Modes for Handling Invalid Data

Edit this page

This page discusses modes in Vega-Lite for handling invalid data (`null` and `NaN` in continuous scales).

Note: Vega-Lite does *not* consider `null` and `NaN` in categorical scales and text encodings as invalid data:

- Categorical scales can treat nulls and NaNs as separate categories
- Similarly, text encodings can directly display nulls and NaNs.

Documentation Overview

Mark Invalid Mode

You can set the invalid data mode via `mark.invalid` (or `config.mark.invalid`) to configure how Vega-Lite handles invalid data (`null` and `NaN` in continuous scales).

Property	Type	Description
invalid	String | Null	Invalid data mode for marks, which defines how the visualization should represent invalid values (`null` and `NaN` in continuous scales without defined output for invalid values) in the marks and their scale domains. • `"filter"` — *Exclude* all invalid values from the visualization's *marks* and *scales*. For path marks (for line, area, trail), this option will create paths that connect valid points, as if the data rows with invalid values do not exist. • `"break-paths-filter-domains"` — Break path marks (for line, area, trail) at invalid values. For non-path marks, this is equivalent to `"filter"`. All *scale* domains will *exclude* these filtered data points. • `"break-paths-show-domains"` — Break paths (for line, area, trail) at invalid values. Hide invalid values for non-path marks. All *scale* domains will *include* these filtered data points. • `"show"` or `null` — Show all data points in the marks and scale domains. Each scale will use the output for invalid values

Property	Type	Description
		defined in `config.scale.invalid` or, if unspecified, by default invalid values will produce the same visual values as zero (if the scale includes zero) or the minimum value (if the scale does not include zero). • `"break-paths-and-show-path-domains"` (default) — This is equivalent to `"break-path-keep-domains"` for path-based marks (line/area/trail) and `"filter"` for other marks. **Note**: If any channel's scale has an output for invalid values defined in `config.scale.invalid`, all values for the scales will be considered "valid" since they can produce a reasonable output for the scales. Thus, fields for such channels will not be filtered and will not cause path breaks.

Examples

To understand how these modes affect common marks, see these examples below, which visualize this dataset:

```
[
  {"a": null, "b": 100},
  {"a": -10, "b": null},
  {"a": -5, "b": -25},
  {"a": -1, "b": -20},
  {"a": 0, "b": null},
  {"a": 1, "b": 30},
  {"a": 5, "b": 40},
  {"a": 10, "b": null}
]
```

by assigning `"a"` to x-axis (as quantitative and ordinal fields) and `"b"` to y-axis.

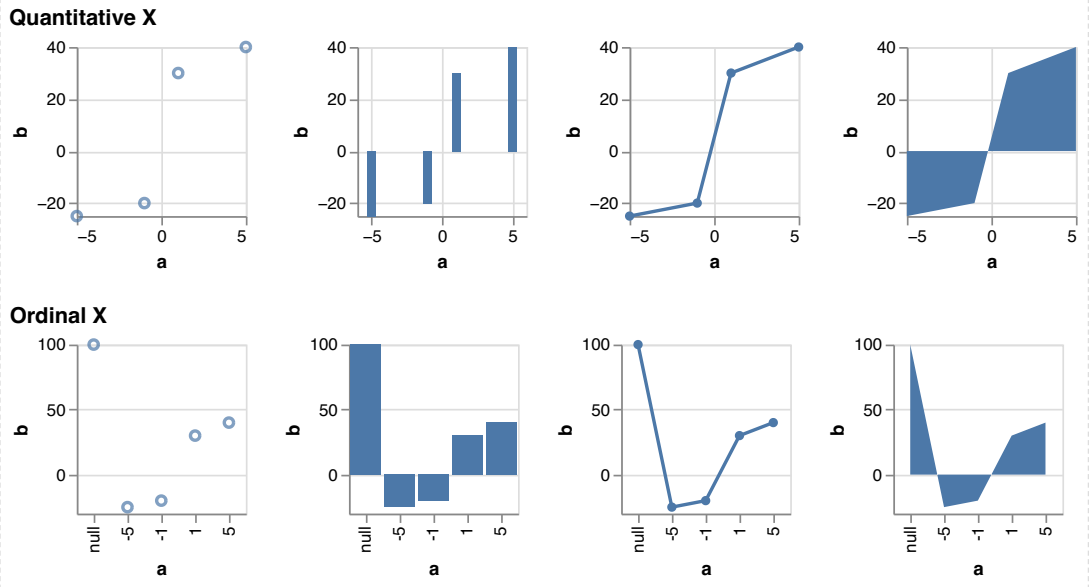
```
{
  "data": {
    "values": [
      {"a": null, "b": 100},
      {"a": -10, "b": null},
      {"a": -5, "b": -25},
      {"a": -1, "b": -20},
      {"a": 0, "b": null},
      {"a": 1, "b": 30},
      {"a": 5, "b": 40},
      {"a": 10, "b": null}
    ]
  },
  "config": {
    "mark": {"invalid": "filter", "tooltip": true}
  },
  "vconcat": [{
    "title": "Quantitative X",
    "hconcat": [{
      "width": 100,
      "height": 100,
      "mark": "point",
      "encoding": {
        "x": {"field": "a", "type": "quantitative"},
        "y": {"field": "b", "type": "quantitative"}
      }
    }, {
      "width": 100,
      "height": 100,
      "mark": "bar",
      "encoding": {
        "x": {"field": "a", "type": "quantitative"},
        "y": {"field": "b", "type": "quantitative"}
      }
    }, {
      "width": 100,
      "height": 100,
```

```
      "mark": {"type": "line", "point": true},
      "encoding": {
        "x": {"field": "a", "type": "quantitative"},
        "y": {"field": "b", "type": "quantitative"}
      }
    }, {
      "width": 100,
      "height": 100,
      "mark": "area",
      "encoding": {
        "x": {"field": "a", "type": "quantitative"},
        "y": {"field": "b", "type": "quantitative"}
      }
    }]
  },{
    "title": "Ordinal X",
    "hconcat": [{
      "width": 100,
      "height": 100,
      "mark": "point",
      "encoding": {
        "x": {"field": "a", "type": "ordinal"},
        "y": {"field": "b", "type": "quantitative"}
      }
    }, {
      "width": 100,
      "height": 100,
      "mark": "bar",
      "encoding": {
        "x": {"field": "a", "type": "ordinal"},
        "y": {"field": "b", "type": "quantitative"}
      }
    }, {
      "width": 100,
      "height": 100,
      "mark": {"type": "line", "point": true},
      "encoding": {
        "x": {"field": "a", "type": "ordinal"},
        "y": {"field": "b", "type": "quantitative"}
      }
    }, {
      "width": 100,
      "height": 100,
      "mark": "area",
      "encoding": {
        "x": {"field": "a", "type": "ordinal"},
        "y": {"field": "b", "type": "quantitative"}
      }
    }]
  }]
}
```

`"filter"`

The `"filter"` invalid mode *excludes* all invalid values from the visualization's *marks* and *scales*.

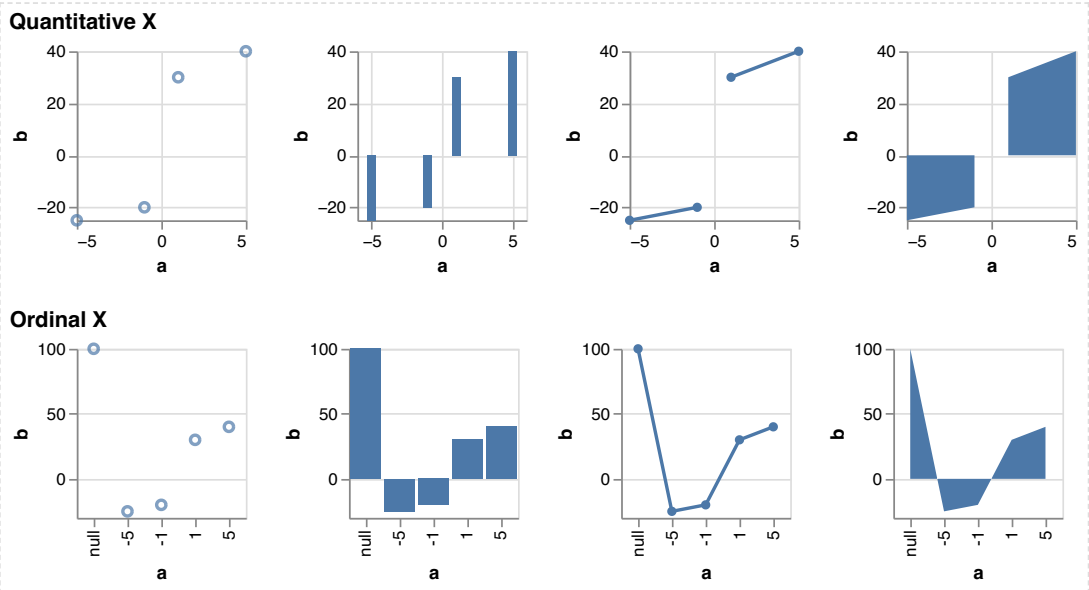
For path marks (for line, area, trail), this option will create paths that connect valid points, as if the points with invalid values do not exist.



Open in Vega Editor

`"break-paths"`

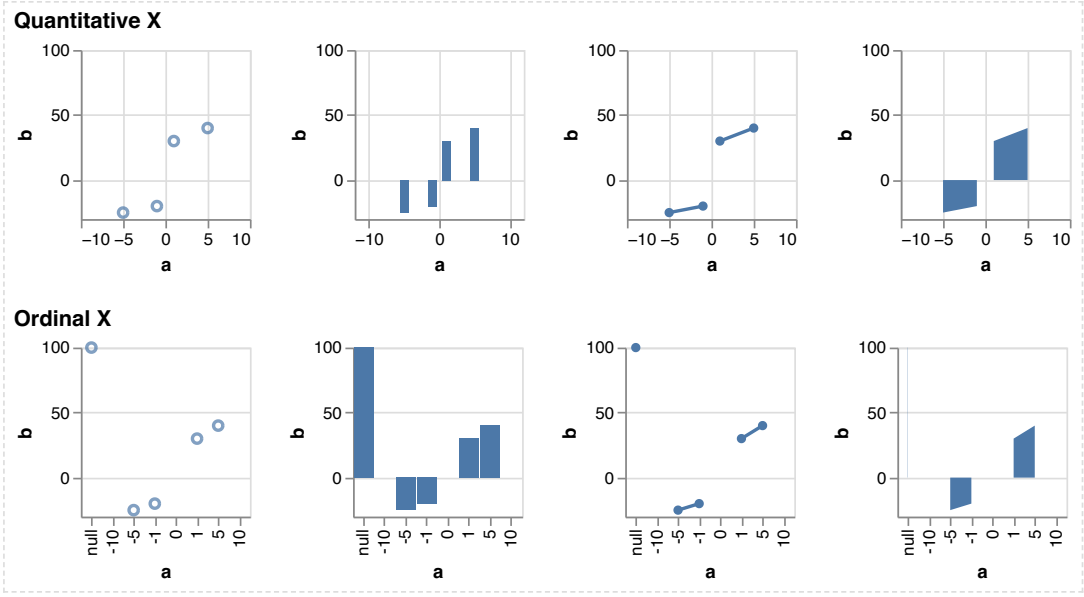
Break path marks (for line, area, trail) at invalid values. For non-path marks, this is equivalent to `"filter"`. All *scale* domains will *exclude* these filtered data points.



Open in Vega Editor

`"break-paths-show-domains"`

This option is like `"break-paths"`, except that all *scale* domains will instead *include* these filtered data points.

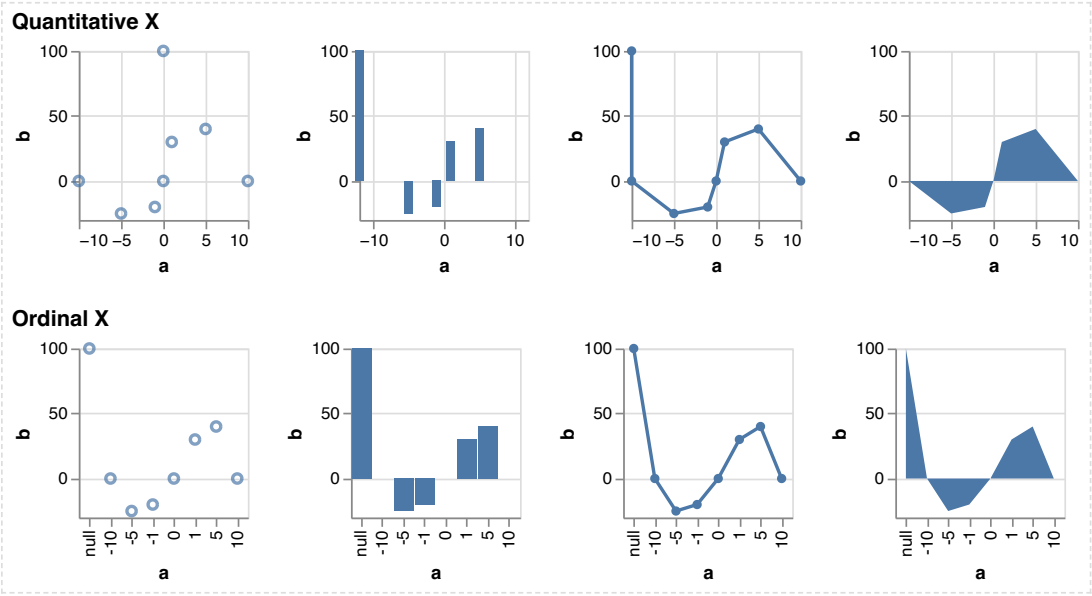


Open in Vega Editor

`"show"`

Include all data points in the marks and scale domains. Each scale will use the output for invalid values defined in `config.scale.invalid`

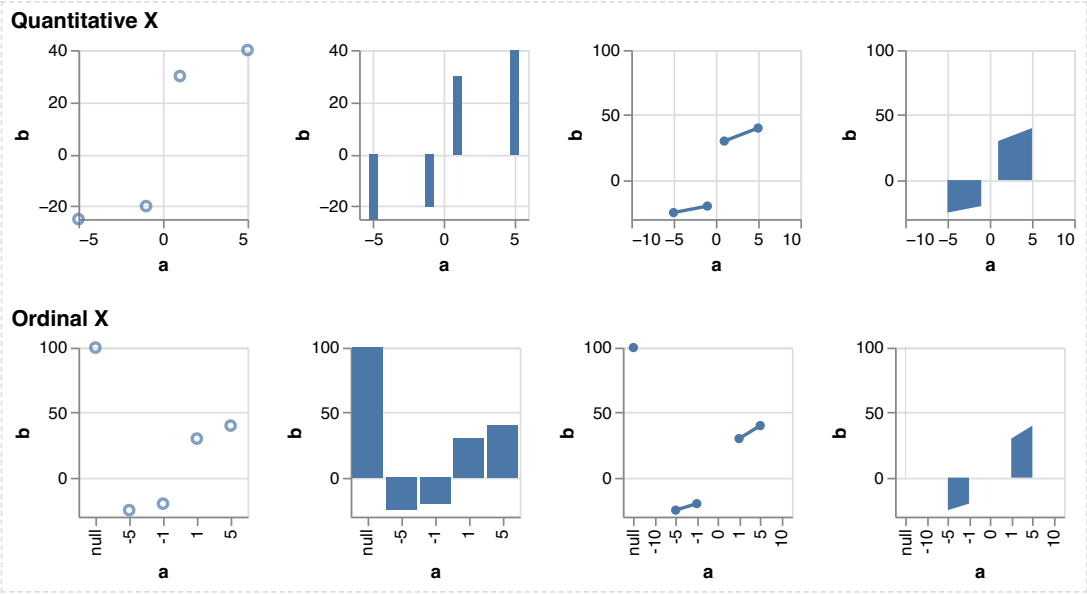
- or, if unspecified, by default invalid values will produce the same visual values as zero (if the scale includes zero) or the minimum value (if the scale does not include zero).



Open in Vega Editor

`"break-paths-and-show-path-domains"` (Default)

For historical reasons, Vega-Lite 5 currently uses `"break-paths-and-show-path-domains"` as the default invalid data mode (to avoid breaking changes). This is equivalent to `"break-path-keep-domains"` for path-based marks (line/area/trail) and `"filter"` for other marks.



Open in Vega Editor

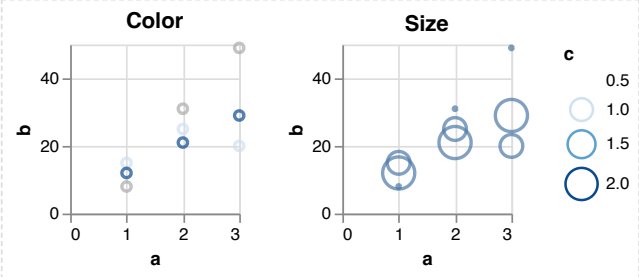
Scale Output for Invalid Values

You can use `config.scale.invalid` to defines scale outputs per channel for invalid values.

Property	Type	Description
invalid	ScaleInvalidDataConfig	An object that defines scale outputs per channel for invalid values (nulls and NaNs on a continuous scale). • The keys in this object are the scale channels. • The values is either `"zero-or-min"` (use zero if the scale includes zero or min value otherwise) or a value definition `{value: ...}`. *Example:* Setting this `config.scale.invalid` property to `{color: {value: '#aaa'}}` will make the visualization color all invalid values with '#aaa'. See https://vega.github.io/vega-lite/docs/invalid-data.html for more details.

Example: Output Color and Size

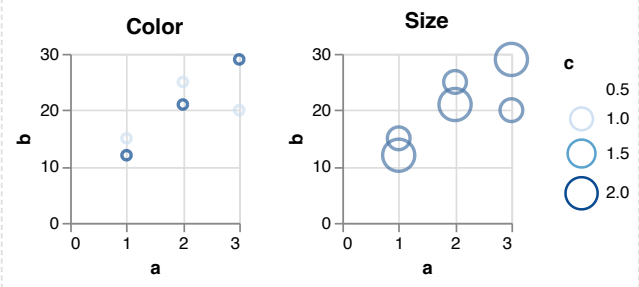
A visualization with `"filter"` invalid data mode will not filter (not exclude) color and size encoding if `config.scale.invalid.color` and `config.scale.invalid.size` are specified.



Open in Vega Editor

```
{
  "data": {
    "values": [
      {"a": 1, "b": 15, "c": 1},
      {"a": 2, "b": 25, "c": 1},
      {"a": 3, "b": 20, "c": 1},
      {"a": 1, "b": 12, "c": 2},
      {"a": 2, "b": 21, "c": 2},
      {"a": 3, "b": 29, "c": 2},
      {"a": 1, "b": 8, "c": null},
      {"a": 2, "b": 31, "c": null},
      {"a": 3, "b": 49, "c": null}
    ]
  },
  "config": {
    "mark": {"invalid": "filter", "tooltip": true},
    "scale": {"invalid": {"color": {"value": "#aaa"}, "size": {"value": 4}}}
  },
  "concat": [{
    "title": "Color",
    "width": 100,
    "height": 100,
    "mark": "point",
    "encoding": {
      "x": {"field": "a", "type": "quantitative"},
      "y": {"field": "b", "type": "quantitative"},
      "color": {"field": "c", "type": "quantitative"}
    }
  }, {
    "title": "Size",
    "width": 100,
    "height": 100,
    "mark": "point",
    "encoding": {
      "x": {"field": "a", "type": "quantitative"},
      "y": {"field": "b", "type": "quantitative"},
      "size": {"field": "c", "type": "quantitative"}
    }
  }]
}
```


Compare this with a similar spec, but without `config.scale.invalid`:



Open in Vega Editor


```
{
  "data": {
    "values": [
      {"a": 1, "b": 15, "c": 1},
      {"a": 2, "b": 25, "c": 1},
      {"a": 3, "b": 20, "c": 1},
      {"a": 1, "b": 12, "c": 2},
      {"a": 2, "b": 21, "c": 2},
      {"a": 3, "b": 29, "c": 2},
      {"a": 1, "b": 8, "c": null},
      {"a": 2, "b": 31, "c": null},
      {"a": 3, "b": 49, "c": null}
    ]
  },
  "config": {
    "mark": {"invalid": "filter", "tooltip": true}
  },
  "concat": [{
    "title": "Color",
    "width": 100,
    "height": 100,
    "mark": "point",
```

```
    "encoding": {
      "x": {"field": "a", "type": "quantitative"},
      "y": {"field": "b", "type": "quantitative"},
      "color": {"field": "c", "type": "quantitative"}
    }
  }, {
    "title": "Size",
    "width": 100,
    "height": 100,
    "mark": "point",
    "encoding": {
      "x": {"field": "a", "type": "quantitative"},
      "y": {"field": "b", "type": "quantitative"},
      "size": {"field": "c", "type": "quantitative"}
    }
  }]
}
```

Edit this page and submit a pull request!