Hairpin: a tool for predicting structural non-coding RNA genes \textit{ad initio}

Jakob Peder Pettersen

Abstract

\textbf{Background:} Structural RNA genes play important and various roles in gene expression and its regulation. Finding such RNA genes in a genome poses a challenge, which in most cases is solved by homology approaches. \textit{Ab initio} methods for prediction exist, but are not that much explored.

\textbf{Results:} We introduce hairpin which identify potential structural RNA genes only based on the sequence. We use the algorithm to predict RNA genes in \textit{Escherichia coli} K-12. When looking at very short regions of the genome, we do not get results differing very much from a random shuffling of the genome. However, at longer stretches it is a clear biological signal. It turns out that none of the regions predicted to code for RNA genes have such an annotation in literature.

\textbf{Conclusions:} Arbitrary DNA sequences seem to give rise to transcripts with secondary structures similar to real ncRNA. We therefore conclude that exclusively looking at secondary structure base-parings is in general a futile approach.

\textbf{Keywords:} hairpin; ncRNA genes; gene predection; structural RNA; stem loops

Background

mRNA, the most known form of RNA, codes for proteins. The rest of the RNA is called non-coding. Nevertheless, non-coding RNA (ncRNA) plays important roles in gene expression and regulation thereof. The most known examples are tRNA and rRNA, both participating in proteins synthesis. Other examples include riboswitches in bacteria and miRNA in eukaryotes, which are examples of regulatory RNA. A region in the genome coding for ncRNA is called a RNA gene. Due to the their importance and various functions, finding RNA genes and their functions might help us to get a better understanding of the organism in question. Unlike protein-coding genes, RNA genes cannot be found by searching for open reading frames, making them harder to predict \cite{1, 2}.

Many ncRNA molecules have structures and motifs required for their function. In this respect, we mainly focus on the secondary RNA structure, this is the two-dimensional structure formed by base-pairings between nucleotides in the same molecule. A RNA gene where the secondary structure is related to its function, is called structured \textit{sensu} Meyer \cite{2}. Three different base-pairings occur in RNA molecules. In addition to the Watson-Crick pairs \{A,U\} and \{G,C\}, we also have the \{G,U\} base pair \cite{2}. One of the most common motifs being observed in structural RNA molecules is the stem-loop, where a stem consisting of two anti-parallel
parts of the molecule base-pair, where the two parts of the stem is separated by a loop comprising a few nucleotides [3].

Most approaches for RNA gene prediction are based on homology [2] where similarities in sequence or structure to known ncRNAs are used. Sometimes however, we may want to discover new RNA genes where information from homology is lacking. Such ab initio methods are rarer and considered more challenging [2]. There exist some specific methods based on nucleotide composition in AT-rich hyperthermophiles [4] and methods dedicated to find specific kinds of RNA genes [5]. A more general prediction method introduced by [6] uses the fact that base pairings lower the minimum free energy involved in RNA folding. The current article’s idea of using hairpins is not new either, [7] discusses this idea, but the level of detail is insufficient to get a clear overview of the procedure.

Implementation

We introduce hairpin, a command-line application for Python 3.5.0 or later, using Biopython [8] and NumPy [9]. The main idea of hairpin is to identify potential hairpins in the input and group them together. First, the entire nucleotide sequence is transcribed into a continuous RNA strand, using the input as the coding strand. The main idea is that structural RNA genes contain more and longer hairpins than expected by chance, distinguishing them from other parts of the genome. The algorithm thus has to consider the following:

• How do we find the hairpins?
• What do we mean by “random chance”?
• Once the hairpins are found, how do we use them to predict which parts of the genome contain RNA genes?

Different from [10], we do not seek to filter out RNA being regulatory parts in protein coding genes, such as attenuation sequences.

Note that the pseudocode written in this article does not directly reflect the actual implementation as it contains many optimizations which would be difficult to understand. Instead, the pseudocode aims to produce the same results.

Finding the hairpins

When searching for hairpins, we look for base-pairings involving consecutive residues. Taking figure 1 as an example; we observe that residues 1-5 (from the lower left) base-pair with residues 11-14, forming a stem. We note the following: Residue 1 can base-pair with residue 14, being 13 nucleotides apart, residue 2 can base-pair with residue 12, being 11 nucleotides apart, residue 3 can base-pair with residue 11, being 9 nucleotides apart and so on. hairpin utilizes this observation such that consecutive nucleotides having potential base-paring partners at a decreasing distance (decrease with two for each nucleotide), is considered a motif. We hence include the parameter \( l_{\text{min}} \) telling the minimum length of such a motif in order to report it. Setting \( l_{\text{min}} \) to a higher value makes it more difficult for motifs to arise by pure chance.

The pseudocode for this procedure is shown in algorithm 1.
Figure 1: Sample RNA stem-loop structure, generated with forna [11]

Estimating the structuredness of a RNA region
Telling whether a region of the RNA transcript is likely to represent structural RNA, requires some way to quantify the “structuredness” of the region. This will be based on the motifs occurring in the region. Instead of using a sophisticated algorithm to predict how the strand will fold as done by [6], we naively look at the number of nucleotides in the region taking part in one or more motifs, hereafter called the base-pairing count. We do not know whether there exists that many nucleotides being able to base-pair in the real RNA structure as the motifs may overlap, but the same nucleotide is never counted twice. However, this approach is meant as a heuristic being easy to compute. The pseudocode is listed in algorithm 2.

Computation of randomizations
Randomized RNA segments are built by sampling nucleotides with replacement from the original transcript. We designate parameters $r_{min}$ and $r_{max}$ being respectively the minimum and maximum length of the RNA regions to probe. For each length $r$ in the closed interval $[r_{min}, r_{max}]$, we take the $r$ first nucleotides and find the hairpins and base-pairing count. The procedure is repeated $n_{rand}$ times with newly generated RNA segments. Finally, for each length $r$, the mean base-pairing count and its standard deviation is computed. See algorithm 3 for pseudocode.

Comparing actual motifs found to the randomized results
We screen through the motifs for finding regions of appropriate size. If we find a such a region, we report the base-pairing count. This count is compared to what to expect by random using the assumption that this quantity is normally distributed under the null hypothesis. From this, a $p$-value is computed. This $p$-value is corrected by
the length of the original sequence by the following logic:

\[ p_{\text{val corrected}} = P(A_{\text{rand}} \geq 1) = 1 - P(c_{\text{rand}} < c_{\text{obs}})^{n_{\text{interval}}} \approx 1 - (1 - p_{\text{val}}^{\text{r_{genome}}})^{\frac{1}{r}} \]

where:

- \( A_{\text{rand}} \) is the number of regions of length \( r \) in the genome having an equal or higher base-paring count than \( c_{\text{obs}} \) under the null hypothesis
- \( c_{\text{rand}} \) is the base-paring count in an interval of length \( r \) under the null hypothesis
- \( c_{\text{obs}} \) is the observed base-paring count
- \( n_{\text{interval}} \) is the number of disjoint regions in the genome of length \( r \)
- \( r \) is the length of the region
- \( r_{\text{genome}} \) is the length of the genome

Thereafter, we go through the intervals, pick the most significant ones and discard the intervals overlapping with it. Furthermore, we account for multiple testing by correcting with the Benjamini–Yekutieli procedure [12], computing \( q \)-values. We keep all regions with \( q \)-value less than \( q_{\text{crit}} \). The procedure is listed in algorithm 4.

The results are finally reported as:

- A .bed track file to be uploaded in a genome browser. It includes the start and end position of significant regions being found, in addition to their \( q \)-values.
- A text file displaying the results for the randomized results with the mean base-pairing count for each length of the regions, together with its standard deviation.
- A plot showing the means and standard deviation of the base-pairing counts of the randomized results together with the lengths and base-pairing counts of the significant regions found.

**Results**

Using the parameters \( d_{\text{min}} = 4, d_{\text{max}} = 25, q_{\text{crit}} = 0.01, l_{\text{min}} = 8 \) and \( n_{\text{rand}} = 2000 \), the algorithm was run on the genome of *Eschericia coli* K12 [13], with four variations:

- \( r_{\text{min}} = 25 \) and \( r_{\text{max}} = 200 \) on the original genome
- \( r_{\text{min}} = 25 \) and \( r_{\text{max}} = 200 \) on a random reshuffling of the genome
- \( r_{\text{min}} = 200 \) and \( r_{\text{max}} = 1000 \) on the original genome
- \( r_{\text{min}} = 200 \) and \( r_{\text{max}} = 1000 \) on a random reshuffling of the genome

Running all the jobs in parallel on a machine with 48 2.4 GHz cores took 8501 seconds. Plots displaying the results are shown in figure 2.

The number of significant regions found is shown in table 1. For \( r_{\text{min}} = 200, r_{\text{max}} = 1000 \) on the original genome, the genome track file was uploaded to the UCSC Microbial Genome Browser [14] and the most significant regions were looked up manually. The results thereof are shown in table 2. Furthermore, the RNA corresponding to the first region listed in table 2 was visualized with *forna* [11] being part of the ViennaRNA Web Services, along with a transcript being of the same length being built from a reshuffling of the nucleotides of the genome. This is shown in figure 3.
Figure 2: Each significant region found is marked with a red dot, telling its length and its base-pairing count. The results for the randomized sequences are shown in blue where the blue line is the mean base-pairing count and the error bars correspond to the standard error. For $r_{\text{min}} = 200$ and $r_{\text{max}} = 1000$, the error bars are too dense to be seen.

**Discussion**

For $r_{\text{min}} = 25$ and $r_{\text{max}} = 200$, significant regions were found also when randomly shuffling the genome. Hence, the results for these parameter values should not be trusted and suggest that the method of correcting for the $p$-values is not appropriate. However, when increasing the size of the regions of search ($r_{\text{min}} = 200$ and $r_{\text{max}} = 1000$), we see positive results for the real genome, but not the shuffled one. This indicates that our algorithm detects a real biological signal. However, when inspecting table 2, we discover that our findings are not as expected. None of the significant regions shown seem to correspond to the Genebank ncRNA genes annotated in the genome browser, nor do the results seem to agree with [10]. The genes overlapping with significant regions are of various types, but seem to be enriched in membrane transporter proteins.

Surprisingly, figure 3 shows no big difference in the folding pattern of the most significant region in table 2 and a random transcript. This leads us to think that RNA strands being completely random also may attain secondary structures resembling those of real ncRNAs. Hence, we it seems likely that structural features does not distinguish ncRNA genes from other genomic elements, agreeing with [15].

According to [4] an integrated approach taking several kinds of information into account, is probably a more fruitful approach. For instance [16] uses thermody-
namic stability (based on secondary structure) in addition to evolutionary multiple alignments and obtains results which are very good. Hence, we might think the same about hairpin: Even though the procedure by itself is insufficient to predict ncRNA genes, it can probably be used in combination with other methods in order to obtain more accurate and robust results.

The bias resulting in the difference between figure 2c and figure 2d still needs an explanation. It could due to features such as repeats or a special composition of nucleotides in the real biological sequences.

**Conclusion**

Hoping to predict ncRNA genes *ab initio* using the secondary structure only, turned to be too optimistic. Even though the secondary structure is essential for the function of ncRNA, their structures seem to drown in random noise. There is hope however, that when used in combination with other kinds of information, hairpin can be useful to predict ncRNA genes.

**References**

1. Eddy, S.R.: Noncoding rna genes. Curr. Opin. Genet. Dev. 9(6), 695–699 (1999)
2. Meyer, I.M.: A practical guide to the art of RNA gene prediction. Briefings in Bioinformatics 8(6), 396–414 (2007). doi:10.1093/bib/bbm011. http://oup.prod.sis.lan/bib/article-pdf/8/6/396/690234/bbm011.pdf
3. Svboda, P., Cara, A.D.: Hairpin rna: a secondary structure of primary importance. Cellular and Molecular Life Sciences CMLS 63(7), 901–908 (2006). doi:10.1007/s00018-005-5558-5
4. Klein, R., Misulovin, Z., Eddy, S.: Noncoding RNA genes identified in AT-rich hyperthermophiles. PROCEEDINGS OF THE NATIONAL ACADEMY OF SCIENCES OF THE UNITED STATES OF AMERICA 99(11), 7542–7547 (2002). doi:10.1073/pnas.11206379
5. Sewer, A., Paul, N., Landgraf, P., Aravin, A., Pfeffer, S., Brownstein, M., Tuschi, T., van Nimwegen, E., Zavolan, M.: Identification of clustered microRNAs using an ab initio prediction method. BMC BIOINFORMATICS 6 (2005). doi:10.1186/1471-2105-6-26
6. Priwitzer, B., Hofacker, I.L., Stadler, P.F.: Prediction of locally stable RNA secondary structures for genome-wide surveys. Bioinformatics 20(2), 186–190 (2004). doi:10.1093/bioinformatics/btg388. http://oup.prod.sis.lan/bioinformatics/article-pdf/20/2/186/532778/btg388.pdf
7. Noel, K., Wiese, K.: Exploring the use of stem-loop characteristics for pinpointing structural RNA genes. In: 2004 IEEE COMPUTATIONAL SYSTEMS BIOINFORMATICS CONFERENCE, PROCEEDINGS, pp. 533–534. IEEE COMPUTER SOC, 10662 LOS VAQUEROS CIRCLE, PO BOX 3014, LOS ALAMITOS, CA 90720-1264 USA (2004). IEEE Comp Soc; Hewlett-Packard Co; Coop Platinum; BioMed Cent; Off Sci; US DOE. IEEE Computational Systems Bioinformatics Conference (CSB 2004), Stanford, CA, AUG 16-19, 2004
8. Cock, P.P., Antao, T., Chang, J.T., Chapman, B.A., Cox, C.J., Dalke, A., Friedberg, I., Hamelryck, T., Kauff, F., Wilczynski, B., de Hoon, M.J.L.: Biopython: freely available python tools for computational molecular biology and bioinformatics. Bioinformatics 25(11), 1422–1423 (2009). doi:10.1093/bioinformatics/btp163
9. van der Walt, S., Colbert, S.C., Varoquaux, G.: The numpy array: A structure for efficient numerical computation. Comput. Sci. Eng. 13(2), 22–30 (2011)
10. Rivas, E., Klein, R.J., Jones, T.A., Eddy, S.R.: Computational identification of noncoding rnas in e. coli by comparative genomics. Current Biology 11(17), 1369–1373 (2001). doi:10.1016/S0960-9822(01)00401-8
11. Kerpedjiev, P., Hammer, S., Hofacker, I.L.: Forna (force-directed rna): Simple and effective online rna secondary structure diagrams. Bioinformatics 31(20), 3374–3376 (2015). doi:10.1093/bioinformatics/btv372
12. Benjamini, Y., Yekutieli, D.: The control of the false discovery rate in multiple testing under dependency. Ann. Stat. 29(4), 1165–1188 (2001)
13. Riley, M., Abe, T., Arnaud, M.B., Berlyn, M.K.B., Blattner, F.R., Chaudhuri, R.R., Glaser, J.D., Horiuichi, T., Kiesler, I.M., Kosuge, T., Mori, H., Perna, N.T., Plunkett, G., Rudd, K.E., Serres, M.H., Thomas, G.H., Thomson, N.R., Wishart, D., Wanner, B.L.: Escherichia coli k-12: a cooperatively developed annotation snapshot - 2005. Nucleic Acids Res. 34(1), 1–9 (2006). doi:10.1093/nar/gkj405
14. Chan, P.P., Holmes, A.D., Smith, A.M., Tran, D., Lowe, T.M.: The ucsf archael genome browser: 2012 update. Nucleic Acids Res. 40(01), 646–652 (2012). doi:10.1093/nar/gkr990
15. Rivas, E., Eddy, S.R.: Secondary structure alone is generally not statistically significant for the detection of noncoding rnas. Bioinformatics 16(7), 583–605 (2000)
16. Washietl, S., Hofacker, I.L., Stadler, P.F.: Fast and reliable prediction of noncoding rnas. Proc. Natl. Acad. Sci. U. S. A. 102(7), 2454–2459 (2005). doi:10.1073/pnas.0409169102

**Competing interests**

The author declare that they have no competing interests.

**Author’s contributions**

Not applicable
Availability of data and materials
The genome used in the analysis is the complete genome of *Escherichia coli* strain K-12 substrain MG1655, available as NC_000913.3 from RefSeq. At http://microbes.ucsc.edu, the same assembly is available in the genome browser. The software in addition to the analyses being run in this experiment are found on https://github.com/yaccos/hairpin.

Acknowledgements
Thanks to Finn Drabløs for providing positive feedback to this article.
Pseudocode

Algorithm 1 Method for finding hairpins in a RNA transcript

Require: RNA string \((s_i)\) of length \(n\), minimal base-pairing distance \(d_{\text{min}}\), maximal base-pairing distance \(d_{\text{max}}\), minimal stem size \(l_{\text{min}}\) to be considered.

Create array \((A_{ij})\) with dimensions \((d_{\text{max}} - d_{\text{min}} + 1) \times n\), each element initialized to \(\text{False}\)

for \(d \leftarrow d_{\text{min}}\) to \(d_{\text{max}}\) do
  for \(i \leftarrow 1\) to \(n - d\) do
    if Base \(s_i\) can form a base pair with base \(s_{i+d}\) then
      \(A_{i,d} \leftarrow \text{True}\)

Search through the matrix \(A\) for pairs \((t,d)\) such that there exists an \(l\) (taken to be the maximum among all candidates) such that:

- \(l > l_{\text{min}}\)
- \(A_{t,d-d_{\text{min}}} A_{t+1,d-d_{\text{min}}-2} \ldots A_{t+l,d-d_{\text{min}}-2(l-1)}\) are \(\text{True}\)

for Each triple \((t,d,l)\) found do
  Report \(t\) as the starting nucleotide
  Report \(t + d\) as the end nucleotide
  Report \(d + 1\) as the motif length

return \((\xi_i)\), the array of motifs found

Algorithm 2 Method for computing base-pairing count

Require: Start position \(a\) and end position \(b\) of the RNA region of probe, \((\xi_i)\) list of motifs.

Initialize a set \(S \leftarrow \emptyset\) which only can hold unique elements
for Each motif \(\xi_i\) in \((\xi_i)\) do
  if The start position of \(\xi_i\) is greater or equal to \(a\) and the end position of \(\xi_i\) is less or equal to \(b\) then
    Add the base-pairing nucleotides of \(\xi_i\) to \(S\)
return \(|S|\), the number of elements in \(S\)

Algorithm 3 Computing randomized RNA strands

Require: RNA strand \((s_i)\), \(n_{\text{rand}}\) the number of randomizations, \(r_{\text{min}}\) the minimum length of regions to search, \(r_{\text{max}}\) the maximum length of the regions to search

Allocate \(C\), a \(n_{\text{rand}} \times (r_{\text{max}} - r_{\text{min}} + 1)\) matrix of integers.
for \(i \leftarrow 1\) to \(n_{\text{rand}}\) do
  Generate an RNA string \((m_i)\) of length \(r_{\text{max}}\) by sampling with replacement from \(s\)
  for \(j \leftarrow r_{\text{min}}\) to \(r_{\text{max}}\) do
    Let \((g_j)\) be the first \(j\) nucleotides of \(m\)
    Find the hairpins \((\xi_i)\) of \(g\)
    \(C_{i,j-r_{\text{min}}+1} \leftarrow \text{The combined base-pairing count for all motifs in } \xi_i\)
Let \((\mu_i)\) be the columnwise means of \(C\)
Let \((\sigma_i)\) be the columnwise standard deviations of \(C\)
return \((\mu_i, \sigma_i)\)
Algorithm 4 Finding putative ncRNA coding regions

Require: \( r_{\text{sequence}} \) the length of the original RNA sequence, \( r_{\text{min}} \) the minimum length of the region to search, \( r_{\text{max}} \) maximum length of region to search, \((\xi_i)\) list of motifs, \( \mu_i \) and \( \sigma_i \) from algorithm 3, \( q_{\text{crit}} \) critical q-value

Let \( S \leftarrow \emptyset \) be a set of regions
for \( i = 1 \) to \( \text{Length} \ (\xi) \) do
  for \( j = 1 \) to \( \text{Length} \ (\xi) \) do
    \( r \leftarrow \xi_i.\text{End}_\text{nucleotide} - \xi_i.\text{Start}_\text{nucleotide} + 1 \) (Length of region)
    if \( r_{\text{min}} \leq r \leq r_{\text{max}} \) then
      Let \( c \) be the base-pairing count in motifs between \( \xi_j.\text{End}_\text{nucleotide} \) and \( \xi_i.\text{Start}_\text{nucleotide} \) inclusive.
      \( z_{\text{obs}} \leftarrow \frac{c - \mu_r - \mu_{r_{\text{min}} + 1}}{\sigma_r - \sigma_{r_{\text{min}} + 1}} \)
      \( p_{\text{value}} \leftarrow 1 - \Phi(z_{\text{obs}}) \) (\( \Phi \) is the cumulative function of the standard normal distribution)
      \( p_{\text{value}} \leftarrow 1 - (1 - p_{\text{value}}) \)
      Add the region between \( \xi_i.\text{Start}_\text{nucleotide} \) and \( \xi_j.\text{End}_\text{nucleotide} \) inclusive, to \( S \)
  end
end
\( R \leftarrow \emptyset \)
while \( S \) is not empty do
  Pick \( \rho \), the region with lowest \( p_{\text{value}} \) from \( S \)
  Remove \( \rho \) from \( S \) together with all regions in \( S \) overlapping with \( \rho \)
  Add \( \rho \) to \( R \)
end
Compute the \( q \)-values by the Benjamini–Yekutieli procedure for the regions in \( R \)
Throw away all regions in \( R \) having a higher \( q \)-value than \( q_{\text{crit}} \)
return \( R \)
(e) RNA structure corresponding to most significant region found in table 2

(f) RNA structure corresponding to a random reshuffling to the genome

Figure 3: RNA structure generated with forna
### Tables

**Table 1: Number of significant regions found for each of the runs**

| Genome   | $r_{min}$ | $r_{max}$ | Number of significant regions found |
|----------|-----------|-----------|-------------------------------------|
| Original | 25        | 200       | 12791                               |
| Shuffled | 25        | 200       | 8400                                |
| Original | 200       | 1000      | 67                                  |
| Shuffled | 200       | 1000      | 0                                   |

**Table 2: Position, $q$-value and overlapping genes for the most significant regions found with $r_{min} = 200$, $r_{max} = 1000$ on the original genome**

| Start position | End position | $q$-value | Symbol of overlapping genes | Product of overlapping genes | Degree of overlap |
|----------------|--------------|-----------|------------------------------|-----------------------------|------------------|
| 71218          | 72132        | 0.0e+00   | yabI                         | Uncharacterized membrane-associated protein | Large            |
|                |              |           | araC                         | DNA-binding transcriptional dual regulator | Smaller           |
| 48041          | 48825        | 1.5e-07   | kefC                         | Potassium:proton antiporter | Complete          |
| 4383506        | 4384503      | 6.1e-07   | yjeN yjeO yjeP               | Predicted protein           | Complete Medium   |
| 130212         | 130669       | 6.1e-07   | yacH                         | Predicted protein           | Complete          |
| 171796         | 172792       | 3.1e-06   | fnuB                         | Fused non-hydroxamate transporter subunits of ABC superfamily: membrane components | Complete          |
| 3249581        | 3250344      | 1.6e-05   | yhaH yqjG                    | Predicted inner membrane protein Glutathionyl-hydroquinone reductase | Small Large       |
| 4325823        | 4326379      | 2.6e-05   | yjdA                         | Clamp-binding protein       | Complete          |
| 1943798        | 1944089      | 9.4e-05   | ruvA                         | Component of RuvABC resolvasome, regulatory subunit | Large            |
| 375929         | 376420       | 9.7e-05   | yaiL                         | Predicted nucleoprotein/polynucleotide-associated enzyme | Large            |
| 521109         | 521866       | 9.7e-05   | ybbP                         | Uncharacterized ABC transporter permease | Complete          |
| 522460         | 523251       | 1.2e-04   | rhsD                         | rhsD element protein        | Large            |
| 601374         | 602194       | 1.5e-04   | pheP                         | Phenylalanine transporter   | Complete          |
| 2510237        | 2510847      | 1.5e-04   | mntH                         | Manganese/divalent cation transporter | Large            |
| 3239387        | 3239621      | 1.5e-04   | ygjV                         | Conserved inner membrane protein | Complete          |
| 360096         | 361011       | 1.8e-04   | cynX lacA                    | Predicted cyanate transporter thiogalactoside acetyltransferase | Medium Large     |
| 276322         | 276669       | 1.9e-04   | mmuM                         | 5-Methyltetrahydrofolate:homocysteine methyltransferase | Complete          |
| 4106427        | 4106655      | 2.2e-04   | pfkA                         | 6-Phosphofructokinase I     | Medium            |
| 355262         | 355595       | 2.4e-04   | codB codA                    | Cytosine transporter        | Medium Large      |
| 1551966        | 1552394      | 3.4e-04   | maeA                         | Malate dehydrogenase, (decarboxylating, NAD-requiring) (malic enzyme) | Large            |
| 2605530        | 2606333      | 5.4e-04   | hydF                         | Hydrogenase 4, membrane subunit | Complete          |
| 2043860        | 2044194      | 5.4e-04   | yeeJ                         | Adhesin                     | Complete          |