Salmon Index - Comprehensive Reference Guide

Overview

Salmon is a fast, alignment-free tool for quantifying transcript abundance from RNA-seq data. It uses a k-mer-based quasi-mapping approach that is 10-30x faster than traditional alignment-based methods while maintaining comparable accuracy.

Website: https://combine-lab.github.io/salmon/
Publication: Patro et al. Nature Methods 2017

Applicable to: RNA-seq, single-cell RNA-seq, metatranscriptomics

---

What is a Salmon Index?

The Salmon index is a pre-computed data structure that enables rapid transcript quantification:

Core Components

1. **K-mer hash table** - Maps short sequences (k-mers) to transcript locations
2. **Suffix array** - Enables efficient k-mer lookups
3. **Transcript metadata** - Lengths, names, gene associations
4. **Decoy sequences** (optional but recommended) - Genome sequences to catch spurious mappings

Index vs Traditional Alignment

Feature	Salmon (k-mer index)	STAR/HISAT2 (genome index)
Target	Transcriptome	Genome
Method	K-mer matching	Seed-and-extend alignment
Speed	Very fast (2-5 min/sample)	Moderate (10-30 min/sample)
Output	Transcript counts	BAM alignment file
Index size	5-8GB (human)	25-30GB (human)
Build time	30-40 minutes	1-2 hours
Memory	16-32GB	30-50GB
Splice-aware	N/A (uses transcripts)	Yes

Gentrome Strategy

What is a Gentrome?

Gentrome = Transcriptome + Genome

A gentrome is a combined FASTA file containing:

1. **Transcripts** (targets for quantification)
2. **Full genome sequences** (decoys to catch spurious mappings)

Why Use Decoys?

Problem without decoys:

- Reads from intergenic/intronic regions may incorrectly map to similar transcripts
- Pseudogene reads inflate parent gene counts
- Unannotated transcripts cause false positives

Solution with decoys:

```
Read sequence
  ↓
Try mapping to transcripts
  ↓
Good match? → Quantify transcript
  ↓
Poor match? → Try decoy sequences (genome)
  ↓
Better match to decoy? → Discard read (not from transcript)
```

Accuracy Improvement

- **Without decoys**: ~5-15% false positive rate
- **With decoys**: ~1-3% false positive rate
- **Improvement**: 10-15% more accurate quantification

When Decoys Matter Most

Scenario	Impact	Example
Gene families	High	HOX genes, olfactory receptors
Pseudogenes	High	GAPDH pseudogenes → GAPDH gene
gDNA contamination	High	Intronic reads → nearby exons
Incomplete annotation	Medium	Novel transcripts
Highly expressed unique genes	Low	Already accurate

Building a Salmon Index

Step-by-Step Process

Step 1: Extract Decoy List Extract chromosome/scaffold names from genome FASTA:

```
# Method 1: Simple extraction
grep "^>" genome.fa | cut -d " " -f1 | sed 's/>//' > decoys.txt

# Method 2: For complex headers
grep "^>" genome.fa | awk '{print $1}' | sed 's/>//' > decoys.txt
```

Command breakdown (Method 1):

```
grep "^>" genome.fa           # Find all header lines (start with >)
  | cut -d " " -f1            # Take first field before space
  | sed 's/>//'               # Remove the leading ">" character
  > decoys.txt                # Write to output file
```

Example input (FASTA header):

```
>1 dna:chromosome chromosome:GRCh38:1:1:248956422:1 REF
>2 dna:chromosome chromosome:GRCh38:2:1:242193529:1 REF
>MT dna:chromosome chromosome:GRCh38:MT:1:16569:1 REF
```

Example output (decoys.txt):

```
1
2
3
...
X
Y
MT
```

Critical:

- Names must exactly match the first word after > in FASTA
- No extra whitespace or blank lines
- One chromosome name per line
- Case-sensitive matching

Step 2: Extract Transcriptome Extract transcript sequences using gene annotation:

```
# Using gffread (recommended)
gffread annotation.gtf -g genome.fa -w transcriptome.fa

# Using AGAT (alternative)
```

```
agat_sp_extract_sequences.pl -g annotation.gtf -f genome.fa -t exon --merge -o transcriptome

# Using bedtools (if you have BED format)
bedtools getfasta -fi genome.fa -bed annotation.bed -fo transcriptome.fa -name -split
```

gffread command breakdown:

```
gffread annotation.gtf        # Input GTF annotation file
  -g genome.fa                # Genome FASTA file
  -w transcriptome.fa         # Write transcript sequences to output
```

What it does:

1. Reads GTF to find exon coordinates for each transcript
2. Extracts corresponding sequences from genome FASTA
3. Concatenates exons to form full transcript sequences
4. Outputs one FASTA entry per transcript

Output: FASTA file with one sequence per transcript

Header format examples:

```
>ENST00000000001.1 ENSG00000000001.1 GENE_NAME
ATGCGTACG...
>ENST00000000002.1 ENSG00000000001.1 GENE_NAME
ATGCGTACG...
```

Common issues:

- **Chromosome mismatch**: GTF has "chr1" but FASTA has "1" (or vice versa)
- **Compressed files**: gffread can't read .gz files directly - decompress first
- **GTF vs GFF3**: Use GTF format, not GFF3 (different attribute syntax)

Step 3: Create Gentrome Concatenate transcripts and genome:

```
# ORDER IS CRITICAL: transcripts first, genome second
cat transcriptome.fa genome.fa > gentrome.fa
```

Command breakdown:

```
cat transcriptome.fa genome.fa    # Concatenate two FASTA files
  > gentrome.fa                   # Write combined output
```

Why order matters:

1. Salmon reads the gentrome file sequentially
2. The decoy list tells Salmon: "sequences with these names are decoys"
3. Salmon uses position + names to distinguish:
   - **Transcripts** (before genome sequences) = quantification targets

- **Genome sequences** (matching decoy names) = decoys to catch spurious hits

4. If you reverse the order (genome first), Salmon will treat genome as transcripts!

Visual representation:

```
gentrome.fa structure:

 >ENST0001 (transcript 1)      ← Quantification targets
 ATGCGTACG...
 >ENST0002 (transcript 2)
 ATGCGTACG...
 ... (all transcripts)

 >1 (chromosome 1)             ← Decoys (names in decoys.txt)
 ATGCGTACG...
 >2 (chromosome 2)
 ATGCGTACG...
 ... (all chromosomes)
```

Size check:

```
# Gentrome should be: transcriptome + genome size
# Human example: ~200MB (transcripts) + 3GB (genome) = ~3.2GB
ls -lh gentrome.fa

# Verify both files were included
grep -c "^>" transcriptome.fa  # Count transcripts
grep -c "^>" genome.fa         # Count chromosomes
grep -c "^>" gentrome.fa       # Should equal sum of above
```

Step 4: Build Index

```
salmon index \
    --transcripts gentrome.fa \
    --decoys decoys.txt \
    --index salmon_index \
    --kmerLen 31 \
    --threads 8
```

Parameters explained:

Parameter	Description	Default	Notes
`--transcripts`	Gentrome FASTA	Required	Path to gentrome.fa (or transcriptome.fa only)

Parameter	Description	Default	Notes
`--decoys`	Decoy names file	None	Highly recommended for accuracy
`--index`	Output directory	Required	Will be created; use descriptive name
`--kmerLen`	K-mer size	31	31 for 75-200bp reads; see k-mer guide
`--threads`	Parallel threads	1	Use 8-16 for faster build
`--sparse`	Sparse index	Off	Reduces memory ~50%, slightly slower quant
`--keepDuplicates`	Keep duplicate txs	Off	Usually leave off
`--gencode`	GENCODE headers	Off	Use if annotation is GENCODE format

Full command breakdown:

```
salmon index \                      # Salmon indexing command
    --transcripts gentrome.fa \     # Input: combined transcriptome + genome
    --decoys decoys.txt \           # File with chromosome names (marks decoys)
    --index salmon_index \          # Output directory name
    --kmerLen 31 \                  # K-mer length (31 for standard reads)
    --threads 8                     # Use 8 CPU cores for parallel building
```

What happens during indexing:

1. Salmon reads gentrome.fa sequentially
2. Builds k-mer hash table from all sequences
3. Marks sequences in decoys.txt as "decoy sequences"
4. Creates auxiliary data structures (suffix array, metadata)
5. Writes all structures to salmon_index/ directory
6. Logs build parameters to versionInfo.json

Time and resources:

- Human genome: ~30-40 minutes on 8 cores
- Memory: 16-32GB RAM
- Output size: 5-8GB

K-mer Length Selection

K-mer length affects sensitivity and specificity:

Default: k=31 (Recommended)

Best for:

- 75-200bp reads (most RNA-seq)
- Illumina short-read sequencing
- Standard sensitivity/specificity

Characteristics:

- Balanced performance
- Recommended by Salmon developers
- Works well for most applications

Short k-mers: k=21

Best for:

- Very short reads (<75bp)
- Degraded RNA samples
- Small RNA sequencing

Characteristics:

- More sensitive (finds more matches)
- Less specific (more false positives)
- Larger index size
- Slower quantification

Trade-offs:

- Better for poor quality data
- May overestimate low-abundance transcripts

Long k-mers: k=41 or k=51

Best for:

- Long reads (PacBio, Nanopore)
- Very high quality Illumina (>200bp)
- Highly specific applications

Characteristics:

- More specific (fewer false positives)
- Less sensitive (may miss divergent sequences)
- Smaller index size
- Faster quantification

Trade-offs:

- Better for high-quality data
- May miss divergent isoforms

Selection Guide

Read Length	Recommended k-mer
<50bp	k=19 or k=21
50-75bp	k=25 or k=27
75-150bp	k=31 (default)
150-250bp	k=31 or k=41
>250bp	k=41 or k=51

Validation: After building, check mapping rates during quantification. If <70%, consider:

- k=31 → k=27 (more sensitive)
- k=31 → k=41 (if quality is high)

Index Output Structure

Directory Contents

After indexing, `salmon_index/` contains:

```
salmon_index/
   complete_ref_lens.bin       # Reference sequence lengths
   ctable.bin                  # Color table for equivalence classes
   ctg_offsets.bin            # Contig offset information
   duplicate_clusters.tsv      # Groups of identical transcripts
   hash.bin                    # K-mer hash table (LARGEST FILE)
   info.json                   # Index metadata
   mphf.bin                    # Minimal perfect hash function
   pos.bin                     # K-mer position information
   pre_indexing.log           # Build log
   rank.bin                    # Rank information
   refAccumLengths.bin        # Cumulative reference lengths
   ref_indexing.log           # Indexing log
   reflengths.bin             # Individual reference lengths
   refseq.bin                 # Reference sequences
   seq.bin                     # Sequence data
   txpInfo.bin                # Transcript information
   versionInfo.json           # Salmon version
   decoys.txt                 # Copy of decoy list
```

Important Files

hash.bin (largest, 3-5GB):

- K-mer hash table

- Core data structure for mapping
- Size depends on k-mer length and transcriptome complexity

txpInfo.bin:

- Transcript names, lengths, genes
- Used for output formatting

duplicate_clusters.tsv (human-readable):

- Groups of identical transcript sequences
- Common for isoforms with same CDS
- Salmon distributes reads across cluster

Example:

```
RetainedTxp DuplicateTxp
ENST00000001.1  ENST00000002.1,ENST00000003.1
```

versionInfo.json:

- Salmon version used for indexing
- Index format version
- Compatibility information

Index Size and Performance

Size Expectations

Organism	Transcripts	Index Size	Build Time (8 cores)
Human	~240,000	5-8 GB	30-40 min
Mouse	~150,000	4-6 GB	25-35 min
Drosophila	~35,000	1-2 GB	10-15 min
Yeast	~6,000	200-400 MB	2-5 min
E. coli	~4,000	100-200 MB	1-2 min

Factors affecting size:

- Number of transcripts
- K-mer length (longer k = smaller index)
- Genome size (if using decoys)
- Sparse indexing (reduces size 50%)

Build Performance

Resource requirements:

Resource	Typical	Minimum	Maximum
Memory	16-24 GB	8 GB	32+ GB
Threads	8	1	16+
Time	30-40 min	10 min (small)	2 hours (very large)
Disk	10-15 GB temp	5 GB	30+ GB

Optimization tips:

- Use local SSD storage (faster I/O)
- Allocate 8-16 threads (diminishing returns beyond)
- Close other applications (memory intensive)
- Use `--sparse` for memory-constrained systems

Sparse Indexing

Sparse indexing reduces memory usage at cost of slightly slower quantification:

```
salmon index \
    --transcripts gentrome.fa \
    --decoys decoys.txt \
    --index salmon_index_sparse \
    --sparse \
    --threads 8
```

Characteristics:

- ~50% smaller index size
- ~50% less memory during indexing
- ~10-20% slower quantification
- Same accuracy

When to use:

- Limited RAM (<16GB)
- Very large transcriptomes
- Disk space constraints
- Quantification speed not critical

Index Compatibility and Reusability

When to Reuse an Index

Can reuse if:

- Same genome assembly version
- Same gene annotation version
- Same Salmon version (or compatible)
- Same k-mer length
- Same decoy strategy (with or without)

Must rebuild if:

- Different genome assembly (e.g., GRCh37 → GRCh38)
- Updated gene annotation with new transcripts
- Salmon major version change (e.g., 1.x → 2.x)
- Different k-mer length needed
- Switching decoy strategy

Version Compatibility

Salmon versions:

- Generally forward-compatible within major versions
- 1.0 → 1.9: Usually compatible
- 1.x → 2.x: Likely need rebuild
- Check release notes for breaking changes

Checking index version:

```
cat salmon_index/versionInfo.json
```

Output:

```
{
  "indexVersion": "2",
  "salmonVersion": "1.10.0",
  "hasDecoys": true,
  "kmerLen": 31
}
```

Transcriptome-Only vs Gentrome

Transcriptome-Only Index

Build:

```
salmon index \
    --transcripts transcriptome.fa \
    --index salmon_index_txome \
    --kmerLen 31
```

Pros:

- Faster to build (no genome sequences)
- Smaller index size
- Simpler workflow

Cons:

- 5-15% less accurate
- Prone to false positives
- Problematic for gene families

When to use:

- High-quality, well-annotated organisms
- Proof-of-concept analyses
- Storage/time severely limited

Gentrome Index (Recommended)

Build: (See main workflow above)

Pros:

- 10-15% more accurate
- Reduces false positives
- Better for gene families
- Handles gDNA contamination

Cons:

- Longer build time
- Larger index
- More complex workflow

When to use:

- All production analyses
- When accuracy is important
- Gene family analysis
- Samples with potential gDNA contamination

Troubleshooting

Out of Memory During Indexing

Symptoms:

- Process killed
- "Cannot allocate memory" error
- System becomes unresponsive

Solutions:

1. **Increase memory allocation**

   ```
   # Allocate 32GB
   salmon index --transcripts gentrome.fa --index idx --threads 8
   ```

2. **Use sparse indexing**

   ```
   salmon index --transcripts gentrome.fa --index idx --sparse
   ```

3. **Reduce threads**

   ```
   # Fewer threads = less parallel memory usage
   salmon index --transcripts gentrome.fa --index idx --threads 4
   ```

4. **Close other applications**

   - Free up system memory
   - Check: `free -h` or `top`

Transcript Extraction Failed

Symptoms:

- gffread crashes
- Empty transcriptome.fa
- Missing transcripts

Common causes:

1. **GTF/GFF format mismatch**

   ```
   # Check format
   head -n 5 annotation.gtf

   # GTF: tab-separated, attributes in column 9
   # GFF3: different attribute format
   ```

2. **Chromosome name mismatch**

   ```
   # Check FASTA headers
   grep "^>" genome.fa | head -n 5
   ```

```
# Check GTF chromosomes
cut -f1 annotation.gtf | sort -u

# Must match exactly!
```

3. **Compressed files**

```
# gffread cannot read .gz directly
gunzip annotation.gtf.gz
gunzip genome.fa.gz
```

4. **Malformed GTF**

```
# Validate GTF
gt gff3validator annotation.gtf

# Or use AGAT
agat_convert_sp_gxf2gxf.pl -g annotation.gtf -o annotation_fixed.gtf
```

Very Low Mapping Rates

Symptoms:

- <30% mapping rate during quantification
- Many reads unmapped
- Suspiciously low counts

Diagnosis:

1. **Check index was built successfully**

```
ls -lh salmon_index/
# Should have hash.bin, txpInfo.bin, etc.

cat salmon_index/versionInfo.json
# Check hasDecoys: true/false, kmerLen
```

2. **Verify species match**

```
# Check a few transcript names
grep "^>" transcriptome.fa | head -n 10

# Should match expected organism
```

3. **Check for corruption**

```
# Rebuild index and compare
md5sum salmon_index/hash.bin
```

Decoy Format Errors

Symptoms:

- "Decoy sequence not found in transcriptome"
- "Decoy names do not match"
- Index builds but quantification fails

Solutions:

1. **Check decoy file format**

```
# Should be plain text, one name per line, no headers
head -n 5 decoys.txt

# Should show:
1
2
3
```

2. **Verify names match FASTA**

```
# Extract FASTA names
grep "^>" genome.fa | cut -d " " -f1 | sed 's/>//' > genome_names.txt

# Compare with decoys
diff decoys.txt genome_names.txt
# Should be identical!
```

3. **Check for whitespace**

```
# Remove trailing whitespace
sed -i 's/[[:space:]]*$//' decoys.txt

# Remove blank lines
sed -i '/^$/d' decoys.txt
```

Duplicate Transcript IDs

Symptoms:

- Warning: "Duplicate transcript ID"
- Index builds but with warnings
- Quantification works but some transcripts missing

Understanding:

- Multiple transcripts with identical sequences
- Common for alternative transcript annotations
- Salmon handles this automatically

Check duplicates:

```
cat salmon_index/duplicate_clusters.tsv
```

Action:

- Usually safe to ignore
- Salmon distributes reads across duplicates
- Final counts are still accurate

Best Practices

For Production Use

Always:

- Use gentrome with decoys
- Keep index build logs
- Document Salmon version
- Use consistent k-mer length
- Test with small dataset first

Recommend:

- Use k=31 for standard RNA-seq
- Allocate 32GB RAM for human genome
- Use 8+ threads for speed
- Publish index to shared location

Avoid:

- Transcriptome-only for important analyses
- Mixing index versions
- Building on network storage (slow)
- Deleting build logs

For Different Applications

Standard RNA-seq:

```
salmon index \
    --transcripts gentrome.fa \
    --decoys decoys.txt \
    --index salmon_idx \
    --kmerLen 31 \
    --threads 8
```

Single-cell RNA-seq:

```
# Same as standard, but often used with alevin
salmon index \
    --transcripts gentrome.fa \
    --decoys decoys.txt \
    --index salmon_idx \
    --kmerLen 31 \
    --threads 8
```

Degraded RNA:

```
# Use shorter k-mer for sensitivity
salmon index \
    --transcripts gentrome.fa \
    --decoys decoys.txt \
    --index salmon_idx \
    --kmerLen 27 \
    --threads 8
```

Memory-constrained:

```
salmon index \
    --transcripts gentrome.fa \
    --decoys decoys.txt \
    --index salmon_idx \
    --sparse \
    --threads 4
```

Command Reference

```
# Basic gentrome index
salmon index -t gentrome.fa -d decoys.txt -i idx

# With all options
salmon index \
    --transcripts gentrome.fa \
    --decoys decoys.txt \
    --index salmon_index \
    --kmerLen 31 \
    --threads 8 \
    --gencode \
    --keepDuplicates

# Sparse index (low memory)
salmon index -t gentrome.fa -d decoys.txt -i idx --sparse

# Transcriptome-only (not recommended)
```

```
salmon index -t transcriptome.fa -i idx

# Check index info
cat salmon_index/versionInfo.json
cat salmon_index/duplicate_clusters.tsv

# Validate decoys
diff <(grep "^>" genome.fa | cut -d" " -f1 | sed 's/>//') decoys.txt
```

Related Documentation

- **Salmon Quantification**: `docs/salmon_quant.md`
- **tximport Integration**: `docs/tximport.md`
- **Differential Expression**: `docs/deseq2.md`
- **Salmon Official Docs**: https://salmon.readthedocs.io/

Document Version: 2.0
Last Updated: January 2026
Salmon Version: 1.10.0+
Applicable to: RNA-seq, scRNA-seq, metatranscriptomics