

# Package: panpathAudit (via r-universe)

September 19, 2026

**Type** Package

**Title** Audit Sequence Preservation in GFA Traversals

**Version** 0.99.2

**Description** Audits source FASTA sequences against named paths and coordinate-aware walks in GFA files. It reports exact recovery, divergence coordinates, digests, and optional base-level statistics. Graph topology is not validated.

**License** MIT + file LICENSE

**Encoding** UTF-8

**Language** en-US

**Depends** R (>= 4.5.0)

**Imports** methods, Rcpp, S4Vectors

**LinkingTo** Rcpp

**Suggests** BiocCheck, BiocStyle, knitr, rmarkdown, testthat (>= 3.0.0)

**VignetteBuilder** knitr

**SystemRequirements** C++20, zlib

**biocViews** Alignment, DataImport, GenomeAssembly, Genetics, Sequencing

**URL** <https://github.com/gkanogiannis/panpathAudit>

**BugReports** <https://github.com/gkanogiannis/panpathAudit/issues>

**NeedsCompilation** yes

**RoxygenNote** 8.1.0

**Config/testthat/edition** 3

**Config/pak/sysreqs** zlib1g-dev

**Repository** <https://biocstaging.r-universe.dev>

**Date/Publication** 2026-09-19 12:40:17 UTC

**RemoteUrl** <https://github.com/BiocStaging/panpathAudit>

**RemoteRef** HEAD

**RemoteSha** 03e80adec188ca08919043877f22bcfb61dd6deb

Contents

|                                    |          |
|------------------------------------|----------|
| panpathAudit-package . . . . .     | 2        |
| auditGFA . . . . .                 | 2        |
| PanPathAuditResult-class . . . . . | 3        |
| <b>Index</b>                       | <b>5</b> |

---

|                      |                                                      |
|----------------------|------------------------------------------------------|
| panpathAudit-package | <i>Audit sequence preservation in GFA traversals</i> |
|----------------------|------------------------------------------------------|

---

Description

Audits FASTA sequences against named GFA paths and walks without claiming to validate graph topology.

---

|          |                                                     |
|----------|-----------------------------------------------------|
| auditGFA | <i>Audit FASTA sequences against GFA traversals</i> |
|----------|-----------------------------------------------------|

---

Description

Compares source sequences with named GFA P paths and coordinate-aware W walks. Graph topology is not validated.

Usage

```
auditGFA(  
  fasta,  
  gfa,  
  mapping = NULL,  
  statistics = c("basic", "comprehensive"),  
  threads = 1L,  
  memoryMiB = 1024L,  
  alignmentMaxCells = 10000000  
)
```

Arguments

- |                   |                                                          |
|-------------------|----------------------------------------------------------|
| fasta             | Character vector of FASTA paths.                         |
| gfa               | One GFA path.                                            |
| mapping           | Optional data frame with one row per FASTA. See Details. |
| statistics        | Either "basic" or "comprehensive".                       |
| threads           | Positive worker count.                                   |
| memoryMiB         | Positive tracked-memory limit in MiB.                    |
| alignmentMaxCells | Positive comprehensive-alignment limit.                  |

**Details**

Gzip input is detected from file content. GFA P records must use blunt overlaps (\* or 0M). W identifiers are formed as sample#haplotype#sequence; their ranges are zero-based and half-open.

Mapping modes are exact, pansn, and prefix. PanSN rows require sample and numeric haplotype; prefix rows require prefix.

Argument, input, and operational failures raise conditions with classes `panpathAudit_argument_error`, `panpathAudit_input_error`, and `panpathAudit_operational_error`. Each inherits from `panpathAudit_error` and carries a diagnostics DataFrame. Diagnostic positions are one-based.

**Value**

A [PanPathAuditResult](#).

**Examples**

```
fasta <- system.file("extdata", "example.fa", package = "panpathAudit")
gfa <- system.file("extdata", "example.gfa", package = "panpathAudit")
result <- auditGFA(fasta, gfa)
auditOutcomes(result)
```

---

PanPathAuditResult-class

*Audit result class*

---

**Description**

Stores traversal outcomes, summary counts, optional comprehensive statistics, provenance, and memory telemetry.

**Usage**

```
auditOutcomes(x)
auditSummary(x)
auditStatistics(x)
auditProvenance(x)
auditTelemetry(x)
```

**Arguments**

x                      A `PanPathAuditResult`.

**Value**

`auditOutcomes()` returns a DataFrame. Other accessors return SimpleList objects. `as.data.frame()` returns the outcomes as a base data frame.

**Outcome columns**

identifier Mapped path or walk identifier.  
 source\_index One-based FASTA index.  
 status IDENTICAL, DIVERGENT, MISSING\_PATH, or MISSING\_SOURCE.  
 record\_type P or W.  
 source\_start\_0based, source\_end\_0based\_exclusive Walk source range.  
 source\_length, graph\_length, addressed\_source\_bases Sequence and addressed-range sizes.  
 divergence\_kind BASE, END, or RANGE\_LENGTH.  
 source\_position\_1based, graph\_position\_1based First divergent base position.  
 segment, orientation Segment and traversal strand at the divergence.  
 traversal\_position\_1based, segment\_position\_1based Divergence coordinates within the traversal and segment.  
 range\_length, walk\_length Expected and reconstructed sizes for a walk range mismatch.  
 source\_sha256, addressed\_source\_sha256, graph\_sha256 SHA-256 digests.  
 source\_blake3, addressed\_source\_blake3, graph\_blake3 BLAKE3 digests.  
 source\_ambiguous\_bases, graph\_ambiguous\_bases, ambiguous\_columns IUPAC ambiguity counts.  
 length\_delta Graph length minus addressed source length.  
 matched, substituted, source\_only, graph\_only Aligned base counts.  
 not\_embedded, missing\_path\_bases, missing\_source\_bases Unrepresented base counts.  
 unaligned\_divergent\_source, unaligned\_divergent\_graph Divergent bases not aligned because of a resource limit.  
 edit\_distance Edit distance when alignment completed.  
 alignment\_status not\_applicable, not\_required, complete, limit\_exceeded, or memory\_limit\_exceeded.  
 Columns that do not apply contain NA.

**Accessor structures**

auditSummary() contains counts, percentages, and total. auditStatistics() is empty in basic mode; otherwise it contains global, by\_source, and by\_record\_type aggregates. Each aggregate reports sequence and base totals, ambiguity counts, edit and alignment totals, base-category counts, and status counts.  
 auditProvenance() records package and reference versions, normalized input paths, mapping and resource options, compression, GFA version and record counts, overlap interpretation, and the false topology\_validated flag.  
 auditTelemetry() contains limit\_bytes, peak\_tracked\_bytes, oversized\_contigs, and threads.

**Examples**

```

fasta <- system.file("extdata", "example.fa", package = "panpathAudit")
gfa <- system.file("extdata", "example.gfa", package = "panpathAudit")
result <- auditGFA(fasta, gfa)
summary(result)

```

# Index

`as.data.frame`, `PanPathAuditResult`-method  
    (`PanPathAuditResult`-class), [3](#)  
`auditGFA`, [2](#)  
`auditOutcomes`  
    (`PanPathAuditResult`-class), [3](#)  
`auditProvenance`  
    (`PanPathAuditResult`-class), [3](#)  
`auditStatistics`  
    (`PanPathAuditResult`-class), [3](#)  
`auditSummary`  
    (`PanPathAuditResult`-class), [3](#)  
`auditTelemetry`  
    (`PanPathAuditResult`-class), [3](#)  
  
`panpathAudit` (`panpathAudit`-package), [2](#)  
`panpathAudit`-package, [2](#)  
`PanPathAuditResult`, [3](#)  
`PanPathAuditResult`-class, [3](#)  
  
`show`, `PanPathAuditResult`-method  
    (`PanPathAuditResult`-class), [3](#)  
`summary`, `PanPathAuditResult`-method  
    (`PanPathAuditResult`-class), [3](#)