that the promising ORF features were generally supported by the **glimmer** predictions. The simplest horrible short-cut would be to throw all the ORF predictions away and rely just on the **glimmer** predictions. To do this simply untick the **ORFS\_100+** Entry and all the corresponding entries will remove themselves from the Features list. Assume all the remaining listed features are “good”. From the **Select** menu choose **All**. In the Features panel the entire list is now highlighted indicating that everything is selected. Now add all features to the original sequence which is stored in the Entry **Cinara\_cedri.fasta**. This is achieved with the **Copy Selected Features To** option of the **Edit** menu. Finally, from the **File** menu select the option **Save An Entry As** and then the sub-option **EMBL Format** applied to the Entry **Cinara\_cedri.fasta**. When asked for a file name, answer **Cinara\_cedri.embl**.

Take a look at the file you have just created in your graphical file manager. You should see all the features you saved plus the whole genome sequence in EMBL format. Wonderful, but there is unfortunately a vital line missing at the top of the file. Drag and drop the icon for the file **Cinara\_cedri.embl** on to the **Text Editor** application icon ( middle of the panel on the left of the screen). Add to the very top of the file a line:

**ID CINARA**

Ensure there are exactly 3 spaces between ID and CINARA. From the **File** menu of the **Text Editor**, select **Save** and then **Quit**. Your task is complete<sup>17</sup>.

That is it for **Artemis**. Iconify<sup>18</sup> all surviving **Artemis** windows rather than close them down. Just in case you need to return for any reason.

<sup>17</sup> Try starting up **DNA\_Plotter** and loading **Cinara\_cedri.embl** as a sequence file. It is very pretty, if nothing else.

<sup>18</sup> Is that *really* a word?



## Comparing genomes with ACT:

We will now investigate the similarities between our newly, if very roughly, annotated genome with one of the closely similar bacteria genomes downloaded from the **EBI**. To do this we need EMBL formatted sequence files, with annotation, for the two genomes to be compared. We also need a further file in which a comparison between the two genomes is stored. We have the genome sequence files, but we need to compute the comparisons to store in the third file remotely.

It is possible for you to do this now. However, the results you will generate will not be optimal (my fault, I have not discovered the correct parameters to use as yet). It is also time consuming, clumsy and error prone. I include the instructions (shaded as previously) to generate reasonable results for a comparison between **Cinara\_cedri** and **Baizongia\_pistaciae**, but I suggest you cheat and use the precomputed comparison file (downloaded from the same site) which I have put in place for you. It is called, quite improbably:

**genome\_blast\_Cinara\_cedri-V-Baizongia\_pistaciae.result**

In your web browser, go to<sup>19</sup>:

**<http://www.webact.org/>**

Click on the **Generate** tab. Elect to **Upload File (raw, EMBL or FASTA format)** in both places offered, using the **Browse** button for two of your genome EMBL format sequence files. Sensibly, one should be:

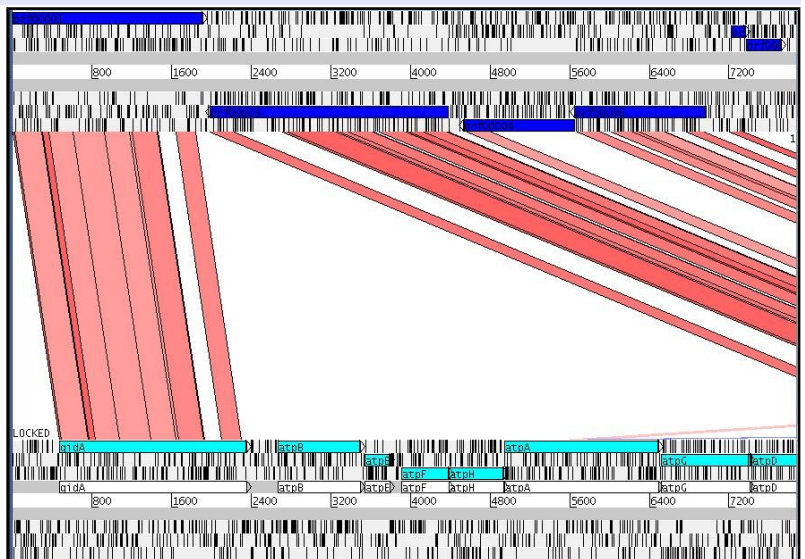
**Cinara\_cedri.embl**

The other could be any of the 3 similar genome files that you started with. If you want your results to exactly match my meanderings, use<sup>20</sup>:

**Baizongia\_pistaciae.embl**

Click on the **Blast Search Options [Show]** button and select **TBlastX** rather than **BlastN**. **BlastN** (the default) compares the DNA of the 2 genomes, **TBlastX** compares the protein translations and generally gives a clearer view of what is going on. Click on the **Submit** button to start the genome comparison rolling. The comparison should not take long ...he typed hopefully. When cooked, note that you could run **ACT** from the web site. However, using the version installed on your workstation has to be the better option, so click on **Download files** to transfer you comparison results to you local disc. The default name **WebACT.zip** is fine. Elect to **Open it with** the suggested application **ark**. From the **Action** menu of the program **ark**, choose **Extract**. Set the **Destination folder** to **GENOME\_ANALYSIS**<sup>21</sup> and click **OK to Extract all files**. Quit the program **Ark**.

Now start up **ACT** by clicking twice on its icon. Load your files by selecting **Open** from the **File** menu. Browse for **Sequence file 1 (Cinara\_cedri.embl)** and **Sequence file 2 (Baizongia\_pistaciae.embl)**<sup>22</sup>. Then **Browse for Comparison file 1**, the file you want is the one with the very long name quoted above (or **blast\_out1** if you generated yourself at **webact**). Once all three files are specified, click **Apply**. With all the abandon you can muster, discard all warning and error messages, throw to one side any log and file manager windows and focus only on the window with the pretty red lines.



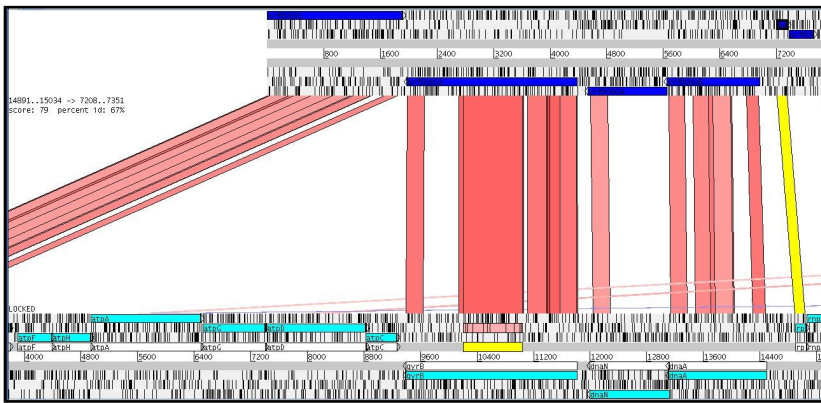
The top sequence display will be **Cinara cedri**, the lower sequence **Baizongia pistaciae**. The red lines joining the 2 indicate regions where the 2 genomes are significantly similar.

<sup>19</sup> This page should be directly available as **WebACT - Home** from the top of your browser window.

<sup>20</sup> Which does give the most interesting results by far.

<sup>21</sup> Should be just a matter of deleting **WebACT-1** from the end of the suggested destination.

<sup>22</sup> Or dragging and dropping the file icons works too.

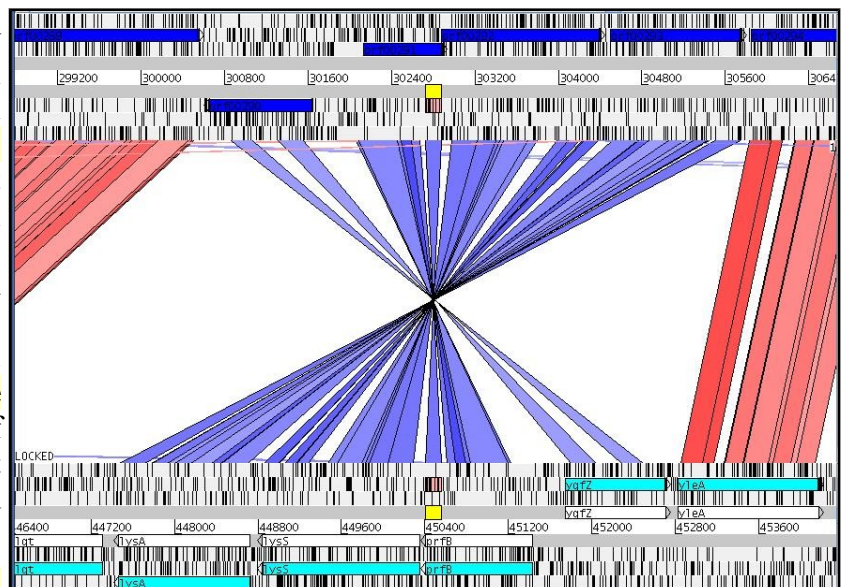
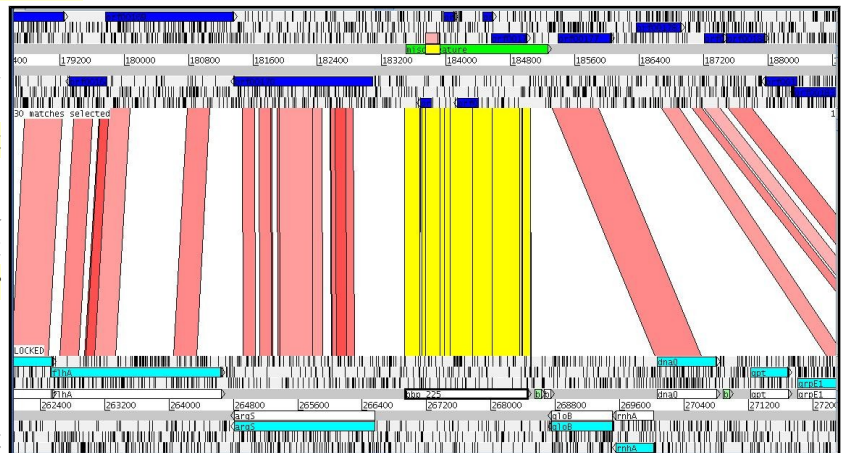


Double click on one of the red lines emanating from the region of **orf00003** in **Cinara cedri**. ACT colours the selected line yellow and adjusts its display to be centred around the regions you selected. Note that the gene name of the matching feature in **Baizongia pistaciae** is consistent with that suggested by your **fasta** searches. Click once on the tiny **orf0007 glimmer** prediction and see that it too matches nicely a gene in **Baizongia pistaciae**, consistent with that suggested by **fasta**. Note the

region in **Baizongia pistaciae** that is totally absent in **Cinara cedri**. See that the gene names in this region are all similar. You can check up what is known of the genes by examining their features (right click the feature, select **View** and then **Selected Features**), maybe the omission of this cluster indicates functional differences between the two bacteria?

Now take a look at the first region you labelled as a **misc\_feature**. To do this, from the **Goto** menu select **Cinara cedri.embl** and then **Navigator**. Elect to **Goto Feature with this Key:** and type **misc\_feature** into the appropriate test box. Click **Goto** and **Close** your **Navigator** window.

The **misc\_feature**, if you remember, was noteworthy because of its unusual GC composition. We speculated that maybe it was an insertion from some foreign source. However, it appears to be very closely matched with a ribosomal RNA in **Baizongia pistaciae.embl** (you will probably need to double click on the lines that have turned yellow to align your plot before this becomes apparent, the rRNA feature is white, view the feature details as you did previously). Maybe this region is a ribosomal RNA in **Cinara cedri** also, which would be consistent with its odd **G+C** properties. Edit the **misc\_feature** you created to reflect this new insight. If you feel particularly energetic, you might take a look at the other **misc\_feature** you created.

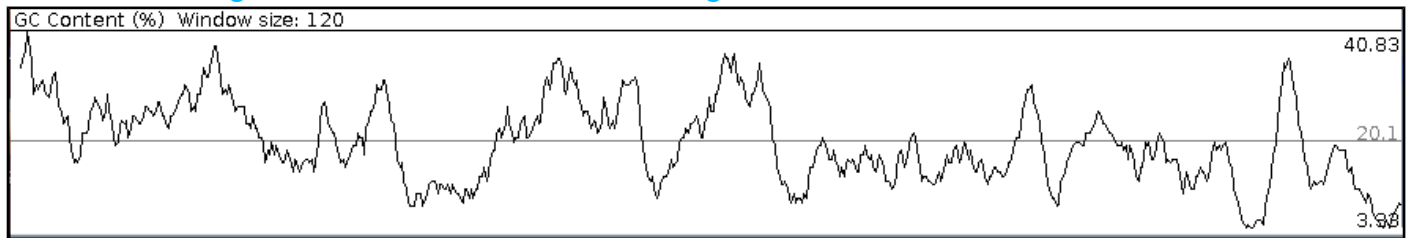


Finally, for this short tour, move along to around base 304800 and double click on the blue lines. Here you see a region that is similar between the 2 bacteria, but has been turned around at some stage. The genes are very similar but are on opposite strands.

These are very powerful tools in every day use at the Sanger Institute (where they are developed) and many other places. They have been used already to annotate many many genomes. We have just scratched the surface of that which they are capable.

## Model Answers

What is the average **G+C** content of the **Cinara cedri** genome?



The numbers on the extreme right of the plot indicate the maximum (**40.83%**), Average (**20.1%**) and minimum (**3.93%**) **G+C** content of the **Cinara cedri** genome. These values relate to all possible sections (**Windows**) of **120** contiguous bases throughout the genome.

So, to spell it out with maximal pedantry, it can be said that:

In this genome there is at least one unbroken stretch of **120** base pairs that is comprised of **40.83%** **Gs** and **Cs**. No unbroken stretch of **120** base pairs in this genome contains more than **40.83%** **Gs** and **Cs**.

In this genome there is at least one unbroken stretch of **120** base pairs that includes as few as **3.93%** **Gs** and **Cs**. No unbroken stretch of **120** base pairs in this genome contains less than **3.93%** **Gs** and **Cs**.

The average **G+C** content of all possible unbroken stretches of **120** base pairs in this genome is **20.1%**

So one might be justified in concluding that this genome has a very significant computational bias towards **As** and **Ts**.

Would you judge the **GC content graph** useful for this genome? If so, why?

The **GC content graph** can help to locate regions likely to be coding for protein wherever there is a significant compositional bias towards either **As/Ts** or **Gs/Cs**. There is a distinct compositional bias towards **As** and **Ts** in the **Cinara cedri** genome, so the analysis should be useful for this genome.

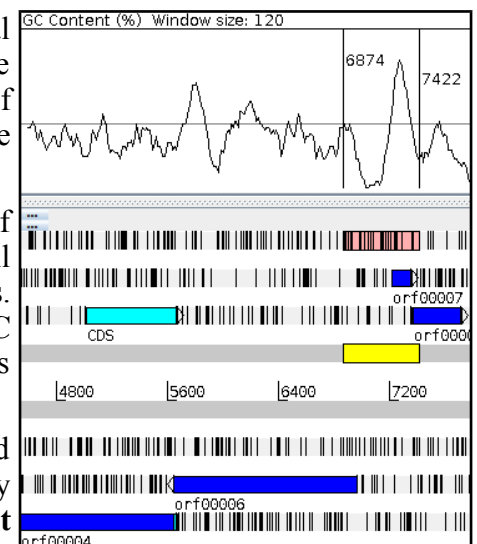
Clearly, a region of DNA that codes for protein must have a sufficiency of all four bases. The redundancy of the **Genetic Code** provides some flexibility by allowing the use codons employing abundant bases preferentially, but to a limited extent only. It is therefore reasonable to assume that evolutionary pressure should result in a reduction of computational bias in protein coding regions.

The **GC content graph** visualizes the way compositional bias varies over the length of a genome. The straight middle line represents the average **GC content** for the entire genome (approximately **20%** in this case). The wiggly line shows local variation in **GC content**, when above the middle line there are more than average **Gs** and **Cs** (reduced compositional biased in this case), when below there are less (enhanced compositional bias in this case).

The **Cinara cedri** genome is **AT** rich, areas of reduced compositional biased (more likely to be protein coding) will be suggested when the wiggly line is above the average. For a **GC** rich genome, areas of reduced compositional biased (more likely to be protein coding) will be suggested when the wiggly line is below the average.

Clearly, measuring **GC content** is independent of strand. The ratio of **G+C** to **A+T** must be identical for both strands as **Cs** in one strand will be matched by **Gs** in the other, **Gs** by **Cs**, **As** by **Ts** and **Ts** by **As**. Consequently, any indication of protein coding provided by the **GC content graph** is not strand specific. This plot can only suggest regions likely to be coding in one strand or the other (and **very rarely ... both**).

The potential of this plot is reduced when used for bacteria (and similar) genomes. These genomes, generally, are compositionally biased, but they code for protein almost everywhere! The **GC content**





**graph** visualizes regions of relatively “unusual” base composition. Regions where the compositional properties are distinct from the background expectation ... that is unusual. Protein coding regions will not be easily identified where they are commonplace and so have too large a contribution to the definition of “background”. Clearly, deviation from background **GC content** will become more obvious the more rarely it occurs. Where most of a genome codes for protein, the method will not be as effective as for a genome where protein coding regions are relatively rare. This said, it does help modestly in this instance.

.....

How would you interpret the **GC content** graph for the features that you gave to **fasta** to compare against other bacteria proteins?

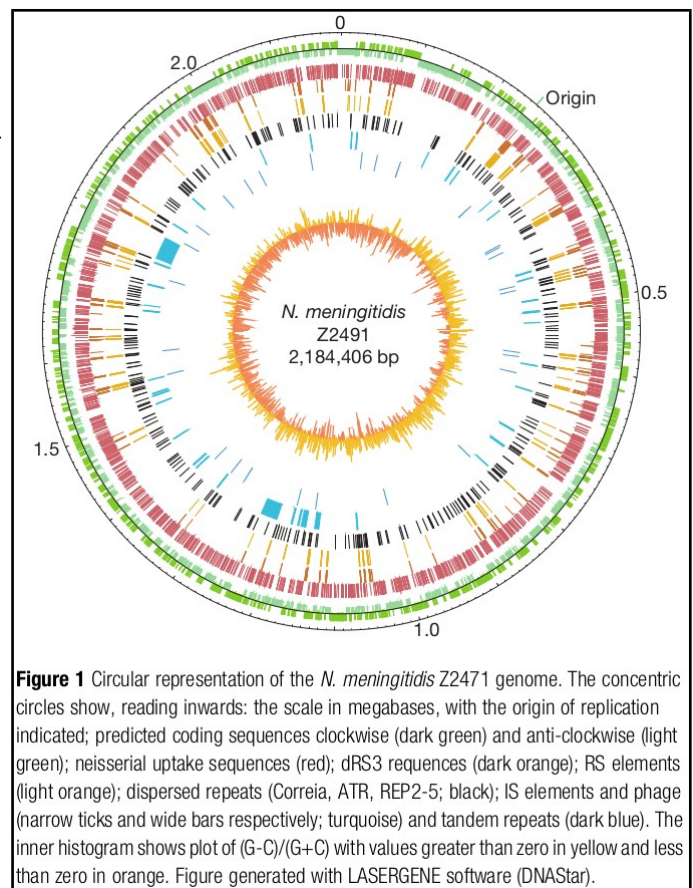
How do your expectations for this plot match the actuality for the regions of the top strand that you are analysing with **fasta**?

How do your expectations for this plot match the actuality for the regions of the bottom strand that you are analysing with **fasta**?

One half of the genome has more **Gs** and **As** than **Cs** and **Ts**. The other half has more **Cs** and **As** than **Gs** and **Ts**. This is true of bacteria genomes in general. Can you explain why?

If you can, you are doing better than I.

<http://www.smorfland.uni.wroc.pl/www/bacasy.html>





Evaluation of **fasta** hits for **orf00003**:

Based on their **Expect** scores, how convincing are the first few hits for this search?

Link to the first few alignments. Do they confirm your evaluation hit quality?

Would you conclude that **glimmer**'s prediction for **orf00003** is accurate?

If **orf00003** is a gene, does the list give any clues as to what sort of gene?

Evaluation of **fasta** hits for the CDS opposite **orf00003**:

Based on their **Expect** scores, how convincing are the first few hits for this search?

Link to the first few alignments. Do they confirm your evaluation hit quality?

What do you conclude about the likelihood of a gene in this ORF?