Package: selecta (via r-universe)

October 8, 2026

Type Package

Title Declarative EQUATOR-Style Flow Diagrams for Clinical Studies

Version 0.6.1

Description Build EQUATOR-style flowcharts for clinical studies by sequentially defining inclusion and exclusion criteria, study arms, and endpoints. The pipe-friendly API supports CONSORT (randomized trials), STROBE (observational cohorts), STARD (diagnostic accuracy), PRISMA (systematic reviews), and MOOSE (observational meta-analysis) diagram layouts, as well as multi-source convergence, split-and-recombine, factorial, and hybrid topologies. Diagrams are rendered via 'grid' graphics in both data-driven (automatic counting) and manual-count modes, with optional 'DiagrammeR'/'Graphviz' output.

License GPL (>= 3)

Encoding UTF-8

LazyData true

URL https://phmcc.codefloe.page/selecta, https://codefloe.com/phmcc/selecta, https://github.com/phmcc/selecta

BugReports https://github.com/phmcc/selecta/issues

Depends R (>= 4.1.0)

Imports data.table, grDevices, grid, stats, tools, utils

Suggests DiagrammeR, knitr, ragg, rmarkdown, testthat (>= 3.0.0)

VignetteBuilder knitr

Config/testthat/edition 3

Config/roxygen2/version 8.0.0

Repository https://phmcc.r-universe.dev

Date/Publication 2026-10-08 04:09:28 UTC

RemoteUrl https://github.com/phmcc/selecta

RemoteRef HEAD

RemoteSha af10eb7ea311d0c0100daf218374c0a3d9c6bfa1

Contents

assess . . . 2
cohort . . . 4
cohorts . . . 5
combine . . . 6
endpoint . . . 7
enroll . . . 9
exclude . . . 10
flowchart . . . 13
flowsave . . . 16
phase . . . 19
print.selecta . . . 20
recdims . . . 21
selectaex0 . . . 23
selectaex2 . . . 24
selectaex3 . . . 25
selectaex6 . . . 25
sources . . . 26
stratify . . . 27
summary.selecta . . . 28

Index **30**

`assess` *Record an Assessment or Procedure Step*

Description

Models a step where participants undergo (or fail to undergo) a test or procedure. This is the primary building block for STARD-style diagnostic accuracy diagrams. The side box shows who did *not* receive the procedure (with optional reasons), and the main flow continues with those who *were* assessed.

Usage

```
assess(
  .flow,
  label,
  criterion,
  not_received = NULL,
  reasons = NULL,
  show_zero = FALSE
)
```

Arguments

.flow	A selecta object.
label	Character string naming the test or procedure (*e.g.,* "Index test", "Reference standard").
criterion	An unquoted logical expression that evaluates to TRUE for rows that did **not** receive the test. Data mode only.
not_received	Integer (manual mode). Number of participants who did not receive this test.
reasons	Named integer vector of reasons for non-receipt (*e.g.,* c("Refused" = 12, "Contraindicated" = 10)).
show_zero	Logical. If TRUE, display zero-count reasons. Default FALSE.

Details

assess() models a test or procedure that only part of the cohort undergoes, the recurring motif of STARD diagnostic-accuracy diagrams. It is implemented as an exclude() step with inverted label semantics: the side box reads "Did not receive *label*" and the continuing box reads "Received *label*", so the main flow carries those who *were* assessed. In data mode, criterion is an unquoted logical expression that is TRUE for participants who did **not** receive the test; in manual mode, not_received gives that count and reasons an optional named breakdown. Chained assess() steps commonly precede a stratify() split on the index-test result, with each terminal box reporting its target-condition breakdown.

Value

The updated selecta object with an assessment step appended.

See Also

exclude for general exclusion steps, endpoint for the terminal diagnosis boxes (STARD)

Other flow construction functions: combine(), endpoint(), enroll(), exclude(), phase(), sources(), stratify()

Examples

```
# STARD diagnostic accuracy flow
enroll(n = 360, label = "Eligible patients") |>
  assess("Index test", not_received = 22,
         reasons = c("Refused" = 12, "Contraindicated" = 10)) |>
  assess("Reference standard", not_received = 18) |>
  stratify(labels = c("Index test positive", "Index test negative"),
           n = c(150, 170), label = "Index test result") |>
  endpoint("Final diagnosis",
           breakdown = list(c("Target +" = 130, "Target -" = 20),
                            c("Target +" = 15, "Target -" = 155)))
```

cohort *Extract the Final Cohort*

Description

Returns the dataset remaining after all exclusion criteria have been applied. When arms are defined via `stratify()`, the result is either a single combined `data.table` or a named list of per-arm `data.table` objects. Data mode only.

Usage

```
cohort(.flow, split = FALSE, arm = NULL)
```

Arguments

`.flow`	A `selecta` object created in data mode (`data` supplied to `enroll()`).
`split`	Logical. If `TRUE` and arms are defined, return a named list of `data.tables` (one per arm). Default `FALSE` returns a single combined `data.table`.
`arm`	Character. Name of a specific arm to extract. If supplied, returns only that arm's `data.table`.

Details

`cohort()` replays the exclusion criteria of a *data-mode* flow against the original dataset and returns the rows that survive to the end, so the analyst can pass the exact analyzed population to downstream modeling. It requires a flow created by supplying `data` to `enroll()`; manual-mode flows carry only counts and therefore raise an error. For an unsplit flow the result is a single `data.table`; after `stratify()` or `allocate()`, `split = TRUE` returns one table per arm and `arm` extracts a single named arm. To inspect the cohort at every intermediate step rather than only the end, use `cohorts()`.

Value

A `data.table` containing the participants remaining after all exclusion criteria. When `split = TRUE`, a named list of `data.tables` (one per arm). When `arm` is specified, a single-arm `data.table`.

See Also

`cohorts` for stage-by-stage snapshots, `enroll` for initializing a data-mode flow

Other cohort extraction functions: `cohorts()`

Examples

```
flow <- enroll(selectaex2, id = "patient_id") |>
  exclude("Ineligible", criterion = eligible == FALSE) |>
  endpoint("Final")
```

```
final <- cohort(flow)
nrow(final)
```

`cohorts` *Extract Cohorts at Every Stage*

Description

Returns a named list of datasets at each step of the enrollment flow, enabling cross-cohort comparisons. Results are reported as a named list, organized by step label. Data mode only.

Usage

```
cohorts(.flow)
```

Arguments

`.flow`	A `selecta` object created in data mode (`data` supplied to `enroll()`).

Details

`cohorts()` replays a *data mode* flow and captures the dataset at every step, returning a named list keyed by step label (with `"_start"` for the initial cohort). Each snapshot exposes both the `included` and the `excluded` rows together with their counts, which is useful for validating a diagram against the data, auditing why particular participants were dropped, or extracting an intermediate population. After a `stratify()` or `allocate()` split, the `included` and `excluded` elements of a per-arm step are themselves named lists with one entry per arm; after a factorial (two-level) split the entries are the cells, keyed `"<parent>: <child>"`. A manual-mode flow has no underlying data and therefore raises an error. To obtain only the final analyzed population, use `cohort()`.

Value

A named list of cohort snapshots, keyed by step label. Each snapshot is itself a list with:

`included` A `data.table` of participants still in the flow after this step.

`excluded` A `data.table` of participants removed at this step (for exclusion steps; `NULL` otherwise).

`n_included` Integer count of included participants.

`n_excluded` Integer count of excluded participants (or `NA`).

See Also

`cohort` for extracting only the final cohort

Other cohort extraction functions: `cohort()`

Examples

```
flow <- enroll(selectaex2, id = "patient_id") |>
  exclude("Ineligible", criterion = eligible == FALSE) |>
  endpoint("Final")

stages <- cohorts(flow)
names(stages)
stages[["Ineligible"]]$n_excluded
```

`combine` *Merge Parallel Streams*

Description

Converges all active parallel streams into a single flow. Used to handle either source convergence or split-and-recombine topologies. After `stratify()`, recombines strata that were characterized independently back into a unified downstream flow.

Usage

```
combine(.flow, label, sublabel = NULL, n = NULL, reasons = NULL)
```

Arguments

`.flow`	A `selecta` object with active parallel streams (from `sources()` or `stratify()`).
`label`	Character string for the merged node.
`sublabel`	Optional character string rendered below `label` inside the same box. Useful for describing the recombined cohort.
`n`	Integer. Explicit post-merge count (manual mode). If omitted, computed as the sum of all active stream counts.
`reasons`	Optional named integer vector of sub-items displayed below the count (*e.g.,* outcome categories).

Details

`combine()` converges the active parallel streams into one node and is the counterpart to both entry splits. After `sources()`, it pools the identification streams of a systematic review; after `stratify()` (or `allocate()`), it recombines strata that were handled independently, producing a split-and-recombine diagram.

By default, the merged count is the sum of the incoming streams after any per-arm exclusions applied since the split—an explicit `n` overrides this in manual mode. In such situations, an additional option is provided (`getOption("selecta.check_arithmetic")`, default `TRUE`), which will check arithmetic and raise an advisory warning if there is a discrepancy between counts.

The optional `sublabel` parameter prints on a second line inside the merged box, which is convenient for naming the recombined cohort.

Value

The updated `selecta` object with a combine step appended. All subsequent steps operate on the single merged stream.

See Also

`sources` for multi-source entry, `stratify` for split-and-recombine flows

Other flow construction functions: `assess()`, `endpoint()`, `enroll()`, `exclude()`, `phase()`, `sources()`, `stratify()`

Examples

```
# PRISMA: merge identification sources
sources(PubMed = 1234, Embase = 567) |>
  combine("Records after deduplication") |>
  exclude("Records removed", n = 352, show_count = FALSE,
          reasons = c("Duplicates" = 340, "Automation" = 12))

# Split-and-recombine: stratify, then combine
enroll(n = 158) |>
  stratify(labels = c("Not screened", "Screened"), n = c(82, 76),
           label = "Screening status") |>
  exclude("Condition not confirmed", n = c(44, 66)) |>
  combine("Confirmed cohort",
          sublabel = "Participants with confirmed diagnosis") |>
  exclude("Incomplete records", n = 7) |>
  endpoint("Final cohort")
```

---

endpoint *Mark the Final Analysis Endpoint*

Description

Adds the terminal node(s) to the enrollment flow. If arms have been defined via `stratify()`, one endpoint box appears per arm.

Usage

```
endpoint(
  .flow,
  label = "Final Analysis",
  breakdown = NULL,
  groups = NULL,
  n = NULL,
  variable = NULL
)
```

Arguments

.flow	A selecta object.
label	Character string for the final box. With groups it labels the shared distributor box above the group boxes. Default "Final Analysis".
breakdown	Optional named numeric vector (or, for a per-arm endpoint, a list of them) itemizing the box total into parts printed *within* the box, beneath the total. This is the STARD final-diagnosis form, where each terminal box reports its target-condition composition, *e.g.* breakdown = c("Target +" = 160, "Target -" = 40). Mutually exclusive with groups.
groups	Optional character vector of group labels (manual mode). When supplied, the endpoint splits into one *separate* terminal box per group, fanning from a shared distributor. Use this for study-design diagrams that end by displaying the groups to be analyzed ("Group A", "Group B", ...). A split endpoint requires a single incoming stream; it cannot follow an unrecombined stratify() or allocate(). Mutually exclusive with breakdown.
n	Optional numeric vector of per-group counts (manual mode), parallel to groups.
variable	Optional character naming a grouping column (data mode). Splits the terminal endpoint by that column, one box per level, with counts tabulated automatically. The data-mode counterpart of groups/n; same single-stream requirement.

Details

endpoint() closes the flow with its terminal node(s) and is usually the last step in a pipeline. When the flow has been split with stratify() or allocate() and not recombined, one endpoint box is drawn per arm, and label and breakdown may be supplied per arm.

Two distinct presentations of detail are available, which are mutually exclusive. breakdown itemizes a single box's total as text lines inside that box (the STARD final-diagnosis form, reporting each box's target-condition composition). Conversely, groups divides the endpoint into separate side-by-side boxes, one per group, fanning from a shared distributor; this design favors study diagrams that end by displaying the groups to be analyzed. The completed object is then passed to flowchart(), flowsave(), or recdims().

Value

The updated selecta object with an endpoint step appended.

See Also

assess for the diagnostic test-receipt steps that precede a STARD endpoint, flowchart for rendering

Other flow construction functions: assess(), combine(), enroll(), exclude(), phase(), sources(), stratify()

Examples

```
enroll(n = 300) |>
  exclude("Excluded", n = 40) |>
```

```
  endpoint("Included in analysis")

# STARD-style per-arm endpoint with a within-box breakdown
enroll(n = 500) |>
  stratify(labels = c("Positive", "Negative"), n = c(200, 300),
           label = "Index test result") |>
  endpoint("Final diagnosis",
           breakdown = list(c("Target +" = 160, "Target -" = 40),
                            c("Target +" = 25, "Target -" = 275)))

# Split endpoint into separate terminal group boxes (manual)
enroll(n = 300, label = "Eligible cohort") |>
  endpoint("Allocated to study group",
           groups = c("Group A", "Group B", "Group C"),
           n = c(100, 100, 100))

# Split endpoint by a grouping column (data mode)
df <- data.frame(id = 1:300, grp = sample(c("A", "B", "C"), 300, TRUE))
enroll(df, id = "id", label = "Eligible cohort") |>
  endpoint("Allocated to study group", variable = "grp")
```

`enroll` *Initialize an Enrollment Flow*

Description

Entry point for building an EQUATOR-style enrollment diagram from a single starting population. Accepts either a `data.frame` (data mode, where counts are computed automatically from exclusion expressions) or a starting count `n` (manual mode, where counts are supplied explicitly at each step).

Usage

```
enroll(data = NULL, id = NULL, n = NULL, label = "Study Population")
```

Arguments

`data`	A `data.frame` or `data.table` in which each row represents one participant. When supplied, exclusion expressions passed to `exclude()` are evaluated against this data to compute counts automatically. If `NULL` (default), the flow operates in manual mode.
`id`	Character string naming the participant ID column in `data`. Defaults to the first column. Ignored in manual mode.
`n`	Integer. Starting population count for manual mode. Must be a non-negative scalar. Ignored when `data` is supplied.
`label`	Character string for the top-level box in the diagram. Default is `"Study Population"`.

Details

`enroll()` begins every single-source pipeline and fixes the operating mode for all subsequent steps. Supplying `data` (with `id`) selects *data mode*, in which later `exclude()` and `stratify()` steps filter and partition the dataset and counts are derived from the data. Alternatively, supplying `n` instead selects *manual mode*, in which counts are taken from the numbers given at each step. The two modes are mutually exclusive, and the resulting object is intended to be extended with the pipe operator. For diagrams with several entry sources that converge (PRISMA, MOOSE), use `sources()` instead of `enroll()`.

Value

An object of class `"selecta"` containing the data (if supplied), mode, starting count, label, and an empty step list. Subsequent pipeline functions (`exclude()`, `stratify()`, `endpoint()`, *etc.*) append steps to this object.

See Also

`sources` for multi-source entry, `exclude` for adding exclusion criteria, `flowchart` for rendering

Other flow construction functions: `assess()`, `combine()`, `endpoint()`, `exclude()`, `phase()`, `sources()`, `stratify()`

Examples

```
# Manual mode
enroll(n = 500, label = "Assessed for eligibility")

# Data mode
enroll(selectaex2, id = "patient_id", label = "Study Population")

# Minimal CONSORT pipeline
enroll(n = 500) |>
  exclude("Ineligible", n = 65) |>
  allocate(labels = c("Treatment", "Control"), n = c(218, 217)) |>
  endpoint("Analyzed")
```

`exclude` *Exclude Participants by Criteria*

Description

Appends an exclusion step to the enrollment flow. Participants matching the criteria are removed and shown in a side box. Optionally, itemized sub-reasons can be displayed below the total.

Usage

```
exclude(
  .flow,
  label,
  criterion,
  n = NULL,
  reasons = NULL,
  show_zero = FALSE,
  show_count = FALSE,
  included_label = NULL,
  collapse_singletons = FALSE
)
```

Arguments

.flow	A selecta object (piped from enroll() or a previous step).
label	Character. Human-readable description for the side box (*e.g.,* "Excluded" or "Lost to follow-up"). After stratify(), may be a character vector with one label per arm (*e.g.,* c("Treatment discontinued", "Initiated treatment")).
criterion	An unquoted logical expression evaluated against the data. Should evaluate to TRUE for rows to be removed. Compound conditions are supported using the vectorized operators & (and), | (or), and ! (not). Do not use the scalar short-circuit operators && or ||, which evaluate only the first element of each vector. Data mode only.
n	Integer. Number of participants removed at this step. After a stratify() step, supply a vector with one value per arm. Manual mode only.
reasons	Exclusion sub-reasons. Accepts these forms: • A *character string* (data mode): the name of a column whose values are tabulated automatically into a flat breakdown. • A *length-2 character vector* (data mode): the names of a reason column and a sub-reason column, cross-tabulated automatically into a two-level breakdown—parents ordered by total, sub-reasons by count. • A *named numeric vector* (manual mode): counts per reason, *e.g.,* c("Disease progression" = 12, "Declined" = 8). An entry may itself be a named numeric vector, giving a two-level breakdown (a reason and its sub-reasons). • A *list* of any of the above (data or manual mode after stratify()): one entry per arm.
show_zero	Logical. If FALSE (default), sub-reasons with a count of zero are hidden. Set to TRUE to display all pre-specified reason categories, including those with zero participants.
show_count	Logical. If FALSE (default), the intermediate count box is suppressed—the count still updates internally but no box is rendered. Set to TRUE to force a count box. Overridden by included_label: providing any included_label always creates a count box regardless of show_count. Also automatically suppressed when the next step is stratify(), endpoint(), or allocate().

`included_label` Character string (or vector). Optional text for the box showing the count remaining after exclusion. When provided, a count box is always rendered regardless of `show_count`. After `stratify()`, may be a character vector with one label per arm.

`collapse_singletons` Logical. When `TRUE`, a parent reason that resolves to a single sub-reason is collapsed to a plain leaf carrying the parent's label and count (dropping the lone, redundant sub-line). Applies to two-level reasons from either a manual nested specification or a two-column data-mode cross-tabulation. Default `FALSE` keeps every parent expanded, for full transparency.

Details

`exclude()` records participants removed at a step and is the most common pipeline verb. In data mode, `criterion` is an unquoted logical expression evaluated against the dataset (rows for which it is `TRUE` are removed) and `reasons` may name one column (a flat breakdown) or two columns (a reason and a sub-reason, cross-tabulated into a two-level breakdown); in manual mode, `n` gives the number removed and `reasons` may be a named numeric vector. After a `stratify()` or `allocate()` split the exclusion applies per arm, in which case `n`, `reasons`, and `included_label` accept per-arm vectors or lists. By default the running count box is suppressed between consecutive exclusions for a compact diagram; supplying `included_label` (or `show_count = TRUE`) forces a count box to be drawn.

When `getOption("selecta.check_arithmetic")` is `TRUE`, the manual counts of the whole flow are audited together before export: an over-exclusion, a split or combine whose parts do not match the running total, and sub-reasons that do not sum to their exclusion total each raise an advisory warning without altering the figures. The audit runs whenever the flow is computed; this includes calls to `flowchart()`, `flowsave()`, and `summary()`, so a single call to any of these functions reports every discrepancy at once.

Eligibility that is more naturally framed as inclusion fits this same model: express it as the exclusion of those who fail the criteria, and use `included_label` to label the retained count (*e.g.,* `included_label = "Eligible cohort"`).

After a `stratify()` step, both `label` and `included_label` accept character vectors (one element per arm) for per-arm labeling—useful in observational designs where attrition mechanisms differ across strata.

Value

The updated `selecta` object with an exclusion step appended.

See Also

`assess` for assessment/procedure steps (STARD), `enroll` for initializing a flow

Other flow construction functions: `assess()`, `combine()`, `endpoint()`, `enroll()`, `phase()`, `sources()`, `stratify()`

Examples

```
enroll(n = 500) |>
```

```
  exclude("Ineligible", n = 65)

# With sub-reasons (manual)
enroll(n = 500) |>
  exclude("Excluded", n = 65,
    reasons = c("Did not meet criteria" = 22,
                "Ineligible comorbidities" = 18,
                "Declined to participate" = 15,
                "Lost to follow-up" = 10))

# Show intermediate count box (opt-in)
enroll(n = 500) |>
  exclude("Ineligible", n = 65, show_count = TRUE) |>
  exclude("Declined", n = 20) |>
  endpoint("Final")

# Or use included_label (always shows count box)
enroll(n = 500) |>
  exclude("Ineligible", n = 65,
          included_label = "Eligible") |>
  endpoint("Final")

# Per-arm labels (observational)
enroll(n = 1000) |>
  stratify(labels = c("Exposed", "Unexposed"), n = c(500, 500),
           label = "Classified by exposure") |>
  exclude(c("Treatment discontinued", "Initiated treatment"),
          n = c(45, 52))

# Per-arm reasons (list of named vectors)
enroll(n = 900) |>
  allocate(labels = c("Drug A", "Placebo"), n = c(450, 450)) |>
  exclude("Discontinued", n = c(30, 25),
          reasons = list(
              c("Adverse event" = 18, "Withdrew consent" = 12),
              c("Adverse event" = 10, "Lost to follow-up" = 15)
          )) |>
  endpoint("Analyzed")

# Compound expression (data mode)
data(selectaex2)
enroll(selectaex2, id = "patient_id") |>
  exclude("Ineligible or duplicate",
          criterion = eligible == FALSE | is_duplicate == TRUE)
```

---

flowchart *Render an Enrollment Flowchart*

Description

Computes counts from the pipeline, lays out nodes, and draws an EQUATOR-style enrollment diagram. This is the primary rendering function for interactive use; for saving to file with auto-sized dimensions, see `flowsave()`.

Usage

```
flowchart(.flow, engine = c("grid", "dot"), count_first = NULL, ...)

## S3 method for class 'selecta'
plot(x, engine = c("grid", "dot"), ...)
```

Arguments

`.flow`	A `selecta` object created by `enroll()` or `sources()` and populated with pipeline steps.
`engine`	Character. Rendering engine: `"grid"` (default) for base R graphics, or `"dot"` to return a Graphviz DOT string (for use with **DiagrammeR** or a locally installed executable).
`count_first`	Logical. If `TRUE`, side-box labels are rendered as `"214 Discontinued"` (bold count before label) rather than the default `"Discontinued (n = 214)"`. Applies to all box types. Default `FALSE`, supplied by the drawing routine rather than restated here, so this argument shows as `NULL` in the usage.
`...`	Additional styling and formatting arguments forwarded to the selected engine; arguments an engine does not recognize are ignored. For `engine = "grid"`: **cex, cex_side, cex_phase** Font-size multipliers for the main, side-box, and phase text **box_fill, phase_fill** Fill colors for boxes and phase strips **vpad, margin** Vertical spacing between elements and the outer margin, in inches **font_family** Font family for text **number_format** Locale-aware count formatter For `engine = "dot"`: **formatting** Label markup: `"plain"` (default) for robust, pixel-accurate centering across all fonts, or `"rich"` for HTML-like inline bold and italic that match the `grid` engine's typography at the cost of small centering drift on non-Helvetica fonts **bullets** Prefix side-box sub-reasons with a bullet; defaults on under `"plain"` (where indentation alone is a weak cue) and off under `"rich"` **font_family, padding_pt, padding_adjust** Font family (default `"Helvetica"`) and the uniform horizontal label padding in points (default 14) with its fine adjustment **ortho** Use orthogonal (right-angled) edge routing **box_fill, side_fill, border_col, arrow_col** Box, side-box, border, and arrow colors

source_fill, source_header_fill, source_header_text Source-box fill, header fill, and header text color

phase_labels, phase_fill, phase_text_col Toggle and color the phase-band labels (on by default when the flow defines phases; the `dot` engine draws them as horizontal left-margin bands rather than the `grid` engine's vertical strips)

rank_sep, node_sep Spacing of ranks and nodes, in inches

number_format Locale-aware count formatter, shared with the `grid` engine

`x` A `selecta` object.

Details

`flowchart()` is the primary rendering entry point and accepts a completed pipeline object. The `grid` engine draws the diagram to the active graphics device using the **grid** system and is intended for publication-quality figures with phase strips, precise dimensions, and locale-aware counts; the `dot` engine instead returns a Graphviz DOT-language string for prototyping or rendering through external Graphviz tooling, and draws nothing itself. Styling, font, and number-format options are forwarded to the chosen engine through `...`; options unsupported by an engine (for example the phase strips, which the `dot` engine does not draw) are ignored. `flowchart()` is normally the last call in a pipeline; for direct file output use `flowsave()`, and to size a canvas use `recdims`.

Value

For `engine = "grid"`: invisibly returns the computed graph structure (a list of `nodes`, `edges`, and `phases` data.tables). For `engine = "dot"`: returns a DOT-language string.

See Also

`flowsave` for saving to file, `recdims` for dimension recommendations, `plot.selecta` for S3 plot method

Other flowchart output functions: `flowsave()`, `print.selecta()`, `recdims()`, `summary.selecta()`

Examples

```
# Build a flow once, then render it. Most of the package's pipeline
# functions are modular and intended to be composed like this rather
# than run in isolation; see the vignettes for fuller treatments.
flow <- enroll(n = 1200) |>
  phase("Enrollment") |>
  exclude("Excluded", n = 150,
    reasons = c("Did not meet criteria" = 55,
                "Declined to participate" = 48,
                "Other reasons" = 47)) |>
  phase("Allocation") |>
  allocate(labels = c("Treatment", "Control"),
           n = c(520, 530)) |>
  phase("Analysis") |>
  endpoint("Final Analysis")

# The "dot" engine returns a Graphviz DOT string and draws nothing,
```

```
# so it runs anywhere without opening a graphics device.
dot <- flowchart(flow, engine = "dot")
substr(dot, 1, 50)

# The "grid" engine draws to the active graphics device. These calls are
# guarded with interactive() so they render in an interactive session but
# are skipped during non-interactive documentation builds, where the
# diagram cannot be sized to the page and would render incorrectly.
if (interactive()) {
  flowchart(flow)            # draws to the active device
  plot(flow)                 # plot() is a thin wrapper around flowchart()

  # Locale-aware counts: a European thousands separator.
  enroll(n = 12500) |>
    exclude("Excluded", n = 1450) |>
    endpoint("Analyzed") |>
    flowchart(number_format = "eu")
}
```

`flowsave` *Save Diagram to File*

Description

Renders the enrollment diagram and saves it to a file. Supported formats are PDF, PNG, SVG, and TIFF (inferred from the file extension). The `grid` engine renders via R graphics devices; the `dot` engine pipes Graphviz output through the system `dot` binary. Dimensions are computed automatically from diagram content via `recdims()` unless overridden.

Usage

```
flowsave(
  x,
  file,
  engine = c("grid", "dot"),
  width = NULL,
  height = NULL,
  units = c("in", "cm", "mm"),
  dpi = 300,
  sans_serif = TRUE,
  quiet = FALSE,
  ...
)
```

Arguments

`x`	A `selecta` object.

file	Character string. Output file path. The format is inferred from the file extension. Supported extensions: .pdf, .png, .svg, .tif/.tiff (all engines); .dot (dot engine only, writes the raw DOT source).
engine	Character string. One of "grid" (the default, uses R's grid graphics) or "dot" (uses the system Graphviz binary). The dot engine requires dot to be installed and on the system PATH.
width	Numeric or NULL. Width in units. If NULL (default), computed automatically. For the dot engine, omit to let Graphviz determine dimensions from layout.
height	Numeric or NULL. Height in units. If NULL (default), computed automatically. For the dot engine, omit to let Graphviz determine dimensions from layout.
units	Character string giving the units of width and height, and of the dimensions computed when either is left unspecified: "in" (inches, the default), "cm", or "mm". Graphics devices are driven in inches regardless, so the conversion is internal. Ignored by the dot engine, which takes no dimensions.
dpi	Integer. Resolution in dots per inch for raster formats (PNG, TIFF). Default 300. Honored by both engines. Mirrors the dpi argument of ggplot2::ggsave().
sans_serif	Logical. dot engine only. If TRUE (default), the rendered SVG/PDF text is displayed in a sans-serif fallback chain (Helvetica, Arial, "Liberation Sans", "DejaVu Sans",sans-serif) regardless of the layout font. Layout boxes are still sized using the metrics of the font set via font_family, so the result preserves all margins. Set to FALSE to retain the layout font as the displayed font.
quiet	Logical. Suppress the message reporting the file written and the dimensions used. The dot engine reports neither, so the setting has no effect there. Default FALSE.
...	Additional styling and formatting arguments forwarded to the selected engine; see flowchart() for the full descriptions. engine = "grid" cex, cex_side, cex_phase, box_fill, phase_fill, vpad, margin, font_family, number_format engine = "dot" formatting, bullets, count_first, number_format, ortho, font_family, padding_pt, padding_adjust, box_fill, side_fill, border_col, arrow_col, source_fill, source_header_fill, source_header_text, phase_labels, phase_fill, phase_text_col, rank_sep, node_sep

Details

flowsave() renders a flow directly to a file, inferring the format from the extension and choosing dimensions automatically unless width and height are given. With engine = "grid" it draws through R's graphics devices, producing either vector formats (.pdf, .svg) or raster formats (.png, .tiff).

For raster formats, flowsave() prefers the **ragg** device when installed, with fallback to the base png()/tiff() devices otherwise. Using these devices is generally advised for raster output over other devices such as cairo since some cairo configurations drop the plotmath italics in the count labels. The dpi argument mirrors ggplot2::ggsave() for raster resolution.

With engine = "dot", flowsave() renders a graphic based on a Graphviz DOT string: a .dot extension writes the source text directly and needs no external software, whereas image output shells out to the system dot binary and therefore requires Graphviz on the PATH.

When sizing automatically, `flowsave()` calls `recdims()` once and reuses the computed layout, so a separate `recdims()` call is unnecessary. With the `grid` engine, the file written and the dimensions used are reported through a `message()` unless `quiet = TRUE`, whether those dimensions were computed or supplied, so that a figure written at an unexpected size is apparent at the point it is written. The `dot` engine instead lets Graphviz size the output from the layout, so it reports nothing.

Value

Invisibly returns the output file path.

See Also

`flowchart` for interactive rendering, `recdims` for dimension recommendations

Other flowchart output functions: `flowchart()`, `print.selecta()`, `recdims()`, `summary.selecta()`

Examples

```
flow <- enroll(n = 500) |>
  exclude("Ineligible", n = 50) |>
  endpoint("Analysis")


# Grid engine (default). Files are written under tempdir() here so
# the example respects CRAN's no-write policy; in practice any
# desired path may be supplied.
flowsave(flow, file.path(tempdir(), "consort.pdf"))
flowsave(flow, file.path(tempdir(), "consort.png"),
         width = 8, height = 10)

# Dimensions may be given, or computed, in metric units.
flowsave(flow, file.path(tempdir(), "consort_metric.pdf"),
         width = 180, height = 240, units = "mm")

# Suppress the message reporting the file written.
flowsave(flow, file.path(tempdir(), "consort_quiet.pdf"), quiet = TRUE)



# DOT engine writing a .dot source file requires no external software.
flowsave(flow, file.path(tempdir(), "consort.dot"), engine = "dot")

# Rasterized DOT output (.svg, .png, .pdf) requires the Graphviz 'dot'
# binary on the system PATH.
if (nzchar(Sys.which("dot"))) {
  flowsave(flow, file.path(tempdir(), "consort.svg"), engine = "dot")

  # DOT engine with Times typography for serif environments.
  flowsave(flow, file.path(tempdir(), "consort_times.svg"), engine = "dot",
           font_family = "Times-Roman",
           sans_serif  = FALSE)
}
```

phase *Label a Phase of the Enrollment Flow*

Description

Adds a vertical phase label to the left margin of the diagram (*e.g.,* `"Enrollment"`, `"Allocation"`, `"Follow-up"`, `"Analysis"`). Phase labels span all subsequent steps until the next `phase()` call or the end of the flow.

Usage

```
phase(.flow, label)
```

Arguments

`.flow`	A `selecta` object.
`label`	Character string. The phase label, rendered as rotated text on the left margin.

Details

`phase()` inserts a stage boundary rather than a flow node. Each call opens a phase whose label is drawn in the left margin, spanning every subsequent step until the next `phase()` or the end of the flow. The purpose of these phase markers is to reflect the stages of analysis in the diagram; as such, they are purely presentational, and they do not alter counts or topology. In the `grid` engine, phase labels are rendered vertically and are wrapped to fit their band by default; conversely, the `dot` engine renders phase labels horizontally due to engine limitations.

Value

The updated `selecta` object with a phase marker appended.

See Also

`flowchart` for rendering with phase labels

Other flow construction functions: `assess()`, `combine()`, `endpoint()`, `enroll()`, `exclude()`, `sources()`, `stratify()`

Examples

```
# Phase labels divide a flow into labeled stages. The printed summary
# marks each phase with a "--- Label ---" banner.
enroll(n = 1200, label = "Records identified") |>
  phase("Enrollment") |>
  exclude("Duplicates", n = 84) |>
  phase("Allocation") |>
  stratify(labels = c("Drug A", "Placebo"), n = c(520, 533)) |>
```

```
  phase("Follow-up") |>
  exclude("Lost to follow-up", n = c(23, 31)) |>
  phase("Analysis") |>
  endpoint("Final Analysis")
```

`print.selecta` *Print an Enrollment Flow Summary*

Description

Displays a concise text summary of the pipeline steps and their parameters. Intended for interactive inspection of a `selecta` object before rendering.

Usage

```
## S3 method for class 'selecta'
print(x, ...)
```

Arguments

`x`	A `selecta` object.
`...`	Ignored.

Details

The `print` method gives a compact, text-only view of a `selecta` object for interactive inspection before rendering. It lists the operating mode, the starting count, and each pipeline step with its key parameters (exclusion reasons, arm labels, endpoint sub-items), and marks phase boundaries with a "— Label —" banner. It does not draw the diagram or open a graphics device; for that use `flowchart()` or `flowsave()`.

Value

Invisibly returns `x`.

See Also

`summary.selecta` for a tabular per-node summary, `flowchart` for rendering

Other flowchart output functions: `flowchart()`, `flowsave()`, `recdims()`, `summary.selecta()`

Examples

```
flow <- enroll(n = 500) |>
  exclude("Ineligible", n = 65,
    reasons = c("No consent" = 30, "Under 18" = 35)) |>
  allocate(labels = c("Drug A", "Placebo"), n = c(218, 217)) |>
  endpoint("Analyzed")
flow
```

recdims *Recommended Figure Dimensions*

Description

Computes recommended width and height in inches based on diagram content. A throwaway graphics device is opened to obtain accurate text measurements, then closed immediately.

Usage

```
recdims(
  x,
  vpad = NULL,
  pad = NULL,
  line_height = NULL,
  count_first = NULL,
  cex = NULL,
  cex_side = NULL,
  cex_phase = NULL,
  phase_width = NULL,
  margin = NULL,
  phase_multiline = NULL,
  phase_max_lines = NULL,
  font_family = NULL,
  number_format = NULL,
  units = c("in", "cm", "mm"),
  ...,
  .measure_dev = NULL,
  .return_graph = FALSE
)
```

Arguments

x	A `selecta` object.
vpad	Numeric. Vertical spacing between elements in inches. Controls the uniform gap between any box edge and the next adjacent element. Default 0.25; override globally with `options(selecta.vpad = 0.35)`.
pad	Numeric. Internal padding within boxes in inches. Default 0.08.
line_height	Numeric. Vertical line spacing in inches, controlling box heights for both main and side boxes. Scales proportionally with `cex`. Default 0.20.
count_first	Logical. If `TRUE`, side-box labels are rendered as `"214 Discontinued"` (bold count before label) rather than the default `"Discontinued (n = 214)"`. Default `FALSE`.
cex	Numeric. Font size multiplier for main box text. Default 0.85.
cex_side	Numeric. Font size multiplier for side box text. Defaults to the value of `cex`.

`cex_phase` Numeric. Font size multiplier for phase labels. Default 0.9.

`phase_width` Numeric. Width of phase label boxes in inches. Default 0.22. When `phase_multiline = TRUE` the strip is widened automatically to fit the wrapped lines, so this acts as a per-line minimum rather than a hard cap.

`margin` Numeric. Fixed margin on all four sides of the canvas in inches. Default 0.25.

`phase_multiline`
Logical. If `TRUE` (the default), a phase label longer than the vertical extent of the boxes it spans is word-wrapped across multiple stacked lines (drawn rotated in the strip), trading strip width for height so the diagram is not stretched vertically to fit a long rotated label. Set to `FALSE` to force every label onto a single line, in which case a label taller than its band stretches the diagram instead. A label that cannot be wrapped (a single word taller than its band) falls back to stretching either way. Labels containing an explicit newline (`"\n"`) are always split on it regardless of this setting. Default `TRUE`.

`phase_max_lines`
Integer. Maximum number of wrapped lines per phase label when wrapping is active; any overflow is collapsed into the final line. Default 3.

`font_family` Character. Font family used for all text in the diagram. Default `"Helvetica"`. Set to `""` to use the device default.

`number_format` Character string or two-element character vector. Locale-aware formatting for participant counts: `"us"` (default, `1,234`), `"eu"` (`1.234`), `"space"` (`1\u202F234`), `"none"` (`1234`), or a custom `c(big.mark, decimal.mark)` pair. Falls back to `getOption("selecta.number_format", "us")` when `NULL`.

`units` Character string giving the units the dimensions are returned in: `"in"` (inches, the default), `"cm"`, or `"mm"`.

`...` Additional arguments. Styling-only parameters that do not affect text measurement (such as `box_fill`, `phase_fill`, `border_col`) are silently ignored, allowing the same call signature to be shared with `flowchart()` and `flowsave()`.

`.measure_dev` Optional zero-argument function that opens a graphics device for text measurement, matching the device that will render the diagram. When `NULL` (the default) a pdf device is used. Advanced use only; see Details.

`.return_graph` Logical. If `TRUE`, attaches the pre-computed graph as an attribute for reuse by `flowsave()`. Default `FALSE`. Internal use only.

Details

`recdims()` computes the canvas size a flow needs at a given typography and layout, so the figure is neither clipped nor surrounded by excess whitespace. It lays the diagram out and measures it on a throwaway graphics device, returning width and height without drawing anything visible. Measurement is performed in inches and the result converted to `units`, with the rounding applied after conversion so the returned tenth is a tenth of the reported unit. Because text metrics are font- and device-dependent, any sizing parameter passed here (`cex`, `font_family`, `phase_multiline`, `number_format`, and so on) should match the values used at render time; styling-only parameters are ignored so the same call can be shared across `recdims()`, `flowchart()`, and `flowsave()`. A parameter left unspecified is not defaulted here but forwarded unset, so it is measured at exactly the value the drawing routine will apply. The advanced `.measure_dev` argument supplies a custom

device opener when measurement must match a non-default device. `flowsave()` calls `recdims()` internally when `width` or `height` is left unspecified, so explicit use is only needed when the dimensions themselves are wanted.

Value

A named numeric vector with elements `width` and `height`, expressed in `units` and rounded up to the nearest tenth. The units are recorded on the result as a `"units"` attribute, so a value carried between functions remains self-describing.

See Also

`flowsave` for saving to file, `flowchart` for interactive rendering

Other flowchart output functions: `flowchart()`, `flowsave()`, `print.selecta()`, `summary.selecta()`

Examples

```
flow <- enroll(n = 500) |>
  exclude("Ineligible", n = 65) |>
  allocate(labels = c("Drug A", "Placebo"), n = c(220, 215)) |>
  endpoint("Analyzed")

recdims(flow)

# Journals commonly specify figure widths in millimeters.
recdims(flow, units = "mm")
```

`selectaex0` *Simulated Observational Cohort (No Arms)*

Description

A synthetic dataset of 3,000 patients in an observational study with no treatment arms. Includes eligibility flags, exclusion reasons, and follow-up loss indicators suitable for demonstrating STROBE-style enrollment diagrams in data mode.

Usage

```
selectaex0
```

Format

A `data.table` with 3,000 rows and the following columns:

patient_id Unique patient identifier.

is_duplicate Logical. Whether the record is a duplicate.

eligible Logical. Whether the patient meets eligibility criteria.

exclusion_reason Character. Reason for exclusion, if applicable.

lost_to_followup Logical. Whether the patient was lost to follow-up.

followup_loss_reason Character. Reason for follow-up loss, if applicable.

Examples

```
data(selectaex0)
str(selectaex0)
```

`selectaex2` *Simulated Two-Arm Randomized Trial*

Description

A synthetic dataset of 2,400 patients in a two-arm randomized controlled trial. Includes screening, eligibility, treatment assignment, and discontinuation variables suitable for demonstrating CONSORT-style enrollment diagrams in data mode.

Usage

```
selectaex2
```

Format

A `data.table` with 2,400 rows and the following columns:

patient_id Unique patient identifier.

is_duplicate Logical. Whether the record is a duplicate.

eligible Logical. Whether the patient meets eligibility criteria.

exclusion_reason Character. Reason for exclusion, if applicable.

treatment Character. Treatment arm assignment (*e.g.,* `"Drug A"`, `"Placebo"`).

discontinued Logical. Whether the patient discontinued the study.

discontinuation_reason Character. Reason for discontinuation, if applicable.

Examples

```
data(selectaex2)
str(selectaex2)
table(selectaex2$treatment)
```

selectaex3 *Simulated Three-Arm Randomized Trial*

Description

A synthetic dataset of 2,400 patients in a three-arm randomized controlled trial. Structure matches `selectaex2` with an additional treatment arm.

Usage

```
selectaex3
```

Format

A `data.table` with 2,400 rows. See `selectaex2` for column descriptions.

Examples

```
data(selectaex3)
str(selectaex3)
table(selectaex3$treatment)
```

selectaex6 *Simulated Six-Arm Dose-Finding Trial*

Description

A synthetic dataset of 3,600 patients in a six-arm dose-finding trial. Structure matches `selectaex2` with six treatment arms.

Usage

```
selectaex6
```

Format

A `data.table` with 3,600 rows. See `selectaex2` for column descriptions.

Examples

```
data(selectaex6)
str(selectaex6)
table(selectaex6$treatment)
```

`sources` *Initialize a Multi-Source Flow*

Description

Entry point for flows that begin with multiple parallel identification streams, such as systematic review diagrams. Each named argument defines a source *group* (column). Individual databases or registers within each group are listed as sub-items inside a single box, mirroring the format of exclusion reasons.

Usage

```
sources(..., headers = NULL)
```

Arguments

`...`	Named integer vectors specifying sources. Each argument name identifies a group and its named elements are individual sources (*e.g.,* `databases = c("PubMed" = 1234, "Embase" = 567)`). Scalar named arguments are treated as individual sources in a single default group.
`headers`	Named character vector mapping group names to column header labels. For example, `headers = c(databases = "Databases and registers", other = "Other methods")`. If omitted, the argument names are title-cased and used as headers.

Details

`sources()` initializes a multi-source flow of the kind used in the identification stage of systematic-review diagrams (PRISMA, MOOSE), where records arrive from several origins and are pooled. Counts are supplied as named numeric values; passing named vectors instead of scalars groups the sources into labeled columns, and at most three groups are supported, matching the standard PRISMA layout. A `sources()` flow is operated in manual mode and is normally followed by `combine()` to merge the streams into a single downstream node. For a conventional single-entry study, use `enroll()` instead.

Value

An object of class `"selecta"` with a `sources` step pre-loaded. The total starting count is the sum of all source counts across all groups.

See Also

`enroll` for single-source entry, `combine` to merge parallel streams into a single flow

Other flow construction functions: `assess()`, `combine()`, `endpoint()`, `enroll()`, `exclude()`, `phase()`, `stratify()`

Examples

```
# Simple multi-source (one column, no header)
sources(PubMed = 1234, Embase = 567, CENTRAL = 89)

# Grouped sources (PRISMA two-column layout)
sources(
  databases = c("PubMed" = 1234, "Embase" = 567, "CENTRAL" = 89),
  other     = c("Citation search" = 55, "Websites" = 34)
)

# Three columns with custom headers
sources(
  previous  = c("Previous review" = 12, "Previous reports" = 15),
  databases = c("PubMed" = 1234, "Embase" = 567, "CENTRAL" = 89),
  other     = c("Citation search" = 55, "Websites" = 34),
  headers   = c(previous  = "Previous studies",
                databases = "Databases and registers",
                other     = "Other methods")
) |>
  combine("Records after deduplication") |>
  exclude("Records removed", n = 352, show_count = FALSE,
          reasons = c("Duplicates" = 340,
                      "Marked ineligible" = 12))
```

`stratify` *Split into Parallel Study Arms or Strata*

Description

Divides the enrollment flow into parallel arms. This is the primary function for splitting a population by any characteristic: treatment assignment, exposure status, diagnostic test result, etc. Subsequent `exclude()` calls apply within each arm independently. While `stratify()` is the primary function, `allocate()` is provided as a convenience alias with default label `"Randomized"`, suitable for interventional trials (CONSORT).

Usage

```
stratify(.flow, variable = NULL, labels = NULL, n = NULL, label = "Stratified")

allocate(.flow, variable = NULL, labels = NULL, n = NULL, label = "Randomized")
```

Arguments

`.flow`	A `selecta` object.
`variable`	Character string naming the column that defines the arms. Data mode only.
`labels`	A character vector of arm labels. In data mode, this can be a named vector to relabel factor levels (*e.g.,* `c(A = "Drug A", B = "Placebo")`). In manual mode, these are the arm names.

n	Integer vector. Number of participants in each arm, in the same order as labels. Manual mode only.
label	Character string for the split box. Defaults to "Stratified" for stratify() and "Randomized" for allocate().

Details

stratify() splits the flow into parallel arms, after which each exclude() (and the eventual endpoint()) applies within every arm. In data mode, variable names a column whose levels define the arms, optionally relabeled through a named labels vector; in manual mode, labels and n give the arm names and per-arm counts directly.

allocate() is an identical alias differing only in its default label ("Randomized"), provided so that interventional trials (CONSORT) read naturally; both record the same step type.

Parallel arms may later be merged with combine() to form a split-and-recombine diagram, and a flow may be split again after combining. A second stratify() or allocate() before combining produces a factorial (two-level) split, supported in both data and manual modes.

Value

The updated selecta object with a stratification step appended. All subsequent pipeline steps operate independently within each arm.

See Also

exclude for per-arm exclusions after splitting, endpoint for per-arm endpoints

Other flow construction functions: assess(), combine(), endpoint(), enroll(), exclude(), phase(), sources()

Examples

```
# Observational study (STROBE)
enroll(n = 3860) |>
  stratify(labels = c("Exposed", "Unexposed"), n = c(1900, 1960),
           label = "Classified by exposure")

# Randomized trial (CONSORT)
enroll(n = 400) |>
  allocate(labels = c("Drug A", "Placebo"), n = c(200, 200))
```

---

summary.selecta *Summarize an Enrollment Flow*

Description

Computes all counts from the pipeline and returns a data.table summarizing each node in the diagram.

Usage

```
## S3 method for class 'selecta'
summary(object, ...)
```

Arguments

object	A selecta object.
...	Ignored.

Details

The summary method runs the same count computation that underlies rendering and returns the result as a clean data.table, one row per node, rather than drawing anything. This is convenient for programmatic checks (confirming arm totals, extracting the final analyzed count) and for embedding flow figures in tables or reports. The returned object is a plain data.table and may be filtered or joined like any other. For a human-readable console view use print.selecta(); to render the diagram use flowchart().

Value

A data.table with columns phase, role, arm, text, and n. Each row corresponds to one node in the computed diagram.

See Also

print.selecta for a console summary, flowchart for rendering

Other flowchart output functions: flowchart(), flowsave(), print.selecta(), recdims()

Examples

```
flow <- enroll(n = 500) |>
  exclude("Ineligible", n = 65) |>
  allocate(labels = c("Drug A", "Placebo"), n = c(218, 217)) |>
  endpoint("Analyzed")
summary(flow)
```

Index

∗ **cohort extraction functions**
  cohort, 4
  cohorts, 5
∗ **datasets**
  selectaex0, 23
  selectaex2, 24
  selectaex3, 25
  selectaex6, 25
∗ **flow construction functions**
  assess, 2
  combine, 6
  endpoint, 7
  enroll, 9
  exclude, 10
  phase, 19
  sources, 26
  stratify, 27
∗ **flowchart output functions**
  flowchart, 13
  flowsave, 16
  print.selecta, 20
  recdims, 21
  summary.selecta, 28

allocate *(*stratify*)*, 27
assess, 2, *8*, *12*
assess(), *7, 8*, *10*, *12*, *19*, *26*, *28*

cohort, 4, *5*
cohort(), *5*
cohorts, *4*, *5*
cohorts(), *4*
combine, 6, *26*
combine(), *3*, *8*, *10*, *12*, *19*, *26*, *28*

endpoint, *3*, 7, *28*
endpoint(), *3*, *7*, *10*, *12*, *19*, *26*, *28*
enroll, *4*, 9, *12*, *26*
enroll(), *3*, *7, 8*, *12*, *19*, *26*, *28*
exclude, *3*, 10, *10*, *28*
exclude(), *3*, *7, 8*, *10*, *19*, *26*, *28*

flowchart, *8*, *10*, 13, *18–20*, *23*, *29*
flowchart(), *18*, *20*, *23*, *29*
flowsave, *15*, 16, *23*
flowsave(), *15*, *20*, *23*, *29*

phase, 19
phase(), *3*, *7, 8*, *10*, *12*, *26*, *28*
plot.selecta, *15*
plot.selecta *(*flowchart*)*, 13
print.selecta, 20, *29*
print.selecta(), *15*, *18*, *23*, *29*

recdims, *15*, *18*, 21
recdims(), *15*, *18*, *20*, *29*

selectaex0, 23
selectaex2, 24, *25*
selectaex3, 25
selectaex6, 25
sources, *7*, *10*, 26
sources(), *3*, *7, 8*, *10*, *12*, *19*, *28*
stratify, *7*, 27
stratify(), *3*, *7, 8*, *10*, *12*, *19*, *26*
summary.selecta, *20*, 28
summary.selecta(), *15*, *18*, *20*, *23*