Supplemental Materials Guide

Project: Provenance-aware observations of mitochondrial organization

David A Stumpf, MD, PhD

Overview

The Northwestern University Libraries ARCH repository contains the supplemental materials accompanying this manuscript (Stumpf, David A, 2026). These materials provide the evidence supporting the reported biological observations through curated analytical reports, the underlying provenance-aware knowledge graph, and the associated computational environment. Together they enable independent examination of the analyses, reproduction of the principal results, and direct exploration of the biological relationships from which the observations were derived. The archived resources also provide a scientific and computational foundation for additional analyses by other investigators using the newly developed provenance-aware datasets.

The supplemental materials support several levels of engagement:

1. Review the manuscript and examine the Excel reports.
2. Reproduce published analyses using the supplied Cypher queries.
3. Reproduce the complete Neo4j ecosystem.
4. Extend the knowledge graph for new scientific investigations.

Contents

In addition to this document, the collection includes four other works:

- Supplemental Excel reports (S1–S21): DOI: https://doi.org/10.21985/n2-mpy4-mg87
- Java code: DOI: https://doi.org/10.21985/n2-q17c-bg70
- Neo4j database dump (wrs Build 9): DOI: https://doi.org/10.21985/n2-4h59-9577
- Curated reference files: DOI: https://doi.org/10.21985/n2-kg6k-c189

Neo4j Knowledge Graph

The Neo4j database represents the provenance-aware ecosystem described in the manuscript. It contains the nodes, relationships, properties, and indexes required to reproduce the reported analyses. The database dump can be restored into a compatible Neo4j installation.

Reproducibility

Explore the Northwestern University ARCH (short name: ARCH) server collection for the project to familiarize yourself with the content.

All principal observations reported in the manuscript were generated directly from the Neo4j knowledge graph using Cypher queries and supporting UDFs included in this archive. Where appropriate, output reports are provided to facilitate independent verification.

Version Information

Knowledge graph: WRS Build 9
Repository: Northwestern University ARCH
Resource type: Dataset
Publisher: Northwestern University

Licenses:

1. Text documents: Open Data Commons Attribution License (ODC-By)
2. Software: Apache License v. 2.0; January 2004: http://www.apache.org/licenses/

Contact

David A. Stumpf, MD, PhD
Professor Emeritus
Email: d-stumpf@northwestern.edu with subject line: Armosis Research Group

Please refer to the corresponding manuscript for scientific background and interpretation.

Supplemental Excel Reports

The Excel reports are curated scientific resources rather than simple program output. Each report contains a description of its purpose, the Cypher command used to generate the results, the execution time, and supporting implementation details that permit independent verification or regeneration of the analysis.

Supplemental reports are downloaded from the Northwestern Library ARCH server and unzipped into the folder supplemental material. From there they can be opened in Excel. There are two types of report. The most common renders a Cypher query result directly into Excel. This method also populated the worksheets with useful information.

1) Description of the report (see table summary)
2) A command that can be directly pasted into the Neo4j Browser to run the report
3) A Cypher query to run in the browser and output the data in the browser window.
4) The run time of the report
5) Messages about the report

supplement	class_name	description
S1	descriptive_statistics	descriptive statistics about the knowledge graph
S2	functional_unit_codons_variants	protein and peptide variants in start codons
S3	protein_bountary_motifs	protein_boundary_motifs
S4	protein_gene_codon_lists	protein-gene codon list by position with variant frequencies, including nonsynonymous amino acids
S5	protein_gene_codon_to_aa	protein-gene variants and their characterization and classification
S6	clusters1_protein_genes	protein-gene clusters.
S7	instantiated_pgids	instantiation patterns of protein-gene variant identifiers (PGIDs)
S8	gene_exception_fingerprint	protein-gene amino acid exception patterns: asymmetric nonsynonymous substitutions
S9	asymmetric_non_synonymous_codon_chi_q	protein amino acid substitutions at different positions compared using chi-square, returning P-values
S10	protein_gene_associations	preferential association of specific protein-gene variant pairings within individual GenBank sequences
S11	ETC_complex_gene_costraint	Electron transport chain assembly observations.
S12	comp_human_protein_gene	cross-species protein-gene boundary motif homology
S13	wrs_igp_report	Woodstock Reference Sequence (WRS) by position with HHI, guideposts, diversity and homologies
S14	variant_igp_guideposts	Genbank sequence boundary motif variants by position
S15	indels	insertions and deletions within the Woodstock Reference Sequence
S16	unique_mt_sequencces	Most full GenBank mitochondrial sequences are unique within the dataset.
S17	de_novo_seq_analytics	individual sequence example showing instantiated protein-gene options and rare variants
S18	armosis_metrics	Armosis metrics for codon to protein assembly. Must be run from Maven with desired settings.
S19	armosis_metrics_saturation	Saturation curves with different pruning of the long tail.
S20	armosis_metrics_saturation_iterator	Iterative saturation curves and statistics describing variation between runs. Must be run from Maven, not Neo4j Browser.

The Provenance-aware Neo4j Ecosystem

Method 1: Restore the archived knowledge graph (recommended)

The easiest path to creating the knowledge graph is to download the dump file and use the Neo4j Browser to load it. The Neo4j ecosystem needs further setup.

Standing up the Neo4j ecosystem requires several steps:

1. Download Neo4j Enterprise Edition from the Neo4j website and install.
2. Open the Neo4j Desktop
3. Create a new dbms with a name and password of your choosing.
    a. Open the dbms folder which has subfolder you will be working with
    b. Open these folders from the Neo4j Desktop (epiclesis to right of Open button)
        i. Plugin
        ii. Configuration (conf)
        iii. DBMS
4. Add the APOC and Graph Data Science plugins from the Neo4j Desktop Verify by finding their jar files in the plugin folder.
5. Start the dbms in Descktop. Success is a start without warning.
6. Open the Neo4 Browser using one of these two methods:
    a. From the Neo4j Desktop, click the Open button.
    b. From your internet browser, navigate to http://localhost:7474/browser/
7. In the Neo4j Browser verify that the apoc functions are operation using this Cypher:
   SHOW FUNCTIONS yield name where name contains "apoc" RETURN name.
   Repeat using “gds” to verify Graph Data Science is operational.
8. If validated, stop the dbms
9. Download the Neo4j_Ecosystem.zip file from the Northwester Library ARCH server.
    a. Unzip the file and it will create a folder with the setup files.
    b. Add the apoc.conf file to the conf folder.
    c. Added the wrs JAR the plugin folder
    d. Open the file neo4j.conf in a text editor (e.g., Notepad++) and at the bottom add/paste the lines from the file neo4j.config.file.added_lines.txt. Save the changes.
10. Restart the dbms in Neo4j Desktop
11. In the Neo4j Browser verify that the wrs functions are operation using this Cypher:
    SHOW FUNCTIONS yield name where name contains"mtwrs" RETURN name
12. Restart the dbms. If it starts without warning, you have the desired WRS ecosystem!
13. Download the Neo4j dump file from ARCH and place it in the dbms folder. Verify that it is visible the Neo4j Descktop towards the bottom.
14. In Descktop click the epiphysis one the dump file row and click: import dump into existing DBMS.
15. Select your dbms and enter the database name (wrs) and start the process of creating the database. In several minutes you should see the wrs knowledge graph in the Neo4j Browser.
16. In the Neo4j Browser, run this Cypher query:
    MATCH p=()-[r:code_for]->() RETURN p LIMIT 25
    you should see displayed two node types and their relationship

You Neo4j Ecosystem is now operational.

Method 2: Rebuild from curated reference files

Alternatively, the database can be built from Java code and the reference files, loaded into folders as just described. From Maven (or similar tool), run the UDF java class Load_everything_from_files in the package mtwrs.loa. The database must be created and empty. The reference files will be transferred to the Neo4j import folder and processed from there. A tracker file will collect the workflow steps, and they will also appear as output in Maven. The load time will vary with the hardware setup; it was ~ 1 hour in the project set up.

Method 3: Regenerate the complete reference dataset from source data

The code can generate the reference files from the source data. This is a computationally intense workflow for some of the reference files which is why the reference files were produced once for expedite re-use as described in Method 2. The project code generates the incremental reference file developed for this project. Other reference files were produced in previous projects (David A. Stumpf, 2025; David A Stumpf, 2025).

Neo4j Documentation

Neo4j has a rich collection of documentation if users require further background information. See: https://neo4j.com/

Ecosystem Folders and Files

The ecosystem folder structure contains the materials needed to reproduce results. The code was developed in Maven with the root folder of mtvar. Packages are in folders roughly concerning different purposes of the included code.

```
+---mtwrs
    +---conn
    ¦       AuthInfo.java
    ¦       connTest.java
    ¦
    +---dmp
    ¦       dmp.java
    ¦       dmp_bases_at_pos.java
    ¦       dmp_block_similarity.java
    ¦       dmp_equals.java
    ¦       dmp_get_variants.java
    ¦       dmp_two_probes_variants.java
    ¦       do_not_use_dmp_to_base_pos.java
    ¦
    +---excelLib
    ¦       excelContainer.java
    ¦       excelFromFile.java
    ¦       excelRept.java
    ¦
    +---genlib
    ¦       create_mt_seq_csv_file.java
    ¦       current_date_time.java
    ¦       dedup_list.java
    ¦       equal_long_lists.java
    ¦       Genbank_phenotype.java
    ¦       id_for_node.java
    ¦       tracker.java
    ¦
    +---graph
    ¦       base10to2.java
    ¦       dedup_sort.java
    ¦       dedup_sort_var.java
    ¦       getHexDec.java
    ¦       ordpath.java
    ¦
    +---guideposts
    ¦       guidepost_metrics.java
    ¦       guidepost_positions.java
    ¦
    +---load
    ¦       conept_ontology.java
    ¦       depreciated_load_protein_gene_seqs.java
    ¦       Load_everything_from_files.java
    ¦       load_guideposts.java
```

```
        ¦   load_mtDNA_amino_acid_codons.java
        ¦   load_ncbi_functional_units.java
        ¦   load_palindrome.java
        ¦   load_seq.java
        ¦   load_wrs_igp_bases.java
        ¦   reload_comparison_species.java
        ¦
        +---neo4jlib
        ¦   file_lib.java
        ¦   neo4j_info.java
        ¦   neo4j_qry.java
        ¦   Provenance.java
        ¦   ReportDescription.java
        ¦   transaction_query.java
        ¦
        +---palindrome
        ¦   palindrome_coverage.java
        ¦
        +---reports
        ¦   armosis_metrics.java
        ¦   armosis_metrics_saturation.java
        ¦   armosis_metrics_saturation_iterator.java
        ¦   asymmetric_non_synonymous_codon_chi_q.java
        ¦   clusters.java
        ¦   clusters1_protein_genes.java
        ¦   codon_exception_stats.java
        ¦   comp_human_protein_gene.java
        ¦   curated_files.java
        ¦   descriptive_statistics.java
        ¦   de_novo_seq_analytics.java
        ¦   ETC_assembly.java
        ¦   ETC_complex_gene_costraint.java
        ¦   ETF_Complex_associations.java
        ¦   functional_unit_codons_variants.java
        ¦   functional_unit_codon_sample_detail.java
        ¦   function_unit_descriptive_facts.java
        ¦   gene_exception_fingerprint.java
        ¦   gene_exception_types.java
        ¦   indels.java
        ¦   indels_from_igp_variants.java
        ¦   instantiated_pgids.java
        ¦   nucleotide_repeats.java
        ¦   protein_bountary_motifs.java
        ¦   protein_codon_options.java
        ¦   protein_gene_associations.java
        ¦   protein_gene_codon_lists.java
        ¦   protein_gene_codon_to_aa.java
        ¦   supplements_materials.java
        ¦   unique_mt_sequenvces.java
        ¦   variant_igp_guideposts.java
```

```
¦     wrs_igp_report.java
¦     wrs_rsrs_compare.java
¦     wrs_seq_analytics.java
¦
+---sorter
¦     variants.java
¦     variants_toList.java
¦
+---species
¦     create_mt_DNA_other_species_nodes.java
¦     taxomy_to_nodes.java
¦     Taxonomy.java
¦     TaxonomyFetcher.java
¦
+---templates
¦     udf.java
¦
+---workflow
¦     Workflow.java
¦
+---wrs
      boundary_codons.java
      codon_exception.java
      create_codon_protein_gene_rel.java
      create_comp_seq_igp_rel1.java
      create_comp_seq_protein_gene_ecosystem1.java
      create_mt_seq_igp_rel.java
      create_protein_gene_ecosystem.java
      gfgVar.java
      igp_components.java
      pos_to_functional_unit.java
      protein_codon_exceptions.java
      seq_var_using_ipg.java
```

References

Stumpf, David A., 2025. A Mitochondrial DNA Knowledge Graph [WWW Document]. URL https://doi.org/10.21985/n2-fx1m-2m11 (accessed 7.3.25).

Stumpf, David A, 2025. The Mitochondrial Palindrome Scaffold and Architectural Domains. bioRxiv 2025.11.11.687892. https://doi.org/10.1101/2025.11.11.687892

Stumpf, David A, 2026. Supplemental materials: Provenance-aware observations of mitochondrial organization [WWW Document]. Northwestern University Institutional ARCH Repository. URL https://arch.library.northwestern.edu/dashboard/collections/6395w7883?locale=en (accessed 7.21.26).