

Identification of CpG islands in genomic sequences

Emily Mitchell

Abstract:

CpG islands are stretches of DNA in which there is a higher than normal concentration of guanine and cytosine. These areas are interesting because they make excellent gene markers and also because of the role they play in gene regulation. We have created a new computer program based on Takai and Jones' algorithm for the location of CpG islands in *in silico* DNA. We have also proposed a change to Takai and Jones' algorithm which allows for the extraction of longer CpG islands. The command line version of our new program, CpGspy, is available for use on request, as is the source code for a GUI-based version of our program which gives users graphical results that are much easier to analyze than typical text-based results.

Introduction:

CpG islands are unmethylated regions of mammalian genomes with unusually high concentrations of guanine (G) and cytosine (C) in them. For example, in humans, the genome average of G + C content is ~ 41% (Antequera 2003, Venter, J.C., et al., 2001). In human CpG islands, the concentration is ~ 67%—considerably higher than the genome average (Antequera 2003). CpG islands are extremely useful as predictors of where a gene can be found in a genome, since roughly 60% of human (Robinson et al. 2004) and 50% of all mammalian genes (Ioshikhes and Zhang 2000) contain unmethylated CpG islands which mark the promoter and exonic regions of genes. While CpG islands can be found in non-promoter regions of the gene, CpG islands in promoter regions tend to have a higher G + C content and a higher observed/expected ratio (Takai and Jones 2002). The promoter region of a gene is usually located between the 5' end of the CpG island and the gene's transcription initiation site (Antequera 2003). CpG islands

might serve as replication origins of DNA, too; their chromatin structure suggests this possibility (Antequera 2003).

The most important feature of CpG islands, beside their habit of overlapping the promoter region of genes, is the fact that they remain unmethylated when about 80% of the CpG dinucleotides in the genome succumb to methylation. The methylated CpG is susceptible to deamination, and once deaminated, the mutation rate is 10-50 times higher than the mutation rate in other dinucleotides. Methylation is important: it can prevent transcription of the DNA of invading molecular parasites, and also prevent recombination between repetitive sequences of DNA, thus keeping the genome stable (Antequera 2003). Methylation of a gene's CpG island silences the gene, since it prevents the transcription factors from binding to the DNA in question. This is not a typical method of transcription regulation, however, because the CpG island associated with a tissue-specific gene remains unmethylated whether or not that gene is turned on in a particular cell (Antequera 2003).

Now that several mammalian genomes have been sequenced and more are on the way, identifying CpG islands in all the genomes and comparing their gene associations, distributions, and phylogenetic relationships is becoming possible. There is a wealth of information to be gained here.

There are several existing computer programs, written in a variety of languages, which locate CpG islands in DNA sequences. The most well known of these programs, CpGi130, was written by Takai and Jones in Perl and published in 2002. The algorithm they created for the extraction of CpG islands is now widely accepted and has been recreated, with some changes, in other programs like CpGIE (Wang and Leung 2003).

Before explaining Takai and Jones' algorithm, a few words on some terminology required for the explanation: A "window" is a stretch of 500 (or another user-defined number) nucleotides. "Reading" a window means that the computer looks at that stretch of DNA, counts Cs, Gs, and CGs, and calculates the G + C content and observed/expected ratio for that window. "Sliding" implies moving the window one nt in either direction, depending on the context; the next window will overlap with the window previously read by window length minus 1 nt.

Takai and Jones' algorithm goes like this:

1. Read in a window.
2. If it does not meet CpG island criteria, slide 1 nt toward the 3' end and read in the next window.
3. Keep sliding like this until you find a window that meets CpG island criteria.
4. Once you have found a window that meets the criteria, jump ahead to the next adjacent but non-overlapping window and read that.
5. Keep jumping toward the 3' end in this manner until you find a window that does not meet CpG island criteria.
6. When you have a window that does not meet CpG island criteria, slide back toward the 5' end 1 nt at a time, reading in each window and evaluating it until you find a window that meets CpG island criteria.
7. Now look at all the adjacent/overlapping windows that you've just located, all of which meet CpG island criteria. Evaluate them as one big window.
8. If that window as a whole meets CpG island criteria, move on to the next non-overlapping window and start over at step 1 of this algorithm.

9. If that big window as a whole does not meet CpG island criteria, trim 1 nt from its 3' end and evaluate it. If it still does not meet the criteria, trim 1 nt from its 5' end and evaluate it. Continue following this pattern until you locate an island that meets CpG island criteria.
10. If you have to keep trimming until the island is too short to meet CpG island criteria, revert to the first window of this island that you found.
11. Go back to step 1.

Takai and Jones' algorithm is well-respected and generally reliable. Unfortunately, however, their implementation of it is very slow, making CpGi130 a cumbersome tool to use. Other implementations of it, like CpGIE, make slight but significant changes that can have extreme impacts on results.

Methods:

Our program, CpGspy, is written in Java. It allows users to choose what parameters for minimum island length (window size), maximum separation between two islands to be combined, minimum OE ratio, and minimum G + C content they would like to run the program with. It also gives users the choice of running it on a sequence using either Takai and Jones' algorithm or our modified version of their algorithm.

That modification to the algorithm falls under the trimming step (step 9 in the list above). Instead of trimming one nt from the 3' end, evaluating, trimming one nt from the 5' end, evaluating, trimming one nt from the 3' end, etc., our modified version of the algorithm slides an ever-shrinking window from 5' to 3' end within the start and end boundaries established when the extended island is examined as a whole (step 7 in the list above). Thus, if the extended but

as-yet-untrimmed island stretched from 1 to 900, our version of the algorithm would evaluate the following windows: 1 - 900, 1 - 899, 2 - 900, 1 - 898, 2 - 899, 3 - 900, 1 - 897, 2 - 898, 3 - 899, 4 - 900, etc., until it found the longest possible island in that stretch of DNA that meets CpG island criteria.

This prevents islands from being trimmed away to nothing and having to be reset to their first window due to the way the CG dinucleotides are distributed throughout the island, as happens in a noticeable percentage of CpG islands. In our tests, 36 of 692 CpG islands in mouse chromosome 18 and 175 of 3,958 CpG islands in human chromosome 1 were trimmed away to nothing by Takai and Jones' unmodified algorithm and had to be reset.

Results:

When CpGspy runs using Takai and Jones' algorithm, it returns identical results to CpGi130. The only exception to this is in the case of CpGi130 returning invalid results that do not meet the criteria the user input. We found two such islands in our results from CpGi130 running on mouse chromosome 18; in both cases the OE ratio was below the threshold we had told CpGi130 to use. CpGspy returned shorter islands that met the user-input criteria in those cases.

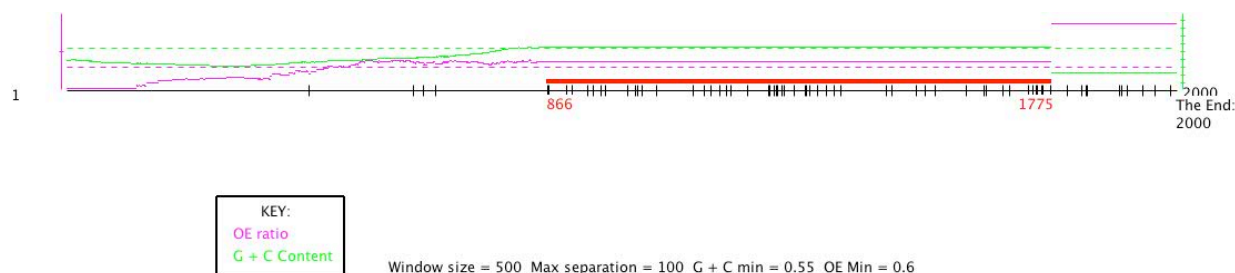
Our modified version of Takai and Jones' algorithm is computationally more expensive than Takai and Jones' symmetrical method of trimming, but it can also make a surprising difference in the length of the final CpG island. In the case of one CpG island in mouse chromosome 18, our modified algorithm ended up with an island 910 nt long while the original Takai and Jones algorithm had to revert to the original window of the island, resulting in a 500 nt long island in its results. The difference in results between the two algorithms usually is not this extreme. Often both algorithms return identical results. But it does not make sense to write off so much of

an island simply because it could not be trimmed symmetrically, particularly when the added cost in computation came out to a mere 2-minute difference in run time on human chromosome 1 (approximately 245 million nt) on the laptop we tested the program on.

To see just how much of a difference our algorithmic change made on the overall length of the CpG islands, we summed up the lengths of all the islands found in mouse chromosome 18 by CpGIE, CpGi130 and our program when all three were run with the same parameters. CpGi130 came up with a total length of 678,757 nt; CpGIE had a total length of 697,820 nt, and CpGspy gave a total length of 701,716 nt. Since all three CpG island finding programs are returning stretches of the contig that meet the CpG island criteria input by the user, it makes sense for a longer total island length to be desirable. After all, if the stretch of DNA meets CpG island criteria, isn't it better to identify it than not to notice it?

Even with the increased computation time needed for our algorithmic change, the command line version of our program ran on all of mouse chromosome 18 (approximately 93 million nt) in 32 minutes on our MacBook with its 2 GHz processor. This compares favorably with CpGIE, as CpGIE took nearly 2 hours to run on the same chromosome on the same computer. Comparing our program to CpGi130, we ran both programs on mouse chromosome 18 on our Dell Inspiron 5100. Our program took slightly more than an hour on that computer; CpGi130 took slightly more than 3 days. Clearly, then, our program has the advantage when it comes to speed.

The GUI version of our program adds a user-friendly interface to the speed of our command line program. It also adds capability: it can create and save graphical results depicting up to 60 kilobases worth of the contig.



The results above are from a 2 kilobase test sequence pulled from mouse chromosome 18. Each tick on the black horizontal line marks the location of a CG dinucleotide. The dashed magenta line shows the minimum observed CpG/expected CpG (OE) ratio, the dashed green line shows the minimum G + C content, and the solid magenta and green lines are OE ratio for the window starting at that nucleotide and G + C content for the window starting at that nucleotide, respectively. The only exception to this is where the lines run above the CpG island (red line): in that case, the green and magenta lines show the C + G content and the OE ratio for the island. One known bug in these graphical results is that if two islands are close enough to combine but do not end up being combined, the lines above the two islands will show what would have been the OE ratio and G + C content for the combined island as a whole.

Despite that bug, we believe these graphical results to be superior to those of CpGIE because they give more information than CpGIE's do. CpGIE's graphs do not show the OE ratio, nor do they show the G + C content. Furthermore, CpGIE does not give users the option of seeing only a portion of their results graphically. This is problematic when the user runs CpGIE on a long sequence such as all of mouse chromosome 18, because the results take several minutes to draw, leaving CpGIE frozen or sluggish until the drawing is done. Also, we have been unable to find a way to find a way to save CpGIE's graphical results short of taking screen shots.

We have been unable to turn the GUI version of our program into a java archive that users can run from the command line, but the working source code is available on request. Anyone

interested should contact Emily Mitchell at emitche2@gmail.com. The command line version of our program, which has been made into a jar file that can be run from the command line of any computer that has Java, is also available on request.

Conclusions:

CpGspy is superior to the other CpG-island detection programs that we tested it against for several reasons. It runs faster than CpGi130 and CpGIE. It can be used to return the exact results CpGi130 would give, and takes a much shorter time to run. And if users choose to use our version of Takai and Jones' algorithm, CpGspy detects a greater length of islands overall than either CpGi130 or CpGIE.

For those with the capability to run the GUI version of CpGspy, our program combines all of those advantages with the fact that it gives useful graphical results to make the study and comparison of CpG islands much easier for researchers.

It is our hope that even though the GUI version of our program has not been packaged into a user-friendly jar file that can be run easily by those with little or no programming experience, someone will eventually take our source code and find a way to package it usefully for distribution. It is also our hope that those with an interest in CpG islands will feel free to take and modify our source code as they see fit in order to benefit the progress of research on the topic.

Acknowledgments:

Thank you to Ali Erkan and Zhongming Zhao for their mentoring. This work was supported by the NIH and the NSF's grant to the VCU BBSI.

Resources:

- Antequera, F. 2003. "Structure, function and evolution of CpG island promoters." CMLS.
- Antequera, F., and Adrian Bird. 1993. "Number of CpG islands and genes in human and mouse." Proc. Natl. Acad. Sci. USA. 90: 11995-9.
- Gardiner-Garden, M., and M. Frommer. 1987. "CpG Islands in vertebrate genomes." Journal of Molecular Biology. Volume 196, Issue 2, p. 261-282.
- Hannenhali, S., and Samuel Levy. 2001. "Promoter prediction in the human genome." Bioinformatics. 17 Suppl.
- Ioshikhes, I. P. and Michael Q. Zhang. 2000. "Large-scale human promoter mapping using CpG islands." Nature Genetics. 26.
- Ponger, L., and Dominique Mouchiroud. 2002. "CpGProD: identifying CpG islands associated with transcription start sites in large genomic mammalian sequences." Bioinformatics. 18.4.
- Robinson, P. N. et al. 2004. "Gene-Ontology analysis reveals association of tissue-specific 5' CpG-island genes with development and embryogenesis." Human Molecular Genetics. 13: 1699-78.
- Takai, D., and Peter Jones. 2002. "Comprehensive analysis of CpG islands in human chromosomes 21 and 22." PNAS.
- Takai, D., and Peter Jones. 2003. "The CpG Island Searcher: A New WWW Resource." In Silico Biology 3, 0021.
- Venter, J.C., et al., 2001. "The sequence of the human genome." Science 291, 1304-1351.
- Wang, Yong and Frederick C.C. Leung. "An evaluation of new criteria for CpG islands in the human genome as gene markers." Bioinformatics 20.7: 1170-1177.
- Yamashita, R., et al. 2005. "Genome-wide analysis reveals strong correlation between CpG islands with nearby transcription start sites of genes and their tissue specificity." Gene 350: 129-136.