Bioinformatics
doi.10.1093/bioinformatics/xxxxxx
Advance Access Publication Date: Day Month Year
Manuscript Category

OXFORD

Subject Section

pqsfinder web: G-quadruplex identification using optimized pqsfinder algorithm

Dominika Labudová [1], Jiří Hon [1] and Matej Lexa [2]*

[1] IT4Innovations Centre of Excellence, Faculty of Information Technology, Brno University of Technology, Božetěchova 2, 61266 Brno, Czech Republic
[2] Faculty of Informatics, Masaryk University, Botanická 68a, 60200 Brno, Czech Republic

* To whom correspondence should be addressed.

Associate Editor: XXXXXXX

Received on XXXXX; revised on XXXXX; accepted on XXXXX

Abstract

Motivation: G-quadruplex is a DNA form in which four guanine-rich regions are held together by Hoogsteen bonding between guanine nucleotides in coordination with potassium ions. G-quadruplexes are increasingly seen as a biologically important component of genomes. Their detection *in vivo* is problematic, however, sequencing and spectrometric techniques exist for *in vitro* detection in isolated DNA. *In silico* methods can be used to analyze nucleotide sequences for potential quadruplex-forming sequences (PQS). We previously devised the *pqsfinder* algorithm for identification of PQS, implemented it in C++ and published it as an R/Bioconductor package. We looked for ways to optimize *pqsfinder* for faster and user-friendly sequence analysis.
Results: We identified two weak points where *pqsfinder* could be optimized. We modified the internals of the recursive algorithm to avoid matching and scoring many sub-optimal PQS conformations that are later discarded. To accommodate the needs of a broader range of users, we created a website for submission of sequence analysis jobs that does not require knowledge of R to use *pqsfinder*.
Availability: https://pqsfinder.fi.muni.cz
Contact: lexa@fi.muni.cz

1 Introduction

In our previous work (Hon *et al.*, 2017) we created *pqsfinder*, an R/Bioconductor package implementing a recursive algorithm to search nucleotide sequences for all combinations of guanine nucleotide clusters (G runs) satisfying a set of rules defining a potential quadruplex-forming sequence (PQS). The algorithm was parametrized to provide the best match between prediction and results of then available experiments (Bedrat *et al.*, 2016; Chambers *et al.*, 2015). Since then, *pqsfinder* has been used to analyze prokaryotic (Mishra *et al.*, 2019; Jain *et al.*, 2018) and eukaryotic genomes (Tokan *et al.*, 2018) as well as used as a reference for similar tools (Berselli *et al.*, 2018; Belmonte Reche and Morales, 2019).

The frequent use of the tool and its application to entire genomes led us to look for improvements that would make it easier for non-programmers to use the tool and to make analyses faster. We optimized the algorithm and the associated code and made these improvements first available in *pqsfinder-2.0* package. The optimized functions were then used in setting up *pqsfinder* web interface running at `https://pqsfinder.fi.muni.cz`.

2 Speed optimization and the new threshold

Based on analysis of *pqsfinder*'s performance issues emerging on DNA sequences with high G content, we implemented two main optimizations: i) tabulation of costly math library functions repetitively used in scoring and ii) reduction of internal search iterations.

The first optimization tabulates bulge length penalties and loop length penalties as they were repetitively computed with the same arguments and use costly math library functions. After optimization, they are computed only once at the beginning of the algorithm.

The second optimization is implemented as an additional skip condition between individual G run searches to reduce the number of search iterations. After each G run identification, a maximal score estimate is

calculated and compared to two values: i) score threshold and ii) maximal score reported so far overlapping the starting position of the first G run in the current PQS. If the maximal score estimate is lower than any of the two values, the search for other G runs is skipped. Consequently, many sub-optimal overlapping PQS are not matched and scored as *pqsfinder* usually finds the highest-scoring PQS in several first iterations.

The speedup of the optimized algorithm can reach three orders of magnitude. It strongly depends on the content and distribution of guanine bases in the sequence. Regions with dense G clusters are processed more than 1000x faster than before, whereas G-sparse regions are processed at approximately the same speed.

We measured the performance gain using the entire human genome sequence, and the speedup was around 800x on standard laptop hardware. The optimized *pqsfinder* algorithm is now convenient for whole-genome analysis and fully competitive in speed with other approaches.

Recently, *pqsfinder*'s default settings showed to be too sensitive, finding too many, possibly irrelevant, hits. The high number of reported PQS is allowed by low default score threshold 26 that we chose as the best-performing threshold on small *in vitro* Lit392 dataset (Hon *et al.*, 2017; Bedrat *et al.*, 2016). Based on later feedback, we decided to increase the score threshold to focus on more stable PQS by default.

We used G4-seq data (Chambers *et al.*, 2015) to find the new score threshold. First, we labeled regions with mismatch levels greater or equal to 25 as positive PQS containing regions. Remaining regions were labeled negative. Second, we randomly sampled regions longer than 50 base pairs to get 50,000 positive and 50,000 negative sequences. Third, we ran *pqsfinder* on all sampled sequences and recorded the maximal PQS score. We found that score threshold 52 best discriminates positive and negative sequences having maximal prediction accuracy 77.3%. Therefore, we chose this value as the new *pqsfinder* minimum score threshold.

3 Web features

The web interface has a client-server architecture. The backend is a REST API implemented in R language using the plumber library and is running in the Stratus.FI cloud. The React framework was used to implement the client application. A unique job ID identifies each user request that can be used to later access the results without having to rerun the computations.

The web application accepts a set of sequences in FASTA format. The sequences can be either uploaded from a file or provided as plain text. We are also planning to support import of gene sequences directly from the NCBI database. The analysis options are by default set to recommended values from the latest *pqsfinder* release but can be easily customized using the web interface.

Identified PQS can be viewed either in a table or an interactive plot. The table contains all important PQS characteristics – start, end, score, strand, number of tetrads, number of bulges, number of mismatches and the nucleotide sequence itself. The sequence is colored to highlight each G run, including bulges and mismatches. The plot visualizes individual positions of the PQS in the sequence and their score (see Figure 1). Results can be downloaded in CSV or GFF formats.

The web server provides pre-computed whole-genome tracks for two human genome releases Grch38 and GRch37. These tracks can be easily imported to UCSC Genome Browser using a track hub link available at the Genomes download page.

4 Conclusions

We report here a major upgrade to a previously published *pqsfinder* software package designed to identify PQS in nucleotide sequences. The software uses a robust scoring scheme that has not been surpassed by other tools so far. However, it lacked in user-friendliness and speed. The reported upgrade provides a speedup of several orders of magnitude on G-rich sequences and introduces a web server for hassle-free access to PQS identification. It should find use in whole-genome sequence analyses as well as small ad-hoc queries by molecular biologists at large.

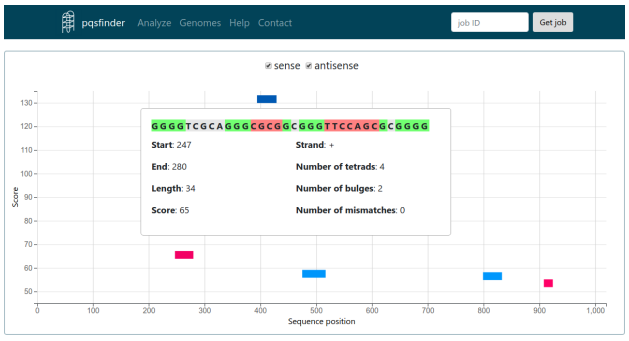

Fig. 1. Identified PQS in an interactive plot displayed on the results page. The tooltip box provides details on selected PQS and visualizes their structure with bulges highlighted in red. The plot can be scaled and filtered to only show PQS on either sense or antisense strand.

Acknowledgements

We thank Tomas Szaniszlo and the rest of the Faculty of Informatics Stratus.FI team for registering the pqsfinder.fi.muni.cz domain and providing a virtual machine for the *pqsfinder* web server. Access to computing and storage facilities owned by parties and projects contributing to the National Grid Infrastructure MetaCentrum provided under the programme Projects of Large Research, Development, and Innovations Infrastructures (CESNET LM2015042), is greatly appreciated.

Funding

This work has been supported by the Czech Science Foundation [15-02891S to M.L.] and ICT tools, methods and technologies for smart cities project of the Brno University of Technology [FIT-S-17-3964 to J.H.].

References

Bedrat, A. *et al.* (2016). Re-evaluation of G-quadruplex propensity with G4Hunter. *Nucleic Acids Res.*, **44**(4), 1746–1759.

Belmonte Reche, E. and Morales, J. C. (2019). G4-iM Grinder: DNA and RNA G-quadruplex, i-Motif and higher order structure search and analyser tool. *bioRxiv*.

Berselli, M. *et al.* (2018). NeSSie: A tool for the identification of approximate DNA sequence symmetries. *Bioinformatics*, **34**.

Chambers, V. *et al.* (2015). High-throughput sequencing of DNA G-quadruplex structures in the human genome. *Nat. Biotechnol.*, **33**.

Hon, J. *et al.* (2017). pqsfinder: an exhaustive and imperfection-tolerant search tool for potential quadruplex-forming sequences in R. *Bioinformatics*, **33**(21), 3373–3379.

Jain, N. *et al.* (2018). G-quadruplex stabilization in the ions and maltose transporters inhibit salmonella enterica growth and virulence. *bioRxiv*.

Mishra, S. *et al.* (2019). Characterization of highly conserved G-quadruplex motifs as potential drug targets in streptococcus pneumoniae. *Sci. Rep.*

Tokan, V. *et al.* (2018). Quadruplex DNA in long terminal repeats in maize LTR retrotransposons inhibits the expression of a reporter gene in yeast. *BMC Genomics*, **19**(1), 1–11.