SOFTWARE

Hairpin: a tool for predicting structural non-coding RNA genes *ad initio*

Jakob Peder Pettersen

Correspondence:

jakobpeder.pettersen@gmail.com

Department of Biotechnology and Food Science, NTNU - Norwegian University of Science and Technology, Sem Sælandsvei 6-8, 7491 Trondheim, Norway
Full list of author information is available at the end of the article

Abstract

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. *Ab initio* methods for prediction exist, but are not that much explored.

Results: We introduce *hairpin* which identify potential structural RNA genes only based on the sequence. We use the algorithm to predict RNA genes in *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.

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-pairings is in general a futile approach.

Keywords: hairpin; ncRNA genes; gene prediction; 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 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 [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 *sensu* Meyer [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 [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-pairing partners at a decreasing distance (decrease with two for each nucleotide), is considered a motif. We hence include the parameter l_{min} telling the minimum length of such a motif in order to report it. Setting l_{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.

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:

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

where:

- A_{rand} is the number of regions of length r in the genome having an equal or higher base-pairing count than c_{obs} under the null hypothesis
- c_{rand} is the base-pairing count in a interval of length r under the null hypothesis
- c_{obs} is the observed base-pairing count
- $n_{interval}$ is the number of disjoint regions in the genome of length r
- r is the length of the region
- r_{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_{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 $d_{min} = 4$, $d_{max} = 25$, $q_{crit} = 0.01$, $l_{min} = 8$ and $n_{rand} = 2000$, the algorithm was run on the genome of *Eschericia coli* K12 [13], with four variations:

- $r_{min} = 25$ and $r_{max} = 200$ on the original genome
- $r_{min} = 25$ and $r_{max} = 200$ on a random reshuffling of the genome
- $r_{min} = 200$ and $r_{max} = 1000$ on the original genome
- $r_{min} = 200$ and $r_{max} = 1000$ 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 $r_{min} = 200$, $r_{max} = 1000$ 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 $r_{min} = 200$ and $r_{max} = 1000$, the error bars are too dense to be seen.

Discussion

For $r_{min} = 25$ and $r_{max} = 200$, significant regions were found also when randomly shuffling the genome. Hence, the results for these parameter value 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 ($r_{min} = 200$ and $r_{max} = 1000$), 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 be 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 out 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](https://doi.org/10.1093/bib/bbm011). <http://oup.prod.sis.lan/bib/article-pdf/8/6/396/690234/bbm011.pdf>
3. Svoboda, 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](https://doi.org/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](https://doi.org/10.1073/pnas.11206379)
5. Sewer, A., Paul, N., Landgraf, P., Aravin, A., Pfeffer, S., Brownstein, M., Tuschl, 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](https://doi.org/10.1186/1471-2105-6-26)
6. Privitzer, 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](https://doi.org/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.J.A., 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](https://doi.org/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](https://doi.org/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](https://doi.org/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., Glasner, J.D., Horiuchi, T., Keseler, 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](https://doi.org/10.1093/nar/gkj405)
14. Chan, P.P., Holmes, A.D., Smith, A.M., Tran, D., Lowe, T.M.: The ucsc archaeal genome browser: 2012 update. *Nucleic Acids Res.* **40**(D1), 646–652 (2012). doi:[10.1093/nar/gkr990](https://doi.org/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](https://doi.org/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 $r_{sequence}$, d_{min} minimal base-pairing distance, d_{max} maximal base-pairing distance and l_{min} minimal stem size to be considered.

Create array (A_{ij}) with dimensions $(d_{max} - d_{min} + 1) \times n$, each element initialized to **False**
for $d \leftarrow d_{min}$ **to** d_{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:

- $l > l_{min}$
- $A_{t,d-d_{min}}, A_{t+1,d-d_{min}-2}, \dots, A_{t+i,d-d_{min}-2i}, \dots, A_{t+(l-1),d-d_{min}-2(l-1)}$ all are **True**

for Each triple (t, d, l) found **do**
 Report t as the starting nucleotide
 Report $t + d$ as the end nucleotide
 Report the $d + 1$ as the motif length
return (ξ_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, (ξ_i) list of motifs.

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

Algorithm 3 Computing randomized RNA strands

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

Allocate C , a $n_{rand} \times (r_{max} - r_{min} + 1)$ matrix of integers.
for $i = 1$ **to** n_{rand} **do**
 Generate an RNA string (m_i) of length r_{max} by sampling with replacement from s
 for $j = r_{min}$ **to** r_{max} **do**
 Let (g_i) be the first j nucleotides of m
 Find the hairpins (ξ_i) of g
 $C_{i,j-r_{min}+1} \leftarrow$ The combined base-pairing count for all motifs in ξ
Let (μ_i) be the columnwise means of C
Let (σ_i) be the columnwise standard deviations of C
return μ, σ

Algorithm 4 Finding putative ncRNA coding regions

Require: $r_{sequence}$ the length of the original RNA sequence, r_{min} the minimum length of the region to search, r_{max} maximum length of region to search, (ξ_i) list of motifs, μ_i and σ_i from algorithm 3, q_{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**

$r \leftarrow \xi_j.\text{End_nucleotide} - \xi_i.\text{Start_nucleotide} + 1$ (Length of region)

if $r_{min} \leq r \leq r_{max}$ **then**

 Let c be the base-pairing count in motifs between $\xi_j.\text{End_nucleotide}$ and $\xi_i.\text{Start_nucleotide}$ inclusive.

$z_{obs} \leftarrow \frac{c - \mu_r - r_{min} + 1}{\sigma_r - r_{min} + 1}$

$p_{value} \leftarrow 1 - \Phi(z_{obs})$ (Φ is the cumulative function of the standard normal distribution)

$p_{value} \leftarrow 1 - (1 - p_{value})^{\frac{r_{sequence}}{r}}$

 Add the region between $\xi_i.\text{Start_nucleotide}$ and $\xi_j.\text{End_nucleotide}$ inclusive, to S

$R \leftarrow \emptyset$

while S is not empty **do**

 Pick ρ , the region with lowest p_{value} from S

 Remove ρ from S together with all regions in S overlapping with ρ

 Add ρ to R

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_{crit}

return R

Figures

(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	yabl	Uncharacterized membrane-associated protein	Large
			araC	DNA-binding transcriptional dual regulator	Small
48041	48825	1.5e-07	kefC	potassium:proton antiporter	Complete
4383506	4384503	6.1e-07	yjeN	predicted protein	Large
			yjeO	conserved inner membrane protein	Complete
			yjeP	predicted mechanosensitive channel	Medium
130212	130669	6.1e-07	yacH	predicted protein	Complete
171796	172792	3.1e-06	fhuB	fused iron-hydroxamate transporter subunits of ABC superfamily: membrane components	Complete
3249581	3250344	1.6e-05	yhaH	predicted inner membrane protein	Small
			yqjG	Glutathionyl-hydroquinone reductase	Large
4325823	4326379	2.6e-05	yjdA	Clamp-binding protein	Complete
1943798	1944089	9.4e-05	ruvA	component of RuvABC resolvase, 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	predicted cyanate transporter	Medium
			lacA	thiogalactoside acetyltransferase	Large
276322	276669	1.9e-04	mmuM	S-methylmethionine:homocysteine methyltransferase	Complete
4106427	4106655	2.2e-04	pfkA	6-phosphofructokinase I	Medium
355262	355595	2.4e-04	codB	cytosine transporter	Medium
			codA	cytosine deaminase	Large
1551966	1552394	3.4e-04	maeA	malate dehydrogenase, (decarboxylating, NAD-requiring) (malic enzyme)	Large
2605530	2606333	5.4e-04	hyfF	hydrogenase 4, membrane subunit	Complete
2043860	2044194	5.4e-04	yeeJ	Adhesin	Complete