

pgfplots-autonode

Collision-aware automatic labels for PGFPlots curves

Christophe Jorssen
christophe.jorssen@gmail.com

Version 1.0.0 — 2026-09-20

Abstract

`pgfplots-autonode` is a format-generic LuaTeX library that measures TeX labels, samples visible PGFPlots paths in canvas coordinates, and chooses a joint collision-aware placement at the end of each axis. It supports linear and logarithmic axes, sloped labels, axis-boundary constraints, priorities, deterministic bounded algorithms, and detailed diagnostics. It is independent of `luacoolprop` and of any application-specific curve generator.

Contents

1	Scope, requirements, and independence	2
2	Tutorial: why joint placement matters	2
2.1	The fixed-position failure	2
2.2	Register first, solve once	3
2.3	Algorithms and bounded work	4
2.4	Seeing what the solver measures	4
3	Case study: a dense thermodynamic chart	5
4	Installation and loading	6
I	Complete reference	6
5	The <code>pgfplots-autonode</code> library	6
5.1	Why placement is deferred	7
5.2	Loading the library	7
5.3	Registering a curve label	7
6	Per-label autonode keys	8
6.1	Candidate positions	8
6.2	Geometry and node appearance	10
6.3	Cost and priority	11
7	Axis-level autonode keys	12
7.1	Lifecycle and solver selection	12
7.2	Collision model and failure policy	13
7.3	Axis defaults for candidate generation	14
7.4	Diagnostics	14

8	Choosing a policy	16
8.1	Sampling and the plot boundary	16
II	Implementation and maintenance	16
9	Architecture	16
10	Lua backend contract	17
11	Independent verification	17

1 Scope, requirements, and independence

The library requires LuaTeX and PGFPlots 1.18 or later. It works with LuaLaTeX, plain LuaTeX, and ConTeXt MkIV. Any PGFPlots curve that supports trailing path commands can use the public `\pgfplotsautonode` interface.

AI-assisted “vibe coding”

This package has been vibe-coded with ChatGPT. Substantial parts of its design, implementation, documentation, and tests were generated or revised with AI assistance. AI-generated code can appear convincing while remaining incorrect or incomplete. Inspect the source and validate the resulting label layout for the intended publication.

Copyright © 2026 Christophe Jorssen. The package is maintained under the LaTeX Project Public License 1.3c or later. The Current Maintainer is Christophe Jorssen (christophe.jorssen@gmail.com). The project repository and issue tracker are hosted at github.com/cjorssen/pgfplots-autonode.

2 Tutorial: why joint placement matters

Consider two families of affine functions, one with slope (+1) and one with slope (-1). Integer vertical offsets make a regular, dense crossing network. It is simple enough that every geometrical conflict is visible, yet it exercises the same placement problem as contour maps, spectra, and engineering charts.

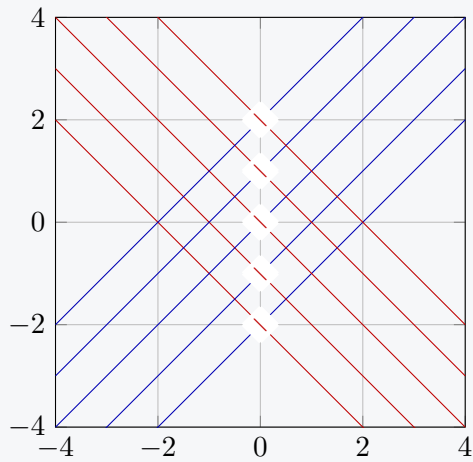
2.1 The fixed-position failure

If every label is placed at the midpoint of its own curve, many nodes collect near crossings. Each plot knows nothing about labels already chosen for other plots, so local `node[pos=.5]` instructions cannot solve the global assignment problem.

Midpoint labels collide in an affine network

```
\begin{tikzpicture}
\begin{axis}[width=11cm,height=7cm,xmin=-4,xmax=4,ymin=-4,ymax=4,
axis equal image,grid=both]
\foreach \b in {-2,...,2}{
\addplot[blue!70!black,domain=-4:4,samples=2] {x+\b}
node[pos=.5,sloped,fill=white,inner sep=1pt] {$+$};
\addplot[red!75!black,domain=-4:4,samples=2] {-x+\b}
node[pos=.5,sloped,fill=white,inner sep=1pt] {$-$};
}
}
```

```
\end{axis}
\end{tikzpicture}
```

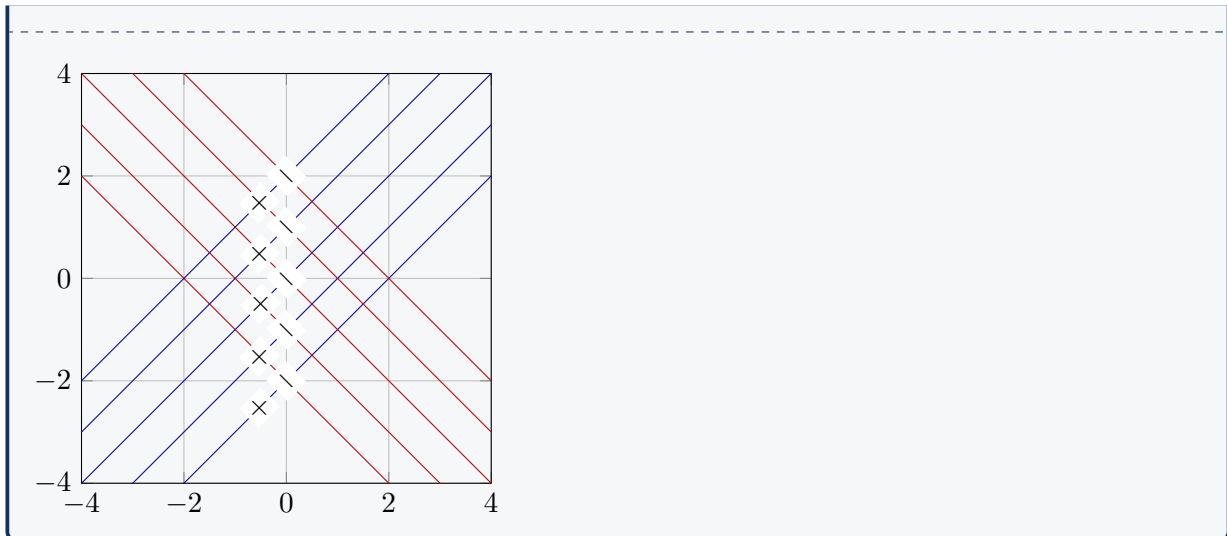


2.2 Register first, solve once

Enable auto node placement on the axis and attach one `\pgfplotsautonode` command to each plot. Registration is cheap: final node drawing is deferred until PGFPlots has established the axis transformation and the solver has seen every label.

The same network solved jointly

```
\begin{tikzpicture}
\begin{axis}[width=11cm,height=7cm,xmin=-4,xmax=4,ymin=-4,ymax=4,
axis equal image,grid=both,auto node placement,
auto node algorithm=repair,auto node bbox mode=oriented,
auto node failure mode=error,auto node candidates=81]
\foreach \b in {-2,...,2}{
\addplot[blue!70!black,domain=-4:4,samples=2] {x+\b}
\pgfplotsautonode[preferred pos=.5,sloped=true]{$+}$};
\addplot[red!75!black,domain=-4:4,samples=2] {-x+\b}
\pgfplotsautonode[preferred pos=.5,sloped=true]{$-$};
}
\end{axis}
\end{tikzpicture}
```



The preferred midpoint remains a soft objective. The solver may move a label along its visible curve to satisfy harder requirements: remaining inside the plot rectangle and avoiding other measured label boxes.

2.3 Algorithms and bounded work

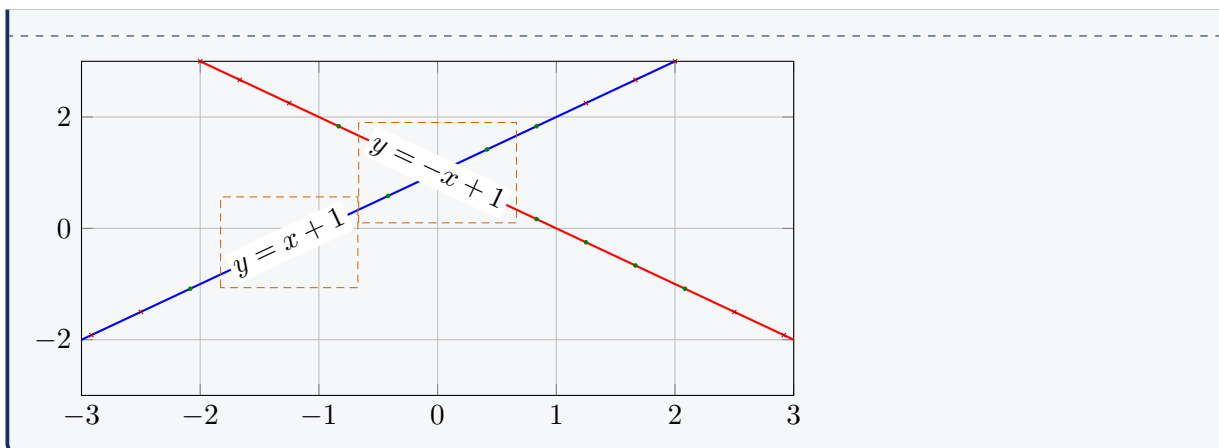
Use **greedy** for a quick deterministic draft, **repair** as the balanced default, **local-search** for additional bounded improvement, and **exact-small** only for genuinely small label sets. Candidate counts, iteration limits, and exact-search state limits make the computational budget explicit and reproducible. The complete reference states the fallback and failure semantics; no heuristic is described as proving a global optimum.

2.4 Seeing what the solver measures

The diagnostic overlay below exposes candidate points, rejected candidates, and the final measured boxes. It is intended for authoring, not final artwork.

Candidate and bounding-box diagnostics

```
\begin{tikzpicture}
\begin{axis}[width=11cm,height=6cm,xmin=-3,xmax=3,ymin=-3,ymax=3,
grid=both,auto node placement,auto node candidates=13,
auto node show candidates=true,
auto node show rejected candidates=true,
auto node show bounding boxes=true]
\addplot[blue,thick,domain=-3:3,samples=2] {x+1}
\pgfplotsautonode[preferred pos=.5,sloped=true]{$y=x+1$};
\addplot[red,thick,domain=-3:3,samples=2] {-x+1}
\pgfplotsautonode[preferred pos=.5,sloped=true]{$y=-x+1$};
\end{axis}
\end{tikzpicture}
```



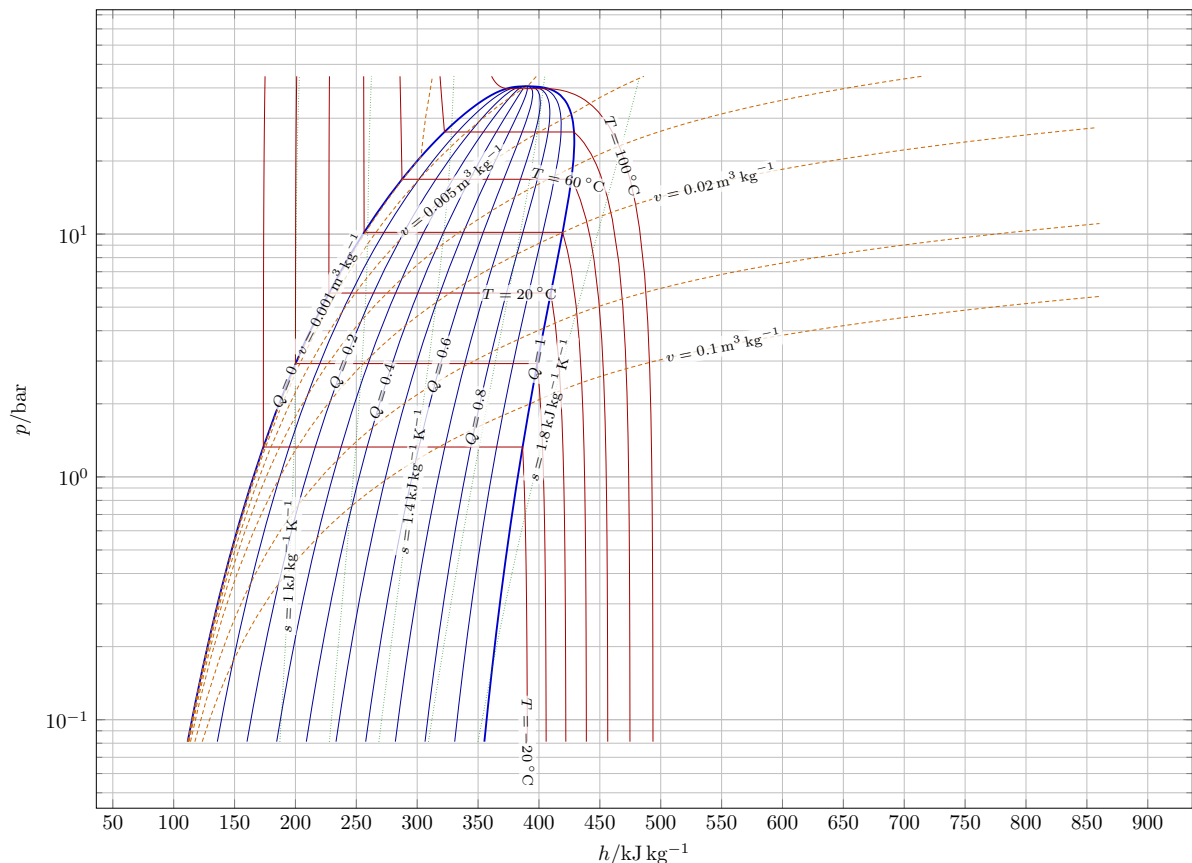
3 Case study: a dense thermodynamic chart

The `luacoolprop` package is a concrete user of `pgfplots-autonode`. It obtains thermodynamic properties from CoolProp, constructs PGFPlots paths, and registers the requested isoline labels through `\pgfplotsautonode`. The placement library then sees labels from all families together; it does not need to know what an isotherm, isentrope, isochore, or quality curve means.

The following example is extracted from the LuaCoolProp manual. Eleven quality curves, seven isotherms, five isentropes, and seven isochores compete for space near the saturation dome. Their preferred positions alone would put several long, sloped labels in the same region. Joint placement measures the actual TeX nodes, searches the visible parts of every path, and keeps the result inside the axis rectangle.

LuaCoolProp client code for a dense PH diagram

```
\begin{tikzpicture}
\begin{axis}[
  lcp fluid=R134a,
  xlabel={\$/\rm kJ\,kg^{-1}}$,
  ylabel={\$/\rm bar}$,
  ymode=log,grid=both,
  auto node placement,
  auto node algorithm=repair,
  auto node bbox mode=oriented,
  auto node failure mode=error,
  width=20cm,height=15cm]
\LCAddPHQuality[
  quality values={0,0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9,1},
  labels=true,sloped labels=true]
\LCAddPHIsotherms[
  temperature values={-20,0,20,40,60,80,100},
  labels=true,sloped labels=true]
\LCAddPHIsentropes[
  entropy values={1.0,1.2,1.4,1.6,1.8},
  labels=true,sloped labels=true]
\LCAddPHIsochores[
  specific volume values={0.001,0.002,0.005,0.01,0.02,0.05,0.1},
  labels=true,sloped labels=true]
\end{axis}
\end{tikzpicture}
```



Dense R134a PH chart generated by LuaCoolProp. Automatic labels are measured and placed jointly by pgfplots-autonode.

This is an application example rather than part of the affine tutorial: the prebuilt vector figure keeps the present manual self-contained, while the complete executable source and its thermodynamic discussion remain in the LuaCoolProp manual.

4 Installation and loading

Install the CTAN package with the package manager of the TeX distribution. A manual TDS installation places the generic PGFPlots library below `tex/generic/pgfplots-autonode`, the Lua module below `tex/luatex/pgfplots-autonode`, and the optional LaTeX wrapper below `tex/latex/pgfplots-autonode`.

In LuaLaTeX, `\usepackage{pgfplots-autonode}` loads PGFPlots and the library. The format-neutral spelling is `\usepgfplotslibrary{autonode}` after loading the appropriate PGFPlots frontend. The complete examples shipped with the package cover LuaLaTeX, plain LuaTeX, and ConTeXt MkIV.

Part I

Complete reference

5 The pgfplots-autonode library

`pgfplots-autonode` is an independent, format-generic PGFPlots library. Any PGFPlots curve can register a label, expose candidate positions, and let the library choose a joint placement after

the axis has been surveyed. It has no dependency on `luacoolprop`, `CoolProp`, or thermodynamic data.

5.1 Why placement is deferred

PGFPlots first surveys plots and only then knows the final axis transformation. Autonode samples each labelled plot in physical canvas coordinates, measures the actual TeX label box, records the visible axis rectangle, and solves the multi-label assignment at the end of the axis. Consequently, collision tests remain meaningful with logarithmic axes, unequal scales, resized figures, and sloped labels.

Two kinds of option

Axis keys such as `auto node algorithm` configure the joint solver. Keys under `/pgfplots/autonode`, passed to `\pgfplotsautonode`, configure one label. This separation is essential: one axis has one solver policy, while each label has its own candidate range, box, priority, and preferred position.

5.2 Loading the library

`\usepgfplotslibrary{<autonode>}`

Loads the generic PGFPlots library after the format's PGFPlots frontend. The command works in LaTeX, plain TeX, and ConTeXt. The LaTeX convenience package `pgfplots-autonode` loads PGFPlots and this library automatically.

Standalone LaTeX loading

```
\usepackage{pgfplots}
\pgfplotsset{compat=1.18}
\usepgfplotslibrary{autonode}
```

Standalone plain LuaTeX loading

```
\input pgfplots.tex
\pgfplotsset{compat=1.18}
\usepgfplotslibrary{autonode}
```

Standalone ConTeXt loading

```
\usemodule[t]{pgfplots}
\usepgfplotslibrary{autonode}
```

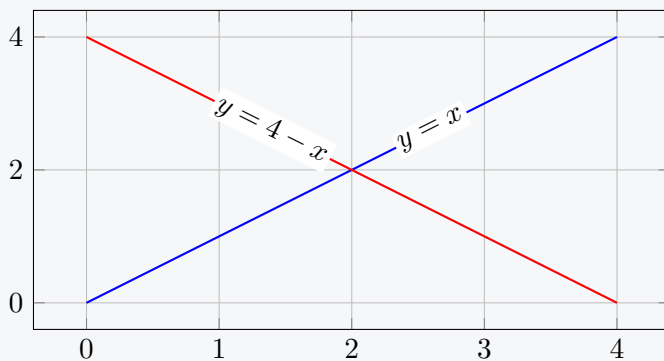
5.3 Registering a curve label

`\pgfplotsautonode[<label options>]{<text>}`

Registers `<text>` as a trailing path command for the immediately preceding `\addplot`. The bracket argument is syntactically required; use empty brackets when no per-label option is needed. End the complete plot statement with one semicolon, after this command.

Two intersecting curves, solved together

```
\begin{tikzpicture}
\begin{axis}[width=10cm,height=5.8cm,grid=both,
  auto node placement,
  auto node algorithm=repair,
  auto node failure mode=allow-minimal-overlap]
\addplot[blue,thick,domain=0:4,samples=40] {x}
  \pgfplotsautonode[preferred pos=.65,sloped=true]{$y=x$};
\addplot[red,thick,domain=0:4,samples=40] {4-x}
  \pgfplotsautonode[preferred pos=.35,sloped=true]{$y=4-x$};
\end{axis}
\end{tikzpicture}
```



Trailing-command syntax

Attach `\pgfplotsautonode` before the plot's final semicolon. A semicolon placed earlier closes the path too soon. The bracket pair is also mandatory, even when the label has no options.

6 Per-label autonode keys

The keys in this section belong to `/pgfplots/autonode` and appear in the optional argument of `\pgfplotsautonode`.

6.1 Candidate positions

preferred pos=*<fraction>*

Default: 0.5

Sets the desired position along the plot. It is a preference, not a fixed position: the solver may move the label to prevent a collision.

Example: preferred pos=0.7.

min pos=*<fraction>*

Default: 0

Excludes candidates before this fraction of the complete plot path.

Example: min pos=0.15.

max pos= $\langle$ *fraction* $\rangle$

Default: 1

Excludes candidates after this fraction of the complete plot path.

Example: `max pos=0.85`.

samples= $\langle$ *positive integer* $\rangle$

Default: axis default, initially 51

Sets the number of candidate positions sampled for this label. More samples improve freedom but increase solving work.

Example: `samples=31`.

Accepted alias: `candidates` forwards to `samples`.

candidate strategy= $\langle$ *choice* $\rangle$

Default: around-preferred

Controls the order and distribution of candidate fractions on the portions of the plotted path inside the visible axis rectangle. The fractions supplied by `preferred pos`, `min pos`, and `max pos` still refer to the complete PGFPlots path: the library intersects that range with the visible path, preserves disconnected pieces, and distributes the requested samples over their combined visible length. A curve wholly outside the rectangle produces no label.

uniform Samples the permitted interval uniformly.

around-preferred Starts near the preferred position and expands outwards.

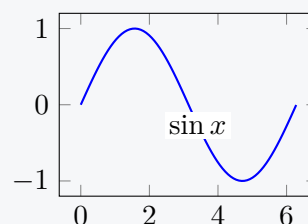
adaptive Uses a deterministic dense neighbourhood around the preferred position plus coarse coverage of the complete interval.

The spaced spelling `around preferred` is accepted as an equivalent choice.

Example: `candidate strategy=adaptive`.

Restricting the search interval

```
\begin{tikzpicture}
\begin{axis}[width=5cm,height=4cm,
auto node placement,domain=0:6.28]
\addplot[blue,thick,samples=80] {sin(
deg(x))}
\pgfplotsautonode[
preferred pos=.55,min pos=.35,max
pos=.8,
samples=41,candidate strategy=
adaptive]{$\sin x$};
\end{axis}
\end{tikzpicture}
```



6.2 Geometry and node appearance

normal shift= $\langle \textit{TeX dimension} \rangle$

Default: 0pt

Moves the label perpendicular to the sampled curve tangent. Positive and negative values select opposite sides.

Example: `normal shift=6pt.`

clearance= $\langle \textit{TeX dimension} \rangle$

Default: 1pt

Adds an invisible safety margin used by collision tests. Increase it for visually airy layouts or decorated nodes.

Example: `clearance=2pt.`

inner sep= $\langle \textit{TeX dimension} \rangle$

Default: 1.5pt

Sets the node's inner padding for both measurement and drawing. A later **inner sep** in **node options** overrides it in both places.

Example: `inner sep=2pt.`

font= $\langle \textit{TeX font commands} \rangle$

Default: empty

Sets the font both while measuring and while drawing the label.

Example: `font={\scriptsize \bfseries }.`

node options= $\langle \textit{TikZ node options} \rangle$

Default: white fill, no draw

Sets the node appearance used in both the scratch measurement and final drawing. Anchors, font, padding, and text width affect the measured box; decorations or effects outside the node's PGF bounding box should be paired with sufficient **clearance**.

Example: `node options={fill=yellow!20,draw=orange,rounded corners}.`

sloped= $\langle \textit{boolean} \rangle$

Default: false

Rotates the label along the local curve tangent.

Example: `sloped=true.`

allow upside down= $\langle \textit{boolean} \rangle$

Default: false

When false, normalises sloped text to remain readable; when true, preserves the tangent orientation even if inverted.

Example: `allow upside down=true.`

`above=<style>`

Default: not applied

Convenience style equivalent to `normal shift=6pt`.

Example: `\pgfplotsautonode [above]{label}`.

`below=<style>`

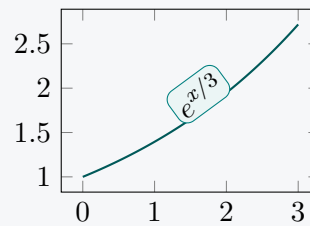
Default: not applied

Convenience style equivalent to `normal shift=-6pt`.

Example: `\pgfplotsautonode [below]{label}`.

A styled, shifted label

```
\begin{tikzpicture}
\begin{axis}[width=5cm,height=4cm,
auto node placement,domain=0:3]
\addplot[teal!70!black,thick,samples
=50] {exp(x/3)}
\pgfplotsautonode[preferred pos=.55,
above,
sloped=true,
inner sep=2pt,clearance=2pt,
node options={fill=teal!8,draw=teal
,
rounded corners,inner sep=2pt}]{
 $e^{x/3}$ };
\end{axis}
\end{tikzpicture}
```



6.3 Cost and priority

`preferred weight=<nonnegative number>`

Default: 8

Weights displacement from `preferred pos`. Raise it when remaining near the semantic location matters more than other soft costs.

Example: `preferred weight=15`.

`overlap weight=<nonnegative number>`

Default: 1000

Weights overlap area during repair and local search. A conflicting pair uses the arithmetic mean of its two labels' weights, so the objective is symmetric. The large default makes avoiding collisions far more important than matching the preferred position.

Example: `overlap weight=2000`.

`priority=<number>`

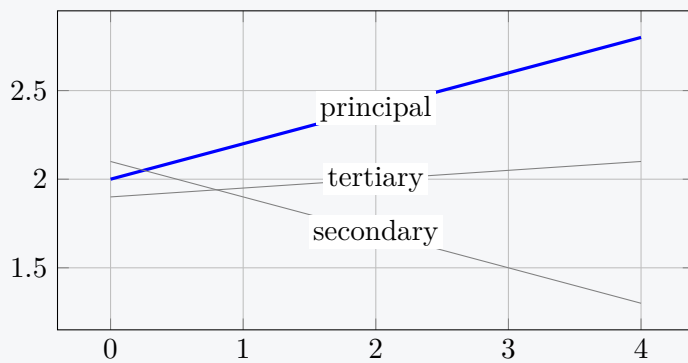
Default: 0

Orders labels when not all can be placed. Larger values are protected first; low-priority labels are hidden first under the corresponding failure policy.

Example: `priority=10`.

Prioritising the principal curve

```
\begin{tikzpicture}
\begin{axis}[width=10cm,height=5.8cm,grid=both,
  auto node placement,auto node algorithm=repair,
  auto node failure mode=hide-low-priority]
\addplot[blue,very thick,domain=0:4] {2+.2*x}
  \pgfplotsautonode[priority=10,preferred pos=.5] {principal};
\addplot[gray,domain=0:4] {2.1-.2*x}
  \pgfplotsautonode[priority=0,preferred pos=.5] {secondary};
\addplot[gray,domain=0:4] {1.9+.05*x}
  \pgfplotsautonode[priority=0,preferred pos=.5] {tertiary};
\end{axis}
\end{tikzpicture}
```



7 Axis-level autonode keys

These keys belong to `/pgfplots`. They are placed in an axis option list or set with `\pgfplotsset`. Except for diagnostic drawing, changes affect only solving; they do not change curve geometry.

7.1 Lifecycle and solver selection

auto node placement=*<style>* Default: not applied by standalone PGFPlots

Resets the label registry when the key is parsed and installs the end-axis solve/render hook. This makes consecutive axes independent. It is mandatory on every axis containing `\pgfplotsautonode` registrations. Merely loading the library deliberately does not modify unrelated PGFPlots axes. Registering a label without this lifecycle key emits a warning instead of silently dropping the label.

Example: `\begin{axis}[auto node placement]`.

auto node algorithm=*<choice>* Default: **repair**

Selects the assignment algorithm.

greedy Fast, deterministic first-fit placement; useful for drafts and large label sets.

repair Greedy placement followed by bounded conflict repair; the balanced default.

local-search Iteratively improves a complete assignment up to the iteration budget.

exact-small Searches assignments for small label sets, subject to the exact limits below.

Example: `auto node algorithm=local-search.`

`auto node max iterations=<positive integer>`

Default: 40

Bounds repair/local-search work. Increasing it can improve dense axes at a predictable runtime cost.

Example: `auto node max iterations=80.`

`auto node exact max labels=<positive integer>`

Default: 10

Limits how many labels may enter `exact-small`; larger problems fall back to a bounded strategy.

Example: `auto node exact max labels=8.`

`auto node exact max states=<positive integer>`

Default: 50000

Caps the assignment states examined by `exact-small`.

Example: `auto node exact max states=100000.`

7.2 Collision model and failure policy

`auto node bbox mode=<choice>`

Default: axis-aligned

Selects the collision box model.

axis-aligned Uses axis-aligned rectangles; fast and conservative for rotated text.

oriented Uses oriented boxes for sloped labels; more precise and more expensive.

Example: `auto node bbox mode=oriented.`

`auto node failure mode=<choice>`

Default: warn

Defines the result when the chosen assignment still contains an unacceptable label-to-label conflict. With the default `auto node allow outside=false`, the plot border remains a hard constraint in every mode: a label with no measured candidate box wholly inside the plot rectangle is hidden with a warning, or causes a compilation error in `error` mode. No overlap policy draws a rejected outside candidate.

warn Keeps unresolved label-to-label overlaps and emits a concise warning; labels with no inside candidate are hidden.

error Stops compilation; appropriate for strict publication builds.

hide-low-priority Hides the least important labels until remaining conflicts are removed.

allow-minimal-overlap Keeps the complete assignment improved by the selected bounded heuristic without treating residual overlap as fatal; it does not claim a global optimum.

Example: `auto node failure mode=hide-low-priority.`

`auto node border margin=⟨TeX dimension⟩` Default: 2pt

Reserves space between label boxes and the visible plot rectangle.

Example: `auto node border margin=4pt.`

`auto node overlap tolerance=⟨TeX dimension⟩` Default: 0.2pt

Ignores microscopic intersection caused by rounding or touching edges.

Example: `auto node overlap tolerance=0.5pt.`

`auto node allow outside=⟨boolean⟩` Default: false

Allows label boxes to extend beyond the plot rectangle. Tick labels and neighbouring figures are not obstacle-aware, so enable this deliberately.

Example: `auto node allow outside=true.`

7.3 Axis defaults for candidate generation

`auto node candidates=⟨positive integer⟩` Default: 51

Sets the default sample count for labels which do not override `samples`.

Example: `auto node candidates=31.`

`auto node candidate strategy=⟨choice⟩` Default: around-preferred

Sets the default per-label strategy.

uniform Uniform interval coverage.

around-preferred Preferred-first expanding order.

adaptive Deterministically combines dense preferred-position sampling with coarse full-interval coverage.

The spelling `around preferred` is accepted too.

Example: `auto node candidate strategy=uniform.`

Numeric controls are validated before solving. Positions must lie in $[0, 1]$, counts and iteration limits must be positive integers, and sizes, clearances, margins, tolerances, and weights must be nonnegative. Invalid Lua backend input emits a warning and uses the documented default; invalid PGF key choices are rejected by PGF's choice handler.

7.4 Diagnostics

`auto node debug=⟨choice⟩` Default: false

Controls textual diagnostics. Bare `auto node debug` means `summary`.

false No diagnostic output.

off Synonym for `false`.

summary Reports configuration and solve outcome.

verbose Also reports labels and assignments.

trace Reports candidate-level details; output can be large.

Example: `auto node debug=verbose.`

auto node show bounding boxes=*(boolean)*

Default: false

Draws the final measured collision rectangles.

Example: `auto node show bounding boxes=true.`

auto node show candidates=*(boolean)*

Default: false

Draws valid sampled candidate locations.

Example: `auto node show candidates=true.`

auto node show rejected candidates=*(boolean)*

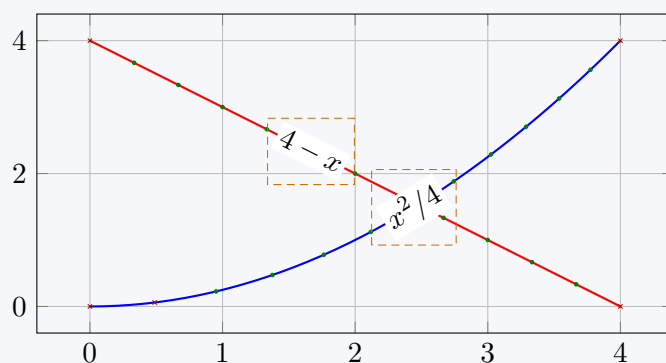
Default: false

Also marks rejected candidates, which is useful when border constraints or overlaps leave few choices.

Example: `auto node show rejected candidates=true.`

Visualising candidates and final boxes

```
\begin{tikzpicture}
\begin{axis}[width=10cm,height=5.8cm,grid=both,
  auto node placement,auto node candidates=13,
  auto node show candidates=true,
  auto node show rejected candidates=true,
  auto node show bounding boxes=true]
\addplot[blue,thick,domain=0:4,samples=40] {x^2/4}
\pgfplotsautonode[preferred pos=.5,sloped=true]{$x^2/4$};
\addplot[red,thick,domain=0:4,samples=40] {4-x}
\pgfplotsautonode[preferred pos=.5,sloped=true]{$4-x$};
\end{axis}
\end{tikzpicture}
```



8 Choosing a policy

For a first diagram, use the defaults: `repair`, axis-aligned boxes, 51 candidates, and warnings. For final artwork with many sloped labels, try `auto node bbox mode=oriented` and increase candidates before increasing iterations. Use priorities plus `hide-low-priority` where a dense network must remain readable. Reserve `exact-small` for genuinely small label sets and a controlled build budget.

The bounded repair and local-search algorithms are deterministic heuristics. They improve the weighted assignment but do not certify a global minimum. Likewise, the candidate strategy named `adaptive` adapts its sampling density to the preferred path neighbourhood; it does not perform a second feedback pass after observing collisions.

Node geometry is measured by typesetting each label once in a scratch TikZ picture before solving. The solver uses the resulting bounds relative to the node anchor, including asymmetric anchors and TikZ padding; sloped bounds are then rotated with the path tangent. Avoid label contents with non-idempotent side effects, because the contents are also typeset when the final node is drawn.

8.1 Sampling and the plot boundary

Autonode samples candidate positions only on the visible pieces of an already surveyed PGFPlots path. This avoids label candidates on clipped portions; it does not discard source-curve points or change the path handed to PGFPlots. That distinction protects curve crossings at the axis border, named-path intersections, disconnected valid domains, and the meaning of a fixed `preferred pos`. Axis limits and clipping may be independent of the parameter used by a plot expression, and neither coordinate need vary monotonically along a curve. A curve-producing package may safely avoid unnecessary samples when it knows its own mathematical domain; autonode does not guess or alter that domain after the fact.

The visual diagnostic keys are intended for authoring only. A reproducible release figure should normally disable them and choose either `auto node failure mode=error` for a hard quality gate or a documented fallback such as `hide-low-priority`.

Part II

Implementation and maintenance

9 Architecture

File	Responsibility
<code>pgflibrarypgfplots.autonode.code.tex</code>	PGFPlots keys, axis lifecycle, path sampling, label measurement, and final TikZ nodes
<code>pgfplots-autonode.lua</code>	candidate validation, canvas geometry, collision costs, bounded assignment algorithms, and diagnostic emission
<code>pgfplots-autonode.sty</code>	optional LuaLaTeX package wrapper
<code>pgfplots-autonode.code.tex</code>	format-neutral direct loader for users who already loaded a PGFPlots frontend

TeX owns all token lists and typesets each label once in a scratch picture to obtain its actual node bounds. Lua receives only numbers, booleans, and stable label identifiers. Candidate positions are sampled on the visible pieces of the surveyed path in final canvas coordinates. Disconnected visible pieces remain distinct, and candidates outside the plot rectangle are rejected unless `auto node allow outside=true` is explicit.

10 Lua backend contract

The module exposes small groups of functions:

- `reset` and `configure` manage one solve;
- `set_axis_rect`, `add_label`, and `set_label_geometry` register measured input;
- `add_candidate` and `emit_candidate_positions` manage samples; and
- `solve` and `solve_and_emit` assign labels.

Public boundaries validate finite numeric values, integer limits, intervals, and enumerated modes. Module state is local; the TeX axis lifecycle resets it before each independent solve. Detailed API comments live next to the implementation in `pgfplots-autonode.lua`.

11 Independent verification

The release test installs only the `pgfplots-autonode` TDS archive in a temporary TEXMF tree. It runs the numerical Lua suite and compiles LuaLaTeX, plain LuaTeX, and ConTeXt fixtures while rejecting any runtime reference to `luacoolprop`. Additional fixtures cover missing lifecycle diagnostics, successive axes, measured asymmetric nodes, clipped paths, disconnected visible domains, and hard axis-border constraints.

Command Index

`\pgfplotsautonode`, 6

`\usepgfplotslibrary`, 6

Key and Value Index

above, 9
allow upside down, 9
auto node algorithm, 11
auto node algorithm=exact-small, 11
auto node algorithm=greedy, 11
auto node algorithm=local-search, 11
auto node algorithm=repair, 11
auto node allow outside, 13
auto node bbox mode, 12
auto node bbox mode=axis-aligned, 12
auto node bbox mode=oriented, 12
auto node border margin, 13
auto node candidate strategy, 13
auto node candidate
 strategy=adaptive, 13
auto node candidate strategy=around
 preferred, 13
auto node candidate
 strategy=around-preferred, 13
auto node candidate strategy=uniform,
 13
auto node candidates, 13
auto node debug, 13
auto node debug=false, 13
auto node debug=off, 13
auto node debug=summary, 13
auto node debug=trace, 14
auto node debug=verbose, 14
auto node exact max labels, 12
auto node exact max states, 12
auto node failure mode, 12
auto node failure
 mode=allow-minimal-overlap, 12
auto node failure mode=error, 12
auto node failure
 mode=hide-low-priority, 12
auto node failure mode=warn, 12
auto node max iterations, 12
auto node overlap tolerance, 13
auto node placement, 11
auto node show bounding boxes, 14
auto node show candidates, 14
auto node show rejected candidates, 14

below, 10

candidate strategy, 8
candidate strategy=adaptive, 8
candidate strategy=around preferred, 8
candidate strategy=around-preferred, 8
candidate strategy=uniform, 8
candidates (alias), 8
clearance, 9

font, 9

inner sep, 9

max pos, 7
min pos, 7

node options, 9
normal shift, 9

overlap weight, 10

preferred pos, 7
preferred weight, 10
priority, 10

samples, 8
sloped, 9