MoSelecta User Guide

v0.3.0

MoSelecta integrates CDS-containing annotation models from multiple prediction sources into a deterministic, auditable final annotation. Genome sequence, protein alignments and transcript evidence are optional. This guide leads from installation and a first run to the scientific pipeline, scoring, selection, backfill, pruning and post-selection processes.

Scope

- Coding gene-model selection and annotation integration using CDS structure as the primary comparison representation.
- Stable retained primary transcripts, optional supported alternatives, bounded CDS-terminus refinement and separate optional UTR augmentation.
- No claim that agreement alone establishes biological correctness; correlated prediction inputs are not independent validation.
- No general recommendation of a particular predictor subset or pruning level for every organism and project.

Document authority

README.md is the entry point; INSTALLATION.md governs installation; docs/configuration.md governs controls and defaults; docs/input-source-guidance.md governs input sources and evidence support; docs/outputs.md governs files and fields; docs/benchmark.md governs benchmark results and methodology; docs/technical/ contains exact methods. This guide explains workflows and scientific meaning without duplicating every expert option or output column.

Contents

1. Purpose, biological scope and limitations
2. Installation and obtaining help
3. Input requirements and preflight checklist
4. First run and routine workflows
5. Main outputs and output profiles
6. Interpreting a run
7. Troubleshooting
8. Scientific pipeline
9. Scoring and evidence
10. Selection, overlap resolution and backfill
11. Final pruning and automatic assessment
12. Primaries, alternatives, refinement and UTRs
13. Reproducibility and audit trail
14. Glossary
15. Technical-reference navigation

The contents entries are internal hyperlinks. PDF bookmarks follow the same Heading 1–3 hierarchy.

1. Purpose, biological scope and limitations

MoSelecta is intended for projects in which several annotation or prediction methods propose overlapping but non-identical coding structures. It reads GFF/GTF records, constructs comparable CDS models, evaluates within-run agreement and optional external evidence, resolves competing models, and writes a standardized final GFF3 with an auditable decision trail.

What MoSelecta does

- Normalizes and validates CDS-containing candidates while preserving source provenance.
- Builds superloci and first-pass subloci, measures structural support and ranks candidates deterministically.
- Resolves overlap among provisional winners, backfills compatible candidates and can atomically replace eligible lower-scoring blockers.
- Applies fixed or reference-free automatic final pruning with explicit rescue and protection routes.
- Freezes public primary identities before optional alternative-transcript selection.
- Optionally assesses sequence integrity, refines explicitly incomplete CDS termini and reconstructs evidence-supported UTRs in a separate product.

What it does not establish

- A high score is not a calibrated probability that a model is correct.
- Source-level mean scores are descriptive for one run, not validated predictor-accuracy estimates.
- Multiple correlated prediction sources do not become independent experimental evidence.
- The ordinary final annotation is CDS-centred; unsupported source UTRs or arbitrary positional children are not propagated automatically.
- A computational locus or overlap relationship is not by itself a biological gene definition.

2. Installation and obtaining help

MoSelecta is pure Python and has no runtime dependencies. Install it with pip on Python 3.10 to 3.15, or run it from the Docker image without installing Python at all. The full instructions, including wheel, source-distribution and source-checkout installs, are in INSTALLATION.md.

Install with pip in a dedicated virtual environment (Linux or macOS):

```
python3 -m venv .venv
source .venv/bin/activate
pip install moselecta
moselecta --version
```

To install this specific release, use pip install moselecta==0.3.0. Windows PowerShell commands are given in INSTALLATION.md.

Alternatively, run the Docker image gemygk/moselecta, which contains MoSelecta and Python:

```
docker pull gemygk/moselecta:0.3.0
docker run --rm gemygk/moselecta:0.3.0 moselecta --version
```

On HPC systems that provide Apptainer or Singularity instead of Docker, use the same image directly:

```
apptainer exec docker://gemygk/moselecta:0.3.0 moselecta --version
```

For example data, see examples/README.md, which describes the Bombus 4 Mb example in examples/MoSelecta-bombus-4mb-example.zip.

Command	Purpose
moselecta --help	Routine inputs and workflow choices.
moselecta --full-help	Advanced biologically interpretable controls.
moselecta --expert-help	Complete supported CLI, thresholds and output-path controls.

The output-prefix parent must already exist and be writable. MoSelecta may also need write permission beside a genome FASTA when a missing or stale .fai index must be created or rebuilt.

3. Input requirements and preflight checklist

Prediction annotations

Supply prediction GFF/GTF files with --inputs. Each usable coding model requires CDS features that can be grouped into a transcript. Source labels must align one-for-one with expanded input paths. Repeated arguments, comma-separated paths and @FILE manifests are accepted.

Optional inputs

Input	Option	Scientific effect
Genome FASTA	--genome-fasta	Enables sequence-integrity metrics, hard definite-stop evaluation, relevant pruning safeguards and CDS terminus refinement.
Protein alignments	--evidence-proteins	Adds source-collapsed protein-CDS support, optional ranking contribution, rescue and evidence-aware pruning.
Transcript annotations	--evidence-transcripts	Adds exon-chain/cDNA-context support, optional ranking contribution, rescue, pruning and UTR evidence.

Preflight

1. Confirm every candidate annotation contains CDS records and consistent transcript/gene relationships.
2. Confirm sequence names and strands agree across annotations, genome and evidence inputs.
3. Choose unique source labels and verify manifest expansion order.
4. Create and test-write the output directory.
5. If using a genome, confirm FASTA readability and permission to create or rebuild its .fai.
6. Start with the standard output profile; request full/debug only for investigation.

Candidate eligibility is not pruning

Malformed records, invalid mixed-contig/strand CDS geometry, a definite internal stop under the accepted hard rule, or coding length below the 30-nt floor can remove a model before locus construction. These exclusions cannot be rescued by later evidence. Intrinsic score concerns and final pruning are later mechanisms with separate reports.

4. First run and routine workflows

Maintained model-only fixture

```
mkdir -p results
moselecta \
  --inputs tests/data/alternative_splice_smoke_a.gff3 tests/data/alternative_splice_smoke_b.gff3 \
  --source-name predictor_a predictor_b \
  --out-prefix results/example
```

Verified fixture outcome: two valid multi-exon candidates form one two-source sublocus. One model is selected; the other is rejected during backfill because it shares a CDS splice site and exceeds the CDS-exon overlap threshold. Automatic pruning resolves to medium and removes no model. The final annotation has one gene, one primary transcript and no accepted alternative. This is a software trace, not a biological benchmark.

Typical genome- and evidence-aware run

```
moselecta \
  --inputs predictions_a.gff3 predictions_b.gff3 \
  --source-name method_a method_b \
  --genome-fasta genome.fa \
  --evidence-proteins proteins.gff3 \
  --evidence-transcripts transcripts.gff3 \
  --out-prefix results/species
```

The second command is schematic: replace paths with project inputs. Omitting genome or evidence leaves the corresponding components neutral/inactive rather than invalidating a model-only run.

Common workflow switches

Need	Option
Primary-only final annotation	--no-alternative-splice
Detailed alternative decisions	--alternative-splice-report
Separate UTR-augmented annotation	--add-utrs
Fixed pruning sensitivity review	--pruning-level none|low|medium|high|very-high
Conservative automatic choice	--pruning-level auto-conservative
Every diagnostic family	--output-profile full
One additional product	--write-output OUTPUT_KEY

5. Main outputs and output profiles

Use <prefix>.final_annotation.gff3 for the normal integrated result. The optional <prefix>.final_annotation.with_utrs.gff3 is separate. The primary-only standardized file is useful for workflows that do not want accepted alternatives.

Product	Interpretation
final_annotation.gff3	Preferred ordinary integrated annotation: frozen primaries plus accepted alternatives.
standardised_primary_models.gff3	Frozen .1 primaries only.
final_annotation.with_utrs.gff3	Optional post-selection UTR-augmented product.
run_configuration.json	Resolved options and run identity.
model_selection_score_summary.tsv	Source-level descriptive summaries; not source accuracy.
final_selected_models.pruned.removed.tsv	Why selected models were removed or protected during final pruning.

Profile	Use
minimal	Compact products and summaries.
standard	Default final products plus normal selection/action ledgers.
full	Every diagnostic family, subject to feature gates.
debug	Full membership with debug execution context.

The exact output keys, suffixes, fields and state glossary are maintained in docs/outputs.md.

6. Interpreting a run

Read in stage order

1. Check input_summary.tsv, excluded_models.tsv and validation_failures.tsv.
2. Inspect model_selection_scores.tsv under full output for signed components and active evidence.
3. Use winner, selected and suppressed ledgers to understand conflict resolution.
4. Use backfill and strict-replacement reports for recovered, rejected or replaced models.
5. Use pruning diagnostics and retained/removed ledgers for final primary membership.
6. Use alternative-splice reports for .2+ decisions; the final GFF3 is the emitted product.

Model scores versus source summaries

model_selection_score is the signed integrated aggregate used for candidate ordering and strict replacement. model_selection_score_scaled clips non-positive values to zero and divides positive values by the theoretical active positive maximum. It supports reporting and configured threshold decisions; it does not replace unscaled ordering.

mean_scaled_score_all and mean_scaled_score_non_singletons summarize per-model scaled scores by source. The non-singleton mean excludes rows with sublocus_model_count == 1. A single-candidate sublocus is not the same as a single-exon model or a one-source run.

Candidate states

State	Meaning
Excluded	Removed before competitive selection by eligibility/validation.
Sublocus winner	Top candidate in a first-pass group; provisional because later conflict stages remain.
Suppressed	Winner blocked during monosublocus overlap resolution.
Backfilled	Compatible non-winner restored after conflict resolution.
Replaced	Holder removed atomically by an eligible strictly higher-scoring candidate.
Pruned	Selected model removed by final pruning.
Retained/primary	Model surviving pruning and frozen as .1.
Alternative	Post-freeze accepted isoform assigned .2 or higher.

7. Troubleshooting

Symptom	Checks and next file
Missing/malformed CDS	Check feature types, Parent/transcript_id grouping and excluded_models.tsv.
Source labels do not match	Expand comma/repeated/@FILE inputs and ensure equal label count/order.
Unexpectedly small candidate set	Read input, exclusion, validation, internal-stop and extreme-short summaries before selection.
Output path error	Create the parent of --out-prefix and verify it is writable.
FASTA index error	Check FASTA/.fai consistency and write permission beside the FASTA.
Contig mismatch	Compare sequence names, prefixes and case across inputs/evidence/genome.
Unexpected pruning	Read pruning_scenario_summary.tsv, auto_pruning_diagnostics.tsv and pruned.removed.tsv.
Candidate absent from final	Trace exclusion → score/winner → suppression → backfill → pruning → alternative decision.
Source mean seems high/low	Treat it as within-run description; inspect model population and singleton composition.

Worked fixture trace

In the maintained two-file fixture, both candidates pass validation and appear in the scored population. They occupy one sublocus, so only one becomes the provisional winner. The other is not a suppressed winner; it is considered by backfill and rejected because it shares a CDS splice site and conflicts by CDS-exon overlap. It therefore never enters the retained-primary population and cannot become an alternative under the fixed alternative rules.

8. Scientific pipeline

The stages below are conceptual checkpoints aligned with the accepted execution. Some implementation functions combine bookkeeping, but the ordering and data dependencies are real.

Stage	Purpose	Main operation	Important audit
1 Parse/normalize	Create CDS models and provenance	Read GFF/GTF, group records, normalize CDS geometry	input/exclusion summaries
2 Early eligibility	Remove candidates that cannot enter comparison	Validation, definite-stop/invalid-geometry and minimum coding length	excluded_models.tsv
3 Superloci	Bound potentially interacting models	Transitive same-strand CDS-span overlap components	superloci/summaries
4 First-pass subloci	Define conservative competition contexts	Shared splice sites or single-exon reciprocal overlap	sublocus members/edges
5 Structural facts	Measure support, consensus and candidate properties	Feature fractions, pairwise/consensus metrics, intrinsic facts	structural_support.tsv
6 Model-only scoring	Combine prediction-model evidence and penalties	Signed weighted model aggregate	model selection scores
7 Genome integrity	Assess translated/splice consistency when FASTA exists	Start/stop, internal stops, phase and motifs	sequence diagnostics
8 External evidence	Build optional protein/transcript support	Source collapse, structural matching and cDNA context	evidence support TSVs
9 Integrated score/ranks	Order candidates for competitive decisions	Add active evidence; scale/report deterministic ranks	model_selection_scores.tsv
10 Sublocus winners	Choose one first-pass candidate per group	Selection comparator	sublocus_winners.tsv
11 Winner de-overlap	Resolve conflicts between independent winners	Monosublocus graph and greedy retention	selected/suppressed TSVs
12 Backfill/replacement	Recover compatible non-winners	Greedy conflict checks; strict atomic replacement	backfill/R1 reports
13 Final pruning	Remove documented low-confidence/problematic selections	Fixed or automatic scenario with rescue/protection	pruned ledgers
14 Freeze primaries	Stabilize public gene and .1 identities	Assign retained-primary IDs	standardised primaries
15 Alternatives	Add supported coding isoforms	Ownership, support/evidence, thresholds and redundancy	alternative reports

Stage	Purpose	Main operation	Important audit
16 CDS refinement	Repair explicitly incomplete starts/stops	Bounded sequence/structure proposal and rollback	refinement audit
17 Final/UTR products	Emit ordinary and optional augmented annotations	Write final GFF3; optionally select UTR evidence	final GFF3 products

Parsing, correction and exclusion

MoSelecta first constructs a normalized CDS model and source identity. Early boundary canonicalisation and phase recovery may make sequence interpretation unambiguous. Malformed or impossible models are reported according to strict/permissive and validation policies. Definite internal stops and the 30-nt coding floor are hard pre-locus decisions, not negative scores and not final pruning.

Why superloci and subloci both exist

A superlocus is a broad same-strand, overlapping genomic context and bounds backfill. A sublocus is a conservative first-pass competition graph: multi-exon candidates connect through shared CDS splice sites; single-exon candidates connect by reciprocal CDS-span overlap; the two exon classes do not connect. This lets comparable structures compete without forcing every spatial overlap into one initial gene hypothesis.

From facts to integrated ranking

Structural, consensus, pairwise and candidate-intrinsic metrics are calculated in the sublocus context. Genome integrity and protein/transcript evidence are optional. The signed model-only aggregate is calculated first; active external evidence then creates the integrated model_selection_score. Scores are not recomputed after every selection decision.

Why winner selection is not the end

Each sublocus yields a provisional winner, but winners from different subloci can still overlap substantially. Monosublocus de-overlap resolves those conflicts. Backfill then revisits compatible non-winners because the first-pass one-winner rule was deliberately conservative. Only after that membership is complete does final pruning apply quality/evidence safeguards.

Post-selection ordering

Pruning freezes the primary population. Alternative-transcript selection cannot replace a primary. CDS terminus refinement can alter explicitly incomplete CDS geometry but not membership or scoring. UTR augmentation is last and writes a separate product so UTR evidence cannot retroactively influence CDS selection.

9. Scoring and evidence

The score families

Field	Definition	Primary use
component scores	One implemented aspect of one candidate	Audit and aggregate terms
model_structural_support_score	Signed weighted model-only aggregate	Model-only comparison and diagnostics
model_selection_score	Model-only aggregate plus active protein/transcript terms	Selection ordering and strict replacement
model_selection_score_scaled	Clipped positive transform using active positive maximum	Reporting and configured thresholds
source means	Means of model-level scaled values by source	Descriptive run diagnostics only

The integrated score is not a raw metric: it already contains weighted positive and negative contributions. It can be negative. The scaled value maps zero/negative scores to zero and therefore cannot preserve ordering among them.

Component overview

Component	Sign	What it measures
Structural support	+	Other-source support for the candidate's introns, splice sites, exons or single-exon interval
Consensus similarity	+	F1 similarity to features supported by at least two sources
Pairwise similarity	+	Mean of the best model comparison independently chosen per other source
Completeness	+	Recovery of supported terminal/internal structure without directional containment loss
Source prior	+	Configured or automatic within-run source weight; not source accuracy
Terminal extension	+	Whether consensus-relative extra bases are terminal rather than internal
Exact structure support	+	Other-source exact CDS-exon-chain support scaled by run depth
Structural corroboration	+	Run-aware whole-chain support from distinct other prediction sources
Unsupported short CDS exon	−	Graded concern for moderately short exons lacking another-source support
Structural coherence	−	Source-collapsed evidence that a candidate joins two separately supported hypotheses
Sequence integrity	−	Internal stops, phase inconsistency, missing termini and noncanonical intron fraction
Intrinsic	−	Short/modulo/short-or-long intron/exon, containment and unsupported-contained-intron concerns
Partial/internal-extra	−	Consensus-relative missing structure or internal extra CDS

Component	Sign	What it measures
Protein evidence	+	Optional source-collapsed protein structural support
Transcript evidence	+	Optional exon-chain/cDNA support, adjusted for calculated low CDS/cDNA context

Support, consensus, pairwise and corroboration are different

Structural support asks whether the candidate's own features have other-source support. Consensus similarity compares the candidate with source-supported consensus features. Pairwise similarity finds the best comparison in each other source and averages those best hits. Exact support requires complete local chain agreement. Corroboration records complete-chain support anywhere in the run and normalizes it by run depth. A candidate can therefore be strong on one concept and weaker on another.

Representative feature weights

Score	Multi-exon feature weights	Single-exon definition
Structural	intron .25; splice .20; internal exon .20; terminal exon .15; exact intron chain .20	overlap support .60; exact interval .40
Consensus	intron .30; splice .25; internal exon .20; CDS bases .15; terminal exon .10	CDS-base F1 .75; overlap support .25
Pairwise	intron .30; splice .25; internal exon .20; CDS bases .15; terminal exon .10	CDS-base F1 1.00
Completeness penalty	missing terminal .50; missing internal .35; containment .15	Same base-level definition when consensus exists

When a feature is not applicable, such as internal CDS exons in a one- or two-exon CDS, remaining weights are renormalized. Applicable absence is a true zero. Exact terminal matching can allow the configured strand-aware 3-prime outer-boundary wobble.

Model-only and integrated aggregates

```
model_selection_score =
  model_structural_support_score
  + protein_weight × protein_evidence_score
  + transcript_weight × transcript_evidence_score_context_adjusted
```

The model-only aggregate has eight positive terms and seven penalty families. Default positive model weights sum to 4.85. Active protein and transcript weights are 3.0 and 2.0; absence of an evidence class gives an effective weight of zero rather than a missing-evidence penalty.

Scaling

```
scaled = 0                                  if score <= 0
scaled = min(score / active_positive_maximum, 1) otherwise
```

Active terms	Default denominator
Model only	4.85
Model + protein	7.85
Model + transcript	6.85

Active terms	Default denominator
Model + protein + transcript	9.85

Penalty bounds do not enter the denominator. A scaled score is not a percentage probability or a value directly comparable across materially different active weights. Unscaled scores drive selection and strict replacement; scaled thresholds are used in pruning and alternative selection.

Low-source conflict dampening

In a two-source/two-model sublocus with an actual other-source comparison, no exact structure support and pairwise similarity below 0.80, structural-support and consensus contributions are multiplied by 0.25 by default. Component facts remain unchanged; only those two aggregate contributions are damped.

Genome sequence and hard exclusions

```
penalty = 1.00×min(internal stops,2) + 0.75×phase inconsistent
        + 0.75×missing stop + 0.50×missing start
        + 1.00×noncanonical intron fraction
```

GT-AG, GC-AG and AT-AC are canonical. Ambiguous termini and exact contig-boundary absence are neutral. This weighted penalty differs from the earlier hard definite-internal-stop/invalid-geometry exclusion and 30-nt coding floor, which occur before loci and cannot be rescued.

Protein and transcript evidence

Protein alignments are collapsed by exact CDS-exon chain within evidence superloci. Each source contributes its best match; single-exon scoring emphasizes CDS-base F1, whereas multi-exon scoring also uses introns, splice sites and internal exons. A default best-source score of 0.5 defines a supporting protein source.

Transcript evidence is built from exons and collapsed by exact intron chain (multi-exon) or exact interval (single-exon). Only scoring-eligible associations contribute. Multi-exon score terms include source-best pairwise support, source-weighted support, chain-source support and confidence; single-exon scoring emphasizes coverage/support. Calculated low CDS/minimum-cDNA context can subtract at most 0.15 from the transcript score used in ranking; unavailable context is neutral.

Ranks and exact ties

Score ranks group values within the default 1e-6 epsilon and use competition numbering. The unique model_selection_order_rank reports the same exact total order used by first-pass primary selection: score, completeness, intrinsic penalty, consensus, structural support, source prior, CDS length and transcript ID. It is assigned within each scored canonical sublocus before later suppression, backfill and pruning, so rank 1 identifies the first-pass winner but does not guarantee final retention.

Exact formulas, all weights, parameter effects, missing-data behavior and diagnostic fields are maintained in docs/technical/scoring.md.

10. Selection, overlap resolution and backfill

Sublocus winner selection

Within each first-pass sublocus, scored candidates are ordered by the selection comparator and the first becomes the provisional winner. Provisional is appropriate here: a winner may still be suppressed, replaced or pruned. The score itself is not provisional and is not later replaced by a different final score.

Monosublocus de-overlap

Winners from different subloci can overlap because first-pass grouping is conservative. Same-strand winners receive an edge when their CDS exons overlap and the overlap divided by either model's CDS length reaches 0.10. Connected components form monosublocus holders. Members are visited in score/comparator order and kept only if they have no edge to a previously kept member.

```
max(overlap_bp / CDS_bp_A, overlap_bp / CDS_bp_B) >= 0.10
```

The threshold is non-reciprocal and uses CDS-exon intersections, not outer CDS spans. A transitive component need not be all-pairwise overlapping; the greedy algorithm can retain multiple mutually compatible members. A losing winner is suppressed, not excluded or pruned.

Ordinary backfill

Backfill revisits scored candidates within each superlocus that are neither already selected nor suppressed winners. Candidates are considered in deterministic selection order. A candidate conflicts if two multi-exon models share any CDS splice site or if CDS-exon overlap reaches the same 0.10 non-reciprocal threshold. Conflict-free candidates are added immediately and can block later candidates.

Backfill can increase the retained population because the one-winner-per-sublocus rule deliberately resolved an initial competition rather than asserting that every non-winner is unusable.

Strict score-improving replacement

A blocked candidate can replace holder winners only under a narrow atomic rule. The candidate must be multi-exon, corroborated by at least one other prediction source and blocked by exactly one or two uncorroborated models that were selected at the monosublocus stage. Backfilled models are not eligible blockers. The multi-exon extension allows eligible multi-exon holders as well as single-exon holders.

```
candidate unscaled score > blocker 1 unscaled score
candidate unscaled score > blocker 2 unscaled score   # when present
```

Scores are compared separately; blocker scores are never summed. Equality fails. All eligibility and conflicts are checked before any mutation. If one safeguard fails, no blocker is removed. Accepted replacement removes every blocker and inserts the candidate as one atomic change.

Conceptual example

1. A and B share a splice site and form one multi-exon sublocus. A is ordered first and becomes its provisional winner.
2. Winner C comes from another sublocus but has a threshold CDS-exon overlap with A. In their monosublocus, A is visited first; C is suppressed.
3. B is a non-winner and therefore a backfill candidate. It is accepted only if it does not share a splice site or threshold CDS-exon overlap with the current selected set.
4. If B is a corroborated multi-exon candidate blocked only by one or two eligible uncorroborated holder winners and is strictly higher-scoring than each, it can replace them atomically.
5. The resulting selected set enters pruning. A later pruning removal does not trigger a new backfill pass.

This example is conceptual and uses no invented score values. Exact predicates, comparator fields, R1/R1-ME mappings and rejection codes are in docs/technical/selection-and-backfill.md.

11. Final pruning and automatic assessment

Why pruning is a separate stage

Selection asks which candidates can coexist after competitive ranking and backfill. Pruning asks whether an already selected model should remain in the final primary population under explicit intrinsic, score, sequence, evidence and opposite-strand-shadow safeguards. Separation preserves the unpruned selection ledger and makes removals auditable.

Rule order and protection

1. Identify singleton-sublocus candidates rather than assuming every singleton must be removed.
2. Apply monosublocus-overlap rescue where a singleton has relevant selected-neighbour context.
3. Apply protein and transcript rescue/protection when configured support meets exact criteria.
4. Evaluate low-score singleton removal.
5. Evaluate low-score nonsingleton removal in its configured scope/regime.
6. Evaluate sequence-integrity nonsingleton removal in its configured scope.
7. Evaluate opposite-strand shadow pairs and retain the stronger member under the shadow rule.

Candidate detection is not removal. A model can meet a low-score or intrinsic predicate yet remain because a rescue/protection route applies. Protein/transcript evidence affects the documented evidence-aware routes; it is not a universal override. Internal-stop rescue has separate conservative safeguards.

Fixed levels

Level	Interpretation
none	No final selected-model pruning.
low	Low-intensity documented routes.
medium	Canonical routine fixed scenario and default low-depth automatic result.
high	Stronger score/sequence scope under the documented matrix.
very-high	Most stringent canonical fixed scenario; not automatically selected without adequate evidence.

The exact threshold matrix belongs in docs/technical/automatic-pruning.md, not in the routine guide. Scaled score thresholds use clipped positive transforms; other structural/intrinsic/evidence predicates remain separate.

How automatic pruning chooses

auto and auto-conservative simulate the canonical medium, high and very-high scenarios against the same immutable pre-pruning population. The simulations are reconciled to preserve monotonic retention. The assessor then asks whether protein/transcript evidence is globally adequate and how prediction sources form a consensus backbone.

Assessment	Question
Evidence adequacy	Do supplied protein/transcript classes cover and support enough selected models to justify evidence-aware decisions?
Consensus backbone	Do source-incidence and deep-consensus metrics indicate strong, intermediate or weak/distributed agreement?

Assessment	Question
Flood-aware view	Does one source contribute a strong singleton excess that masks an otherwise stronger multisource backbone?
Scenario increments	How many additional models/routes/sources are removed by high beyond medium, and very-high beyond high?
User constraints	Do configured automatic minimum/maximum levels change the unconstrained recommendation?

Without an adequacy-passing evidence class, automatic selection never chooses very-high; fewer than three model sources resolve to medium. Standard auto can use an eligible flood-adjusted backbone, whereas auto-conservative uses the unadjusted class and has a more conservative default maximum. Advisory input-count warnings do not alter the recommendation.

Recommendation versus effective level

Diagnostics report the unconstrained recommendation and the effective level after user minimum/maximum constraints. A warning or source-count ratio is advisory, not an automatic exclusion. If scenario accounting or monotonicity validation fails, the implementation falls back to medium with low confidence and reports the failure.

Opposite-strand shadows

Shadow pruning is not a generic low-score route. It considers opposite-strand selected models with at least 150 bp CDS-exon overlap, at least 0.80 overlap of the loser and a scaled-score margin of at least 0.05, then applies its documented evidence/sequence protections. Removed/retained ledgers record the pair and reason.

Read pruning_scenario_summary.tsv, auto_pruning_diagnostics.tsv and final_selected_models.pruned.removed.tsv together. Exact criteria, thresholds, protections and reason codes are in docs/technical/automatic-pruning.md.

12. Primaries, alternatives, refinement and UTRs

Freeze primaries first

After pruning, retained models receive stable public gene and `.1` transcript identities. Every later step treats that population as authoritative. Alternative selection may add `.2+` isoforms but cannot replace a primary; refinement can alter explicitly incomplete CDS geometry but not membership; UTR augmentation cannot change CDS selection.

Alternative-transcript selection

A possible alternative must associate uniquely with one retained primary, be structurally distinct under the exact/intron-chain/relaxed-containment rules, have calculable frame, satisfy the configured prediction-model support policy and any external-evidence requirement, pass numerical score/overlap/frame thresholds, survive greedy redundancy checks and fit the cap. UTR-only differences do not create coding alternatives.

The default exact support policy accepts an exact multisource model route, exact protein CDS-chain support or a coherent exact transcript association. The exact-plus-mosaic policy can additionally accept a qualifying full structural mosaic. Candidate/primary score ratio uses unscaled scores only when the primary score is positive.

CDS terminus refinement

With genome sequence, the default post-selection stage examines retained CDS sides explicitly classified as missing a start or stop. It follows a preferred compatible structure, searches the established frame for a valid codon, applies extension/truncation, ambiguity, splice, internal-stop and cross-gene guards, then validates the whole proposed CDS. Unsafe proposals roll back. It never rescales, reselects, reprunes or renumbers models.

A successful repair is represented by a cloned final-annotation model with consistent CDS parts, phases, sequence metrics and children. The original selected model remains the immutable provenance record. Audit rows record one outcome for every incomplete biological side.

UTR augmentation

UTR splice-junction validation runs only after coding-model selection is frozen. For the ordinary final annotation, MoSelecta assesses inherited biological 5-prime and 3-prime UTR sides independently. A definite geometry failure, intron shorter than the effective 20 bp default minimum, or resolved disallowed motif suppresses only the affected side. The authoritative CDS, public IDs and independently valid opposite side remain unchanged.

With --add-utrs, the same validator covers every prediction-derived and transcript-derived candidate before support pooling, consensus, automatic extension-limit learning or side selection. GT-AG, GC-AG and AT-AC use the shared CDS motif definition; there is no separate UTR motif option. Transcript evidence rejected for an anomalous UTR side remains available for its separate CDS scoring and model-selection roles. Without genome FASTA, length and geometry checks still run, while motif status is recorded as not evaluated and does not alone reject a side. All profiles report compact totals. Full/debug, or an explicit output request, adds only rejected or unusual exception rows; passing and routine missing-FASTA rows are omitted. Reporting never changes decisions. The ordinary final_annotation.gff3 and optional final_annotation.with_utrs.gff3 may therefore differ in UTR structure while retaining the same coding-model identities and CDSs.

Process	Can change membership?	Can change CDS geometry?	Output
Alternative selection	Adds alternatives only	Uses candidate CDS as accepted	ordinary final GFF3
CDS terminus refinement	No	Yes, only explicit incomplete sides after safeguards	ordinary final GFF3 + audit

Process	Can change membership?	Can change CDS geometry?	Output
UTR augmentation	No	No (consumes authoritative CDS)	separate with_utrs GFF3 + audit

Exact rules are in docs/technical/alternative-transcripts.md, cds-terminus-refinement.md and utr-augmentation.md.

13. Reproducibility and audit trail

- Record the exact MoSelecta version, command, input checksums, source-label order and environment.
- Retain run_configuration.json and standard decision ledgers with the final GFF3.
- For method review, retain full scoring/evidence/pruning products and warnings.
- Do not compare scaled scores across materially different active weight/evidence configurations without accounting for the denominator.
- Do not treat source means or automatic source warnings as validated predictor-performance results.
- Trace a model in chronological state order instead of inferring its fate from the final GFF3 alone.

Minimal scientific provenance bundle

Item	Why retain it
Input checksums and source labels	Reconstruct candidate/evidence identity and order.
run_configuration.json	Reconstruct requested and effective options.
model_selection_scores.tsv	Reconstruct signed score components and evidence integration.
selection/backfill ledgers	Reconstruct conflict membership, suppression and recovery.
pruning diagnostics and removed ledger	Reconstruct effective level, protection and removal reasons.
final GFF3 and optional UTR GFF3	Preserve emitted biological products separately.

14. Glossary

Term	Concise definition
Candidate model	Valid CDS-containing prediction model entering competitive assessment.
Superlocus	Transitive same-strand CDS-span overlap component; broad comparison/backfill context.
Sublocus	First-pass competition group formed by shared splice sites or single-exon reciprocal overlap.
Monosublocus	Conflict component among provisional winners formed by threshold CDS-exon overlap.
Structural support	Agreement of candidate features with other prediction sources in its sublocus.
Consensus	Source-supported structural features against which the candidate is compared.
Corroboration	Complete-chain support from distinct other prediction sources, normalized by run depth.
Suppression	Conflict-based loss of a sublocus winner during winner de-overlap.
Backfill	Post-conflict recovery of compatible scored non-winners.
Strict replacement / R1	Atomic replacement of one/two eligible blockers by a corroborated, strictly higher-scoring multi-exon candidate.
Pruning	Post-selection removal under explicit fixed/automatic rules and protections.
Model-level score	Per-candidate weighted aggregate used for ordering or thresholds.
Source-level mean	Descriptive aggregation of per-model scaled scores for one source.

15. Technical-reference navigation

Subject	Maintained authority
Installation	INSTALLATION.md
Options, defaults and interactions	docs/configuration.md and CLI help
Files, fields and states	docs/outputs.md
Scoring, evidence, scaling and ranks	docs/technical/scoring.md
Selection, conflicts and backfill	docs/technical/selection-and-backfill.md
Automatic pruning	docs/technical/automatic-pruning.md
Alternative transcripts	docs/technical/alternative-transcripts.md
Structural coherence	docs/technical/structural-coherence.md
CDS refinement	docs/technical/cds-terminus-refinement.md
UTR augmentation	docs/technical/utr-augmentation.md
Testing and packaging	docs/development/testing-and-regression.md
Benchmarking	docs/benchmark.md
Example data	examples/README.md

Release status: draft. Project-owner decisions remain for final version/date, repository and support metadata, citation wording, licence confirmation and any public biological validation claims.