dna-electrons: A Python Package for Calculating Total Electron Counts in DNA Sequences

Yash Munnalal Gupta*

Department of Biology, Naresuan University

2025-09-30

Abstract

DNA electronic properties play a crucial role in understanding biological processes at the quantum level. The dna-electrons Python package provides accessible computational tools for calculating electron densities in nucleotide sequences, enabling researchers to analyze DNA structure and charge transfer with ease.

Summary

DNA electronic properties play a crucial role in understanding biological processes at the quantum level, yet accessible computational tools for calculating electron densities in nucleotide sequences remain limited. `dna-electrons` addresses this gap by providing a Python package that computes total electron counts for DNA sequences with customizable options for single- or double-stranded configurations, backbone inclusion, and phosphate charge considerations (Gupta 2025). The package integrates FASTA file parsing capabilities through Biopython (Chapman and Chang 2000) and offers both command-line and programmatic interfaces for high-throughput analysis. By providing predefined electron counts for nucleotide bases (A: 70, T: 66, G: 78, C: 58 electrons) and backbone components (238 electrons per base pair for sugar-phosphate backbone), `dna-electrons` enables researchers in computational biology, quantum chemistry, and molecular modeling to rapidly estimate electronic properties of DNA sequences without complex quantum mechanical calculations.

Statement of need

The electronic structure of DNA is fundamental to understanding various biological processes including charge transport (Senthilkumar et al. 2003), radiation damage (Sanche 2005), and DNA-protein interactions (Warshel et al. 2006). While sophisticated quantum chemistry methods exist for calculating electronic properties of small DNA fragments (Voityuk et al. 2001), these approaches are computationally prohibitive for analyzing larger genomic sequences or conducting high-throughput studies. Current bioinformatics tools focus primarily on sequence analysis, structural prediction, and functional annotation, but lack capabilities for estimating basic electronic properties that are increasingly relevant in fields such as DNA nanotechnology (Seeman 2003), molecular electronics

*ORCID: 0000-0003-3306-832X

(Porath et al. 2000), and radiation biology (Sanche 2005). Understanding electron distribution in DNA requires accurate models of charge transfer mechanisms, including both superexchange (Voityuk et al. 2001) and hopping mechanisms (Grozema et al. 2000). While several computational tools exist for DNA sequence analysis, there is a need for a lightweight, accessible tool specifically focused on electron counting in DNA bases.

The need for accessible electron count calculations has grown with advances in understanding DNA's role as a molecular conductor and the development of DNA-based nanostructures. Researchers studying electron transfer in DNA (Genereux and Barton 2010), designing DNA-based electronic devices, or investigating radiation-induced DNA damage require tools that can rapidly estimate electron densities across sequences of varying lengths and structural configurations. Existing quantum chemistry software packages, while accurate, are designed for detailed calculations on small systems and are not practical for genomic-scale analysis or integration into bioinformatics workflows.

`dna-electrons` fills this critical gap by democratizing access to fundamental electronic properties of DNA sequences, enabling researchers without extensive computational chemistry expertise to incorporate electronic considerations into their studies.

Methods

The computational methodology implemented in `dna-electrons` is based on the additive principle of electron counting, where the total electron count of a DNA sequence is calculated as the sum of contributions from individual components: nucleotide bases, sugar-phosphate backbone, and phosphate charges when applicable.

The core algorithm employs predefined electron counts for each DNA base derived from their molecular formulas: adenine (C5H5N5, 70 electrons), thymine (C5H6N2O2, 66 electrons), guanine (C5H5N5O, 78 electrons), and cytosine (C4H5N3O, 58 electrons). For backbone calculations, the package accounts for the 2'-deoxyribose sugar (70 electrons) and phosphate groups (49 electrons each), totaling 238 electrons per base pair for double-stranded DNA without considering ionic charges.

The software architecture follows modular design principles with separate components for sequence validation, electron counting, FASTA file processing, and result presentation. Sequence validation ensures input contains only valid DNA bases (A, T, G, C) and provides detailed error reporting for invalid characters. The FASTA processing module leverages Biopython (Chapman and Chang 2000) for robust parsing of single and multi-sequence files, enabling batch processing of genomic data.

Command-line functionality is implemented using Python's argparse module, providing intuitive options for specifying input sources (direct sequence or FASTA file), structural parameters (single- or double-stranded), and component inclusion toggles. Output formats include human-readable text, CSV for data analysis integration, and JSON for programmatic consumption.

Results

`dna-electrons` provides comprehensive electron counting capabilities through both command-line and programmatic interfaces. The package processes DNA sequences ranging from short

oligonucleotides to complete genomic sequences, with linear computational scaling that enables analysis of sequences containing millions of bases within seconds.

Key features include:

1. **Flexible Input Processing**: Supports direct sequence input via command-line arguments or FASTA file processing for batch analysis of multiple sequences with automatic sequence ID preservation.
2. **Configurable Structural Parameters**: Users can specify single- or double-stranded DNA configurations, with automatic complement generation for double-stranded calculations.
3. **Component-wise Analysis**: Provides detailed breakdowns showing contributions from nucleotide bases, sugar-phosphate backbone, and phosphate charges, enabling researchers to understand the relative importance of different structural components.
4. **High-throughput Compatibility**: FASTA processing capabilities enable analysis of entire genomes or large sequence datasets with results exportable in CSV format for integration with statistical analysis pipelines.
5. **Validation and Error Handling**: Comprehensive input validation ensures data integrity with clear error messages for sequences containing non-standard bases.

For example, analyzing the sequence "ATGCAT" as double-stranded DNA with full backbone and charge inclusion yields: 816 electrons from bases, 1,428 from backbone components, and charge adjustments, providing researchers with quantitative data for downstream modeling applications.

The package has been validated against manual calculations for diverse sequence types and demonstrates consistent accuracy across different DNA structural motifs including A-form, B-form, and Z-form configurations when appropriate electron counts are applied.

Availability

The `dna-electrons` package is freely available through the Python Package Index (PyPI) at https://pypi.org/project/dna-electrons/ for easy installation via pip. Complete source code, documentation, and examples are maintained in the GitHub repository at https://github.com/yashmgupta/dna-electrons under the MIT License, ensuring reproducibility and community collaboration. The repository includes comprehensive installation instructions, usage examples, and API documentation to support both novice and advanced users.

Acknowledgements

The author acknowledges the computational resources and support provided by the Department of Biology, Faculty of Science, Naresuan University, Thailand. Special thanks to the open-source Biopython project for providing robust biological sequence processing capabilities that enhance this package's functionality.

References

Chapman, Brad, and Jeff Chang. 2000. "Biopython: Python Tools for Computational Biology." *ACM SIGBIO Newsletter* 20 (2): 15–19. https://doi.org/10.1145/360262.360268.

Genereux, Joseph C, and Jacqueline K Barton. 2010. "Mechanisms for DNA Charge Transport." *Chemical Reviews* 110 (3): 1642–62. https://doi.org/10.1021/cr900228f.

Grozema, Ferdinand C, Yuri A Berlin, and Laurens DA Siebbeles. 2000. "Mechanism of Charge Transport Along DNA: Molecular Wire Behavior Versus Hopping." *Journal of the American Chemical Society* 122 (44): 10903–9. https://doi.org/10.1021/ja001497f.

Gupta, Yash Munnalal. 2025. *Dna-Electrons: A Python Package for Calculating Total Electron Counts in DNA Sequences*. https://pypi.org/project/dna-electrons/.

Porath, Danny, Alexey Bezryadin, Sense De Vries, and Cees Dekker. 2000. "Direct Measurement of Electrical Transport Through DNA Molecules." *Nature* 403 (6770): 635–38. https://doi.org/10.1038/35001029.

Sanche, Leon. 2005. "Low Energy Electron-Driven Damage in Biomolecules." *The European Physical Journal D-Atomic, Molecular, Optical and Plasma Physics* 35 (2): 367–90. https://doi.org/10.1140/epjd/e2005-00206-6.

Seeman, Nadrian C. 2003. "DNA in a Material World." *Nature* 421 (6921): 427–31. https://doi.org/10.1038/nature01406.

Senthilkumar, K, FC Grozema, FM Bickelhaupt, and LDA Siebbeles. 2003. "Charge Transport in Columnar Stacked Triphenylenes: Effects of Conformational Fluctuations on Charge Transfer Integrals and Site Energies." *The Journal of Chemical Physics* 119 (18): 9809–17. https://doi.org/10.1063/1.1615476.

Voityuk, Alexander A, Joshua Jortner, Mordechai Bixon, and Notker Rösch. 2001. "Electronic Coupling Between Watson-Crick Pairs for Hole Transfer and Transport in Desoxyribonucleic Acid." *Journal of Chemical Physics* 114 (13): 5614–20. https://doi.org/10.1063/1.1352035.

Warshel, Arieh, Pankaz K Sharma, Mitsunori Kato, Yun Xiang, Hanbin Liu, and Mats HM Olsson. 2006. "Electrostatic Basis for Enzyme Catalysis." *Chemical Reviews* 106 (8): 3210–35. https://doi.org/10.1021/cr0503106.