The **sankey** package
Draw Sankey diagrams via TikZ

Paul Gaborit (`paul.gaborit@gmail.com`)

v3.0.2 – 2025/01/10

Abstract

The `sankey` package provides macros and environments to build *Sankey diagrams*[1], i.e. *flow diagrams* in which the width of the arrows is proportional to the flow rate. The initial idea for the first implementation came out from this question on TeX.StackExchange.

This manual contains three parts: User manual (p.2), Examples (p.22) and Installation & Implementation (p.43).

---

[1] `https://en.wikipedia.org/wiki/Sankey_diagram`

This manual contains three parts: User manual (p.2), Examples (p.22) and Installation & Implementation (p.43).

Note: the `sankey.dtx` and `sankey.ins` files are attachments of the current PDF document.

Part I

User manual

Contents

1 Preamble

To use the `sankey` package, insert the following line in your preamble:

```
\usepackage{sankey}
```

Note: the `sankey` package requires automatically the `xparse`, `etoolbox`, `xfp` and `tikz` packages, as well as the `calc`, `decorations.markings` and `dubins` (cf. 5.2 on page 20) TikZ libraries.

2 The `sankeydiagram` environment

`sankeydiagram` (*env.*) A `sankeydiagram` environment nested in a `tikzpicture` environment activates the sankey macros:

```
\begin{tikzpicture}
  \begin{sankeydiagram}[... options ...]
    ... sankey macros ...
  \end{sankeydiagram}
\end{tikzpicture}
```

3 Sankey diagram options

The `sankey` package uses `pgfkeys` to set options via *key=value* pairs with default path `/sankey` (and `/sankey/node parameters` for Sankey node parameters).

Options can be defined in three ways:

- via the optional argument of the `sankeydiagram` environment:

```
\begin{sankeydiagram}[debug=true]
  \sankeynode{name=a,quantity=1,angle=0,at={0,0}}
\end{sankeydiagram}
```

`\sankeyset`

- via the `\sankeyset` macro:

```
\begin{sankeydiagram}
  \sankeyset{debug=true}
  \sankeynode{name=a,quantity=1,angle=0,at={0,0}}
\end{sankeydiagram}
```

- temporarily modified for a single macro:

```
\begin{sankeydiagram}
  \sankeynode[debug=true]{name=a,quantity=1,angle=0,at={0,0}}
\end{sankeydiagram}
```

At the begining of each Sankey diagram, all options are initialized with initial values then the `every diagram` style is applied.

`every diagram` (*Key*) `/sankey/every diagram` (initially: `empty`)

This style is installed at beginning of each Sankey diagram.

For instance, to use a ratio of 5mm/10 by default (instead of 1cm/10) for all Sankey diagrams, add the following line:

```
\sankeyset{every diagram/.style={ratio=5mm/10}}
```

3.1 Keys to choose the scale

The scale or ratio of the Sankey diagram is the ratio between the `ratio length` and the `ratio quantity`.

`ratio quantity` (*Key*) `/sankey/ratio quantity=⟨number⟩` (initially: `10`)

Quantity (in units of flow) to define ratio. The ⟨`number`⟩ can be any math expression.

`ratio length` (*Key*) `/sankey/ratio length=⟨distance⟩` (initially: `1cm`)

Distance (a graphical distance) to define scale.

`ratio` (*Key*) `/sankey/ratio=⟨distance⟩/⟨number⟩` (initially: `1cm/10`)

Fix the ratio to ⟨`distance`⟩/⟨`number`⟩.

The initial ratio is 1 cm/10 units.

Note: the `sankey` package uses the `xfp` package to evaluate all math expressions that use quantities (in units of flow). You can therefore use quantities of a very large or very small order of magnitude. In contrast, for graphic distances, the sankey package uses the `pgfmath` package (all calculations must not exceed ±16383.99999).

3.2 Keys to define rotate offset

`rotate` (*Key*) `/sankey/rotate=⟨angle⟩` (initially: `0`)

The `rotate` key stores an offset angle applied to all Sankey nodes. This is useful when using the `rotate` option within a `tikzpicture` or a `scope`. This TikZ option is only applied to coordinates and not to TikZ nodes (remember that Sankey nodes are TikZ nodes). It's up to the author to keep the `rotate` option of the `tikzpicture` and that of the `sankeydiagram` synchronous.

3.3 Keys to define drawing parameters

`minimum radius` (*Key*) `/sankey/minimum radius=⟨distance⟩` (initially: `5mm`)

The minimum radius used by `\sankeyturn` and `\sankeydubins`.

`outin steps` (*Key*) `/sankey/outin steps=⟨integer⟩` (initially: `10`)

Number of steps used by the `\sankeyoutin` macro to simulate flow lanes with constant width.

3.4 Keys to choose drawing styles

`fill/.style` (*Key*) `/sankey/fill/.style=⟨style⟩` (initially: `line width=0pt,fill=white`)

This TikZ style is used to *fill* all sankey paths.

`draw/.style` (*Key*) `/sankey/draw/.style=⟨style⟩` (initially: `draw=black,line width=.4pt`)

This TikZ style is used to *draw* all sankey paths.

`start style` (*Key*) `/sankey/start style=⟨style name⟩` (initially: `none`)

There are three predefined *start* styles: `none`, `simple`, `arrow`.

`end style` (*Key*) `/sankey/end style=⟨style name⟩` (initially: `none`)

There are three predefined *end* styles: `none`, `simple`, `arrow`.

3.5 Keys to define new *start* and *end* styles

`new start style` (*Key*) `/sankey/new start style={⟨name⟩}{⟨fill path⟩}{⟨draw path⟩}`

Define the new start style named ⟨`name`⟩ with its ⟨`fill path`⟩ and its ⟨`draw path`⟩.

`new end style` (*Key*) `/sankey/new end style={⟨name⟩}{⟨fill path⟩}{⟨draw path⟩}`

Define the new end style named ⟨`name`⟩ with its ⟨`fill path`⟩ and its ⟨`draw path`⟩.

The ⟨`fill path`⟩ and the ⟨`draw path`⟩ are build in a TikZ scope where the origin is the center of the current Sankey node (its name is accessible via `\name`) and the coordinate system is rotated by its orientation.

3.6 The *debug* key

`debug` (*Key*) `/sankey/debug=⟨boolean⟩` (default: `true`)(initially: `false`)

To debug a sankey diagram.

4 Sankey nodes and flows

4.1 Create Sankey nodes

`\sankeynode` `\sankeynode[⟨options⟩]{⟨node parameters⟩}`

The `\sankeynode` macro defines a Sankey node. The ⟨`options`⟩ can be any Sankey diagram keys. To define a Sankey node, you must provide a *name*, a *quantity* and an *angle* as ⟨`node parameters`⟩.

`name` (*Key*) `/sankey/node parameters/name=⟨name⟩`

The ⟨`name`⟩ of the new Sankey node (and the associated TikZ node).

`quantity` (*Key*) `/sankey/node parameters/quantity=⟨quantity⟩`

The quantity (in flow unit) of the new Sankey node. The ⟨`quantity`⟩ can be any math expression.

`angle` (*Key*) `/sankey/node parameters/angle=⟨angle⟩`

The orientation of the flow (0 points to the right) of the new Sankey node.

`at` (*Key*) `/sankey/node parameters/at=⟨at⟩` (initially: `0,0`)

The position of the new Sankey node (a TikZ coordinate **without** round brackets or parentheses).

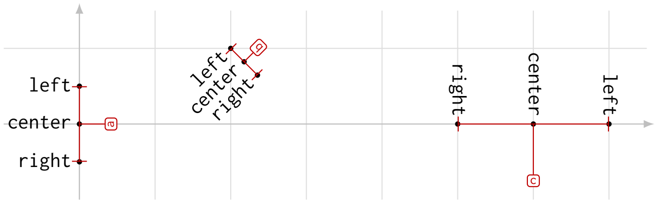
`anchor` (*Key*) `/sankey/node parameters/anchor=⟨anchor⟩` (initially: `center`)

Specify the anchor of the Sankey node. Possible values are `center`, `left` or `right`.

`as` (*Key*) `/sankey/node parameters/as=⟨name⟩`

Copy the *name*, the *quantity*, the *angle* and the *position* of the Sankey node named ⟨`name`⟩.

A Sankey node is also a Tikz node but with only three anchors: `left`, `center` and `right`[2]:



```
\begin{tikzpicture}
  \begin{sankeydiagram}[debug]
    \sankeynode{name=a,quantity=10}
    \sankeynode{name=b,quantity=5,angle=45,at={2,1},anchor=left}
    \sankeynode{name=c,quantity=20,angle=-90,at={5,0},anchor=right}
    \foreach \nodename/\pos in {a/left,b/below left,c/above}{
      \foreach \ancname in {left,center,right}{
        \node[node font=\ttfamily\footnotesize,\pos=1mm of \nodename.\ancname,
        inner sep=0pt,rotate=\sankeygetnodeorient{\nodename},anchor=east]
        {\ancname\vphantom{g}};
        \fill[black] (\nodename.\ancname) circle(1pt);
      }
    }
  \end{sankeydiagram}
\end{tikzpicture}
```

4.1.1 Choose default parameters

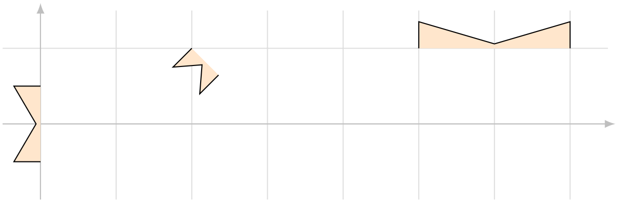
`every node/.style` (*Key*) `/sankey/every node/.style={⟨node parameters⟩}` (initially: `empty`)

The ⟨`node parameters`⟩ defined by the `every node` style is installed at the creation of every Sankey node.

4.1.2 Create starting and ending nodes via macros

`\sankeynodestart` `\sankeynodestart[⟨options⟩]{⟨node parameters⟩}`

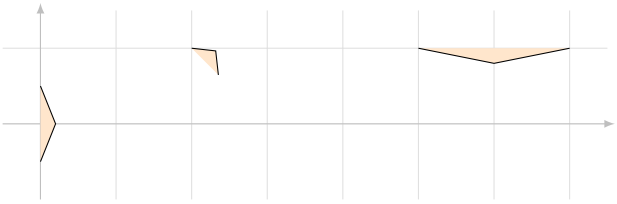
The `\sankeynodestart` creates and fills/draws a starting Sankey node:



```
\begin{tikzpicture}
  \begin{sankeydiagram}[start style=arrow,fill/.style={fill=orange!20}]
    \sankeynodestart{name=a,quantity=10}
    \sankeynodestart{name=b,quantity=5,angle=45,at={2,1},anchor=left}
    \sankeynodestart{name=c,quantity=20,angle=-90,at={5,1},anchor=right}
  \end{sankeydiagram}
\end{tikzpicture}
```

`\sankeynodeend` `\sankeynodeend[⟨options⟩]{⟨node parameters⟩}`

The `\sankeynodeend` creates and fills/draws an ending Sankey node:



---

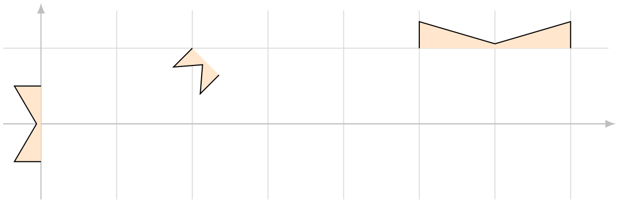
[2] In fact, to be able to use the TikZ `fit` library, the `north`, `north east` and `north west` anchors exist and are equal to `left`, the `east` and `west` anchors exist and are equal to `center` and the `south`, `south east` and `south west` anchors exist and are equal to `right`.

```
\begin{tikzpicture}
  \begin{sankeydiagram}[end style=simple,fill/.style={fill=orange!20}]
    \sankeynodeend{name=a,quantity=10}
    \sankeynodeend{name=b,quantity=5,angle=45,at={2,1},anchor=left}
    \sankeynodeend{name=c,quantity=20,angle=-90,at={5,1},anchor=right}
  \end{sankeydiagram}
\end{tikzpicture}
```

4.1.3 Create starting and ending nodes via options

`start` (*Key*) `/sankey/node parameters/start=⟨boolean⟩` (default: `true`)(initially: `false`)

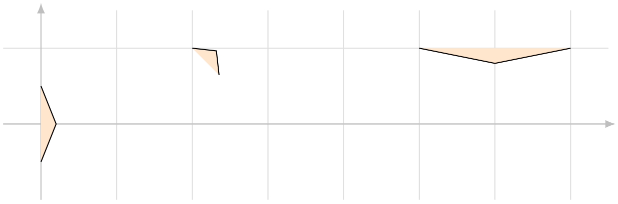
The `\sankeynode` macro acts as the `\sankeynodestart` macro if you add the `start` option to its options.



```
\begin{tikzpicture}
  \begin{sankeydiagram}[start style=arrow,fill/.style={fill=orange!20}]
    \sankeynode{name=a,quantity=10,start}
    \sankeynode{name=b,quantity=5,angle=45,at={2,1},anchor=left,start}
    \sankeynode{name=c,quantity=20,angle=-90,at={5,1},anchor=right,start}
  \end{sankeydiagram}
\end{tikzpicture}
```

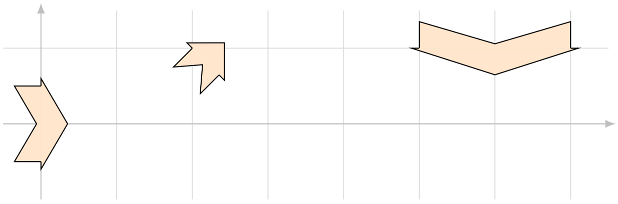
`end` (*Key*) `/sankey/node parameters/end=⟨boolean⟩` (default: `true`)(initially: `false`)

The `\sankeynode` macro acts as the `\sankeynodeend` macro if you add the `end` option to its options.



```
\begin{tikzpicture}
  \begin{sankeydiagram}[end style=simple,fill/.style={fill=orange!20}]
    \sankeynode{name=a,quantity=10,end}
    \sankeynode{name=b,quantity=5,angle=45,at={2,1},anchor=left,end}
    \sankeynode{name=c,quantity=20,angle=-90,at={5,1},anchor=right,end}
  \end{sankeydiagram}
\end{tikzpicture}
```

Although rarely necessary, you can mix these two parameters:



```
\begin{tikzpicture}
  \begin{sankeydiagram}
    \sankeyset{
      end style=arrow,
      start style=arrow,
      fill/.style={fill=orange!20}
    }
    \sankeynode{name=a,quantity=10,start,end}
    \sankeynode{name=b,quantity=5,angle=45,at={2,1},anchor=left,start,end}
    \sankeynode{name=c,quantity=20,angle=-90,at={5,1},anchor=right,start,end}
  \end{sankeydiagram}
\end{tikzpicture}
```

4.2 Retrieve information from Sankey nodes

`\sankeygetnodeqty`

`\sankeygetnodeqty{⟨node name⟩}`

The expandable command `\sankeygetnodeqty` returns the quantity assigned to the Sankey node named ⟨`node name`⟩.

`\sankeyqtytolen`

`\sankeyqtytolen{⟨quantity⟩}`

The expandable `\sankeyqtytolen` macro converts ⟨`quantity`⟩ to graphical length using the current ratio.

`\sankeygetnodeorient`

`\sankeygetnodeorient{⟨node name⟩}`

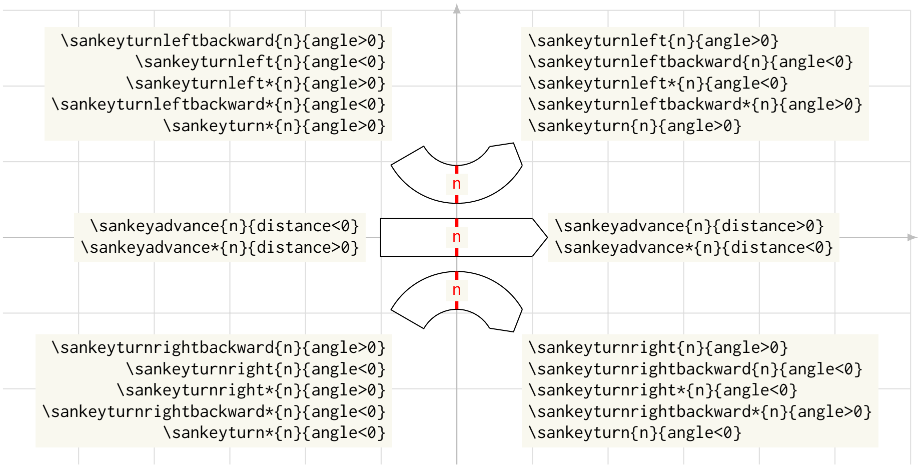
The expandable command `\sankeygetnodeorient` returns the angle (orientation) assigned to the Sankey node named ⟨`node name`⟩.

4.3 Move nodes

All the macros of this section move a Sankey node and fill/draw a portion of the Sankey flow. Then the previous position of the Sankey node is accessible via the `-old` suffix (i.e. if you move the `a` node, its previous position is the `a-old` node).

The starred version of each of these macros moves in the opposite direction to their non-starred version.

Except for the `\sankeyturn` macro, a negative value (distance or angle) moves in the opposite direction (the `\sankeyturn` macro is an exception: a negative angle turns right while a positive value turns left).



```
\sankeyturnleftbackward{n}{angle>0}
        \sankeyturnleft{n}{angle<0}
       \sankeyturnleft*{n}{angle>0}
\sankeyturnleftbackward*{n}{angle<0}
            \sankeyturn*{n}{angle>0}
```

```
\sankeyturnleft{n}{angle>0}
\sankeyturnleftbackward{n}{angle<0}
\sankeyturnleft*{n}{angle<0}
\sankeyturnleftbackward*{n}{angle>0}
\sankeyturn{n}{angle>0}
```

```
 \sankeyadvance{n}{distance<0}
\sankeyadvance*{n}{distance>0}
```

```
\sankeyadvance{n}{distance>0}
\sankeyadvance*{n}{distance<0}
```

```
\sankeyturnrightbackward{n}{angle>0}
        \sankeyturnright{n}{angle<0}
       \sankeyturnright*{n}{angle>0}
\sankeyturnrightbackward*{n}{angle<0}
             \sankeyturn*{n}{angle<0}
```

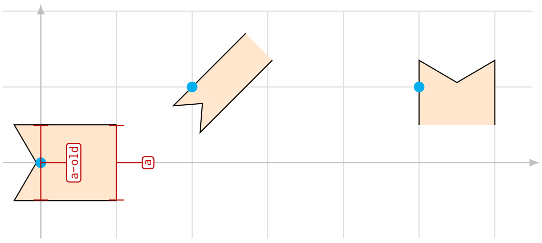
```
\sankeyturnright{n}{angle>0}
\sankeyturnrightbackward{n}{angle<0}
\sankeyturnright*{n}{angle<0}
\sankeyturnrightbackward*{n}{angle>0}
\sankeyturn{n}{angle<0}
```

4.3.1 Macro to move straight (forward or backward)

`\sankeyadvance`

`\sankeyadvance[⟨options⟩]{⟨node name⟩}{⟨distance⟩}`

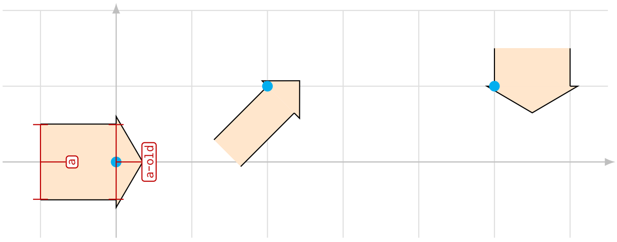
The `\sankeyadvance` moves the sankey node straight ahead and fills/draws this portion of the sankey path. A positive ⟨`distance`⟩ moves forward while a negative ⟨`distance`⟩ moves backward.



```
\begin{tikzpicture}
  \begin{sankeydiagram}[start style=arrow,fill/.style={fill=orange!20}]
    \sankeynodestart{name=a,quantity=10}
    \sankeyadvance{a}{1cm}
    \sankeynode[debug]{as=a}
    \sankeynode[debug]{as=a-old}
    \sankeynodestart{name=b,quantity=5,angle=45,at={2,1},anchor=left}
    \sankeyadvance{b}{1cm}
    \sankeynodestart{name=c,quantity=10,angle=-90,at={5,1},anchor=right}
    \sankeyadvance{c}{5mm}
    \fill[cyan] (0,0) circle(2pt) (2,1) circle(2pt) (5,1) circle(2pt);
  \end{sankeydiagram}
\end{tikzpicture}
```

`\sankeyadvance*[⟨options⟩]{⟨node name⟩}{⟨distance⟩}`

The `\sankeyadvance*` moves the sankey node straight back and fills/draws this portion of the sankey path. A positive ⟨`distance`⟩ moves backward while a negative ⟨`distance`⟩ moves forward.

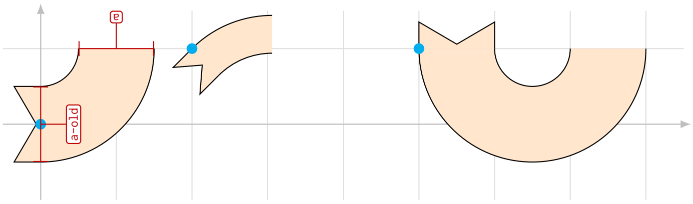


```
\begin{tikzpicture}
  \begin{sankeydiagram}[end style=arrow,fill/.style={fill=orange!20}]
    \sankeynodeend{name=a,quantity=10}
    \sankeyadvance*{a}{1cm}
    \sankeynode[debug]{as=a}
    \sankeynode[debug]{as=a-old}
    \sankeynodeend{name=b,quantity=5,angle=45,at={2,1},anchor=left}
    \sankeyadvance*{b}{1cm}
    \sankeynodeend{name=c,quantity=10,angle=-90,at={5,1},anchor=right}
    \sankeyadvance*{c}{5mm}
    \fill[cyan] (0,0) circle(2pt) (2,1) circle(2pt) (5,1) circle(2pt);
  \end{sankeydiagram}
\end{tikzpicture}
```

4.3.2 Macro to turn forward or backward

\sankeyturn `\sankeyturn[⟨options⟩]{⟨node name⟩}{⟨angle⟩}`

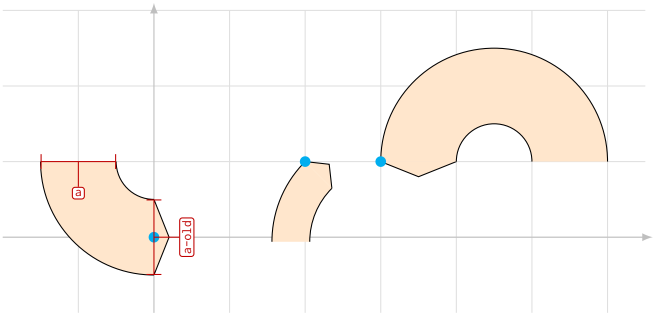
The `\sankeyturn` macro moves the sankey node by turning to one side or the other and fills/draws this portion of the sankey path. A *positive* ⟨angle⟩ turns left while a *negative* ⟨angle⟩ turns right.



```
\begin{tikzpicture}
  \begin{sankeydiagram}[start style=arrow,fill/.style={fill=orange!20}]
    \sankeynodestart{name=a,quantity=10}
    \sankeyturn{a}{90}
    \sankeynode[debug]{as=a}
    \sankeynode[debug]{as=a-old}
    \sankeynodestart{name=b,quantity=5,angle=45,at={2,1},anchor=left}
    \sankeyturn[minimum radius=1cm]{b}{-45}
    \sankeynodestart{name=c,quantity=10,angle=-90,at={5,1},anchor=right}
    \sankeyturn{c}{180}
    \fill[cyan] (0,0) circle(2pt) (2,1) circle(2pt) (5,1) circle(2pt);
  \end{sankeydiagram}
\end{tikzpicture}
```

`\sankeyturn*[⟨options⟩]{⟨node name⟩}{⟨angle⟩}`

The `\sankeyturn*` macro moves the sankey node backward by turning right or left and fills/draws this portion of the sankey path. A *positive* ⟨angle⟩ turns left while a *negative* ⟨angle⟩ turns right.



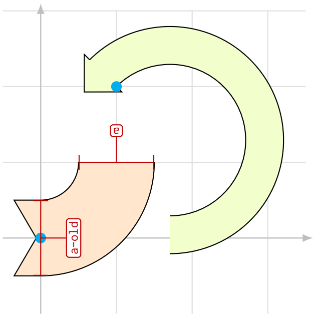
```
\begin{tikzpicture}
  \begin{sankeydiagram}[end style=simple,fill/.style={fill=orange!20}]
    \sankeynodeend{name=a,quantity=10}
    \sankeyturn*{a}{90}
    \sankeynode[debug]{as=a}
    \sankeynode[debug]{as=a-old}
    \sankeynodeend{name=b,quantity=5,angle=45,at={2,1},anchor=left}
    \sankeyturn*[minimum radius=1cm]{b}{-45}
    \sankeynodeend{name=c,quantity=10,angle=-90,at={3,1},anchor=right}
    \sankeyturn*{c}{180}
    \fill[cyan] (0,0) circle(2pt) (2,1) circle(2pt) (3,1) circle(2pt);
  \end{sankeydiagram}
\end{tikzpicture}
```

4.3.3 Macros to turn left (forward or backward)

`\sankeyturnleft`

`\sankeyturnleft[⟨options⟩]{⟨node name⟩}{⟨angle⟩}`

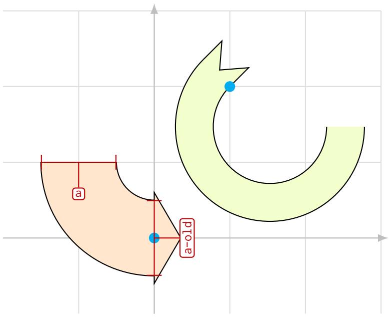
The `\sankeyturnleft` macro moves the sankey node by turning left and fills/draws this portion of the sankey path. A *positive* ⟨angle⟩ turns forward while a *negative* ⟨angle⟩ turns backward.



```
\begin{tikzpicture}
  \begin{sankeydiagram}[start style=arrow,end style=arrow]
    \sankeyset{fill/.style={fill=orange!20}}
    \sankeynodestart{name=a,quantity=10}
    \sankeyturnleft{a}{90}
    \sankeynode[debug]{as=a}
    \sankeynode[debug]{as=a-old}
    \sankeyset{fill/.style={fill=lime!20}}
    \sankeynodeend{name=b,quantity=5,angle=-135,at={1,2},anchor=left}
    \sankeyturnleft[minimum radius=1cm]{b}{-225}
    \fill[cyan] (0,0) circle(2pt) (1,2) circle(2pt);
\end{sankeydiagram}
\end{tikzpicture}
```

`\sankeyturnleft*[⟨options⟩]{⟨node name⟩}{⟨angle⟩}`

The `\sankeyturnleft*` macro moves the sankey node backward by turning left and fills/draws this portion of the sankey path. A *positive* ⟨angle⟩ turns backward while a *negative* ⟨angle⟩ turns forward.

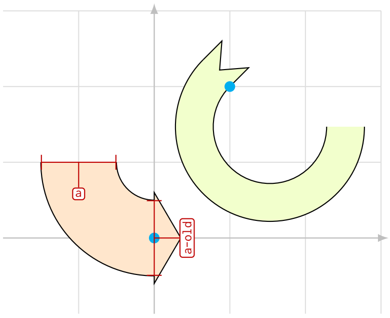


```
\begin{tikzpicture}
  \begin{sankeydiagram}[start style=arrow,end style=arrow]
    \sankeyset{fill/.style={fill=orange!20}}
    \sankeynodeend{name=a,quantity=10}
    \sankeyturnleft*{a}{90}
    \sankeynode[debug]{as=a}
    \sankeynode[debug]{as=a-old}
    \sankeyset{fill/.style={fill=lime!20}}
    \sankeynodestart{name=b,quantity=5,angle=-135,at={1,2},anchor=left}
    \sankeyturnleft*[minimum radius=.75cm]{b}{-225}
    \fill[cyan] (0,0) circle(2pt) (1,2) circle(2pt);
\end{sankeydiagram}
\end{tikzpicture}
```

`\sankeyturnleftbackward`

`\sankeyturnleftbackward[⟨options⟩]{⟨node name⟩}{⟨angle⟩}`

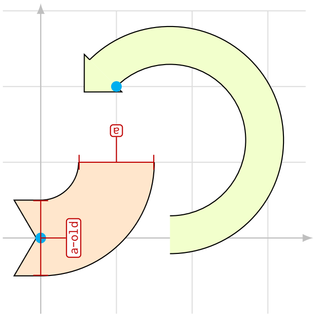
The `\sankeyturnleftbackward` macro moves the sankey node backward by turning left and fills/draws this portion of the sankey path. A *positive* ⟨`angle`⟩ turns backward while a *negative* ⟨`angle`⟩ turns forward.



```
\begin{tikzpicture}
  \begin{sankeydiagram}[start style=arrow,end style=arrow]
    \sankeyset{fill/.style={fill=orange!20}}
    \sankeynodeend{name=a,quantity=10}
    \sankeyturnleftbackward{a}{90}
    \sankeynode[debug]{as=a}
    \sankeynode[debug]{as=a-old}
    \sankeyset{fill/.style={fill=lime!20}}
    \sankeynodestart{name=b,quantity=5,angle=-135,at={1,2},anchor=left}
    \sankeyturnleftbackward[minimum radius=.75cm]{b}{-225}
    \fill[cyan] (0,0) circle(2pt) (1,2) circle(2pt);
\end{sankeydiagram}
\end{tikzpicture}
```

`\sankeyturnleftbackward*[⟨options⟩]{⟨node name⟩}{⟨angle⟩}`

The `\sankeyturnleftbackward*` macro moves the sankey node by turning left and fills/draws this portion of the sankey path. A *positive* ⟨`angle`⟩ turns forward while a *negative* ⟨`angle`⟩ turns backward.



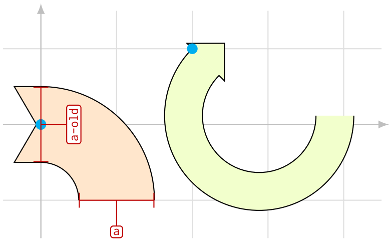
```
\begin{tikzpicture}
  \begin{sankeydiagram}[start style=arrow,end style=arrow]
    \sankeyset{fill/.style={fill=orange!20}}
    \sankeynodestart{name=a,quantity=10}
    \sankeyturnleftbackward*{a}{90}
    \sankeynode[debug]{as=a}
    \sankeynode[debug]{as=a-old}
    \sankeyset{fill/.style={fill=lime!20}}
    \sankeynodeend{name=b,quantity=5,angle=-135,at={1,2},anchor=left}
    \sankeyturnleftbackward*[minimum radius=1cm]{b}{-225}
    \fill[cyan] (0,0) circle(2pt) (1,2) circle(2pt);
\end{sankeydiagram}
\end{tikzpicture}
```

4.3.4 Macros to turn right (forward or backward)

`\sankeyturnright`

`\sankeyturnright[⟨options⟩]{⟨node name⟩}{⟨angle⟩}`

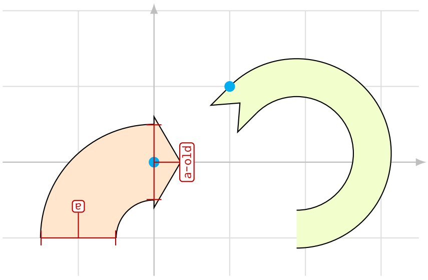
The `\sankeyturnright` macro moves the sankey node by turning right and fills/draws this portion of the sankey path. A *positive* ⟨`angle`⟩ turns forward while a *negative* ⟨`angle`⟩ turns backward.



```
\begin{tikzpicture}
  \begin{sankeydiagram}[start style=arrow,end style=arrow]
    \sankeyset{fill/.style={fill=orange!20}}
    \sankeynodestart{name=a,quantity=10}
    \sankeyturnright[fill/.style={fill=orange!20}]{a}{90}
    \sankeynode[debug]{as=a}
    \sankeynode[debug]{as=a-old}
    \sankeyset{fill/.style={fill=lime!20}}
    \sankeynodeend{name=b,quantity=5,angle=45,at={2,1},anchor=left}
    \sankeyturnright[minimum radius=.75cm]{b}{-225}
    \fill[cyan] (0,0) circle(2pt) (2,1) circle(2pt);
\end{sankeydiagram}
\end{tikzpicture}
```

`\sankeyturnright*[⟨options⟩]{⟨node name⟩}{⟨angle⟩}`

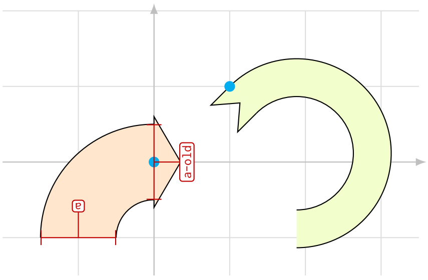
The `\sankeyturnright*` macro moves the sankey node backward by turning right and fills/draws this portion of the sankey path. A *positive* ⟨`angle`⟩ turns backward while a *negative* ⟨`angle`⟩ turns forward.



```
\begin{tikzpicture}
  \begin{sankeydiagram}[start style=arrow,end style=arrow]
    \sankeyset{fill/.style={fill=orange!20}}
    \sankeynodeend{name=a,quantity=10}
    \sankeyturnright*[fill/.style={fill=orange!20}]{a}{90}
    \sankeynode[debug]{as=a}
    \sankeynode[debug]{as=a-old}
    \sankeyset{fill/.style={fill=lime!20}}
    \sankeynodestart{name=b,quantity=5,angle=45,at={1,1},anchor=left}
    \sankeyturnright*[minimum radius=.75cm]{b}{-225}
    \fill[cyan] (0,0) circle(2pt) (1,1) circle(2pt);
\end{sankeydiagram}
\end{tikzpicture}
```

`\sankeyturnrightbackward` `\sankeyturnrightbackward[⟨options⟩]{⟨node name⟩}{⟨angle⟩}`

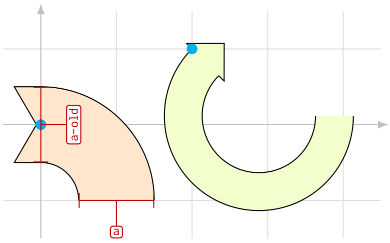
The `\sankeyturnrightbackward` macro moves the sankey node backward by turning right and fills/draws this portion of the sankey path. A *positive* ⟨angle⟩ turns backward while a *negative* ⟨angle⟩ turns forward.



```
\begin{tikzpicture}
  \begin{sankeydiagram}[start style=arrow,end style=arrow]
    \sankeyset{fill/.style={fill=orange!20}}
    \sankeynodeend{name=a,quantity=10}
    \sankeyturnrightbackward[fill/.style={fill=orange!20}]{a}{90}
    \sankeynode[debug]{as=a}
    \sankeynode[debug]{as=a-old}
    \sankeyset{fill/.style={fill=lime!20}}
    \sankeynodestart{name=b,quantity=5,angle=45,at={1,1},anchor=left}
    \sankeyturnrightbackward[minimum radius=.75cm]{b}{-225}
    \fill[cyan] (0,0) circle(2pt) (1,1) circle(2pt);
\end{sankeydiagram}
\end{tikzpicture}
```

`\sankeyturnrightbackward*[⟨options⟩]{⟨node name⟩}{⟨angle⟩}`

The `\sankeyturnrightbackward*` macro moves the sankey node forward by turning right and fills/draws this portion of the sankey path. A *positive* ⟨angle⟩ turns forward while a *negative* ⟨angle⟩ turns backward.



```
\begin{tikzpicture}
  \begin{sankeydiagram}[start style=arrow,end style=arrow]
    \sankeyset{fill/.style={fill=orange!20}}
    \sankeynodestart{name=a,quantity=10}
    \sankeyturnrightbackward*[fill/.style={fill=orange!20}]{a}{90}
    \sankeynode[debug]{as=a}
    \sankeynode[debug]{as=a-old}
    \sankeyset{fill/.style={fill=lime!20}}
    \sankeynodeend{name=b,quantity=5,angle=45,at={2,1},anchor=left}
    \sankeyturnrightbackward*[minimum radius=.75cm]{b}{-225}
    \fill[cyan] (0,0) circle(2pt) (2,1) circle(2pt);
\end{sankeydiagram}
\end{tikzpicture}
```

4.4 Links between nodes

The macros described in this section fill/draw a lane between two Sankey nodes.

Note: since Sankey nodes are oriented, linking `A` node to `B` node does not produce the same result as linking `B` node to `A` node!

`\sankeyoutin` `\sankeyoutin[⟨options⟩]{⟨node A⟩}{⟨node B⟩}`

The `\sankeyoutin` macro fills/draws a lane from ⟨`node A`⟩ to ⟨`node B`⟩ using a Bézier curve with regular steps (10 steps by default) to simulate constant width lane.

Note: The constant width and the minimum curvature are *not* guaranteed!

`\sankeydubins` `\sankeydubins[⟨options⟩]{⟨node A⟩}{⟨node B⟩}`

The `\sankeydubins` macro fills/draws a lane between ⟨`node A`⟩ and ⟨`node B`⟩ using a Dubins path[3].

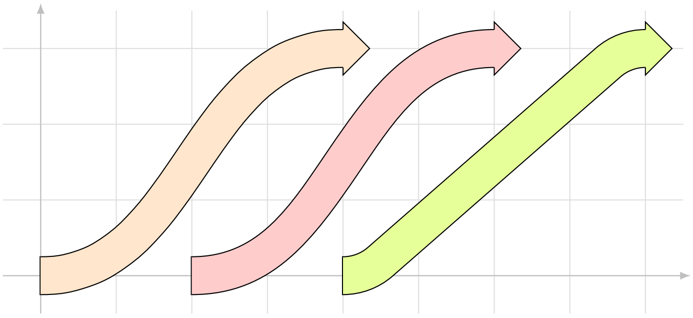
Note: The constant width and the minimum curvature are guaranteed.

---

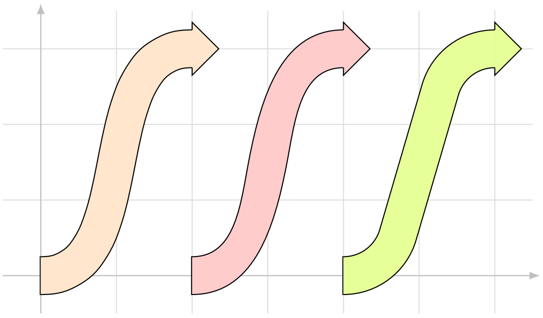
[3] `https://en.wikipedia.org/wiki/Dubins_path`

4.4.1 Comparison between `outin` and `dubins` paths

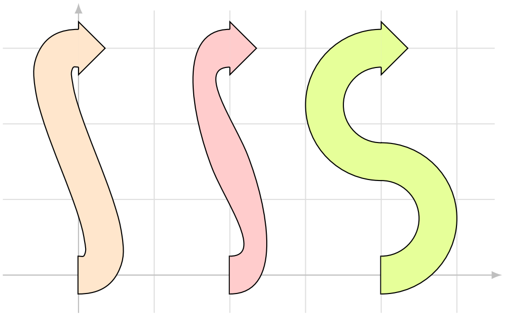
The following diagrams compare `outin` path with 10 steps (orange), `outin` path with 2 steps (red) and `dubins` path (lime) in various positions.



```
\begin{tikzpicture}
  \begin{sankeydiagram}[start style=simple,end style=arrow]
    \sankeyset{fill/.style={fill=orange!20}}
    \sankeynodestart{name=a,quantity=5}
    \sankeynodeend{name=b,at={4,3},quantity=5}
    \sankeyoutin{a}{b}

    \sankeyset{fill/.style={fill=red!20}}
    \sankeynodestart{name=a,at={2,0},quantity=5}
    \sankeynodeend{name=b,at={6,3},quantity=5}
    \sankeyoutin[outin steps=2]{a}{b}

    \sankeyset{fill/.style={fill=lime!40}}
    \sankeynodestart{name=a,at={4,0},quantity=5}
    \sankeynodeend{name=b,at={8,3},quantity=5}
    \sankeydubins[minimum radius=5mm]{a}{b}
  \end{sankeydiagram}
\end{tikzpicture}
```

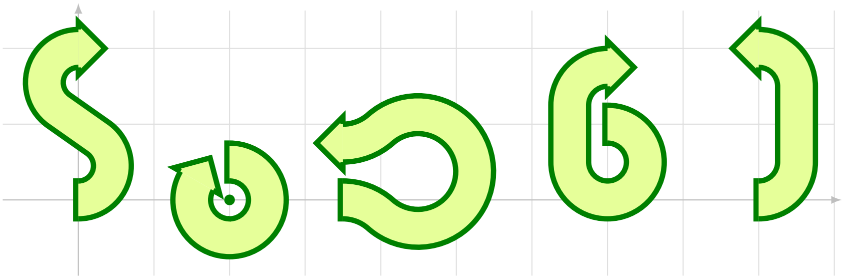


```
\begin{tikzpicture}
  \begin{sankeydiagram}[start style=simple,end style=arrow]
    \sankeyset{fill/.style={fill=orange!20}}
    \sankeynodestart{name=a,quantity=5}
    \sankeynodeend{name=b,at={2,3},quantity=5}
    \sankeyoutin{a}{b}

    \sankeyset{fill/.style={fill=red!20}}
    \sankeynodestart{name=a,at={2,0},quantity=5}
    \sankeynodeend{name=b,at={4,3},quantity=5}
    \sankeyoutin[outin steps=2]{a}{b}

    \sankeyset{fill/.style={fill=lime!40}}
    \sankeynodestart{name=a,at={4,0},quantity=5}
    \sankeynodeend{name=b,at={6,3},quantity=5}
    \sankeydubins[minimum radius=5mm]{a}{b}
  \end{sankeydiagram}
\end{tikzpicture}
```



```
\begin{tikzpicture}
  \begin{sankeydiagram}[start style=simple,end style=arrow]
    \sankeyset{fill/.style={fill=orange!20}}
    \sankeynodestart{name=a,quantity=5}
    \sankeynodeend{name=b,at={0,3},quantity=5}
    \sankeyoutin{a}{b}

    \sankeyset{fill/.style={fill=red!20}}
    \sankeynodestart{name=a,at={2,0},quantity=5}
    \sankeynodeend{name=b,at={2,3},quantity=5}
    \sankeyoutin[outin steps=2]{a}{b}

    \sankeyset{fill/.style={fill=lime!40}}
    \sankeynodestart{name=a,at={4,0},quantity=5}
    \sankeynodeend{name=b,at={4,3},quantity=5}
    \sankeydubins[minimum radius=5mm]{a}{b}
  \end{sankeydiagram}
\end{tikzpicture}
```

4.4.2 Examples of `dubins` paths

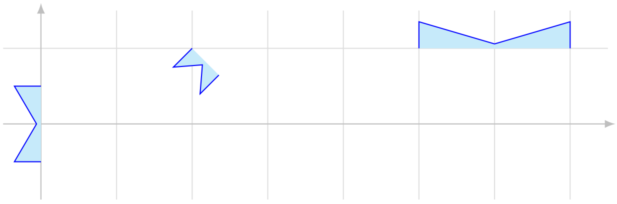


```
\begin{tikzpicture}
  \begin{sankeydiagram}[start style=simple,end style=arrow]
    \sankeyset{
      fill/.style={fill=lime!40},
      draw/.style={draw=green!50!black,line width=2pt},
    }

    \sankeynodestart{name=a,quantity=5}
    \sankeynodeend{name=b,at={0,2},quantity=5}
    \sankeydubins[minimum radius=2mm]{a}{b}

    \fill[green!50!black] (2,0) coordinate (c) circle(2pt);
    \sankeynodestart{name=a,at={[shift={(c)}]90:5mm},quantity=5}
    \sankeynodeend{name=b,at={[shift={(c)}]150:5mm},angle=60,quantity=5}
    \sankeydubins[minimum radius=2.5mm]{a}{b}

    \sankeynodestart{name=a,at={3.5,0},quantity=5}
    \sankeynodeend{name=b,at={3.5,.75},angle=-180,quantity=5}
    \sankeydubins[minimum radius=5mm]{a}{b}

    \sankeynodestart{name=a,at={7,1},quantity=5}
    \sankeynodeend{name=b,at={7,1.75},quantity=5}
    \sankeydubins[minimum radius=2.5mm]{a}{b}

    \sankeynodestart{name=a,at={9,0},quantity=5}
    \sankeynodeend{name=b,at={9,2},angle=180,quantity=5}
    \sankeydubins[minimum radius=2.5mm]{a}{b}
  \end{sankeydiagram}
\end{tikzpicture}
```

4.5 Pure filling/drawing macros

\sankeystart `\sankeystart[⟨options⟩]{⟨name⟩}`

The `\sankeystart` fills/draws a starting extremity attached to the preexisting Sankey node ⟨name⟩:

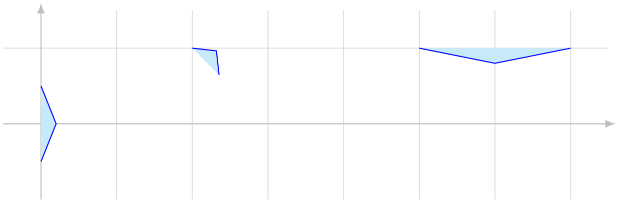


```
\begin{tikzpicture}
  \begin{sankeydiagram}
    [start style=arrow,fill/.style={fill=cyan!20},draw/.style={draw=blue}]
    \sankeynode{name=a,quantity=10}
    \sankeystart{a}
    \sankeynode{name=b,quantity=5,angle=45,at={2,1},anchor=left}
    \sankeystart{b}
    \sankeynode{name=c,quantity=20,angle=-90,at={5,1},anchor=right}
    \sankeystart{c}
  \end{sankeydiagram}
\end{tikzpicture}
```

\sankeyend `\sankeyend[⟨options⟩]{⟨name⟩}`

The `\sankeyend` fills/draws an ending extremity attached to the preexisting Sankey node ⟨name⟩:

```
\begin{tikzpicture}
  \begin{sankeydiagram}
    [end style=simple,fill/.style={fill=cyan!20},draw/.style={draw=blue}]
    \sankeynode{name=a,quantity=10}
    \sankeyend{a}
    \sankeynode{name=b,quantity=5,angle=45,at={2,1},anchor=left}
    \sankeyend{b}
    \sankeynode{name=c,quantity=20,angle=-90,at={5,1},anchor=right}
    \sankeyend{c}
  \end{sankeydiagram}
\end{tikzpicture}
```



4.6 Forked node

4.6.1 Create and fork a Sankey node

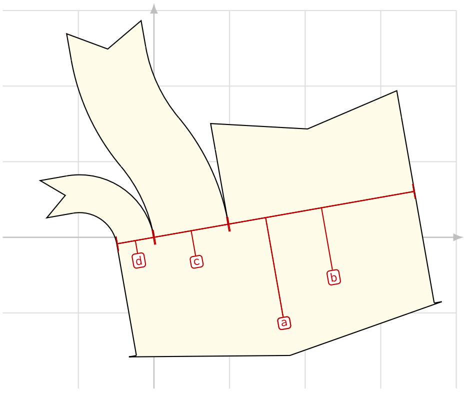
When creating a new Sankey node, the `forked` and `fork anchor` keys allow to fork the node directly *and* to anchor it on an anchor of a forked subnode.

`forked` (*Key*) `/sankey/node parameters/forked={⟨quantity/name pairs⟩}`

The ⟨`quantity/name pairs`⟩ is a comma separated list of *quantity/name* pairs (one for each subnode, from left to right). The sum of all quantities *must* be equal to the quantity of the new node to fork.

`fork anchor` (*Key*) `/sankey/node parameters/fork anchor=⟨node.anchor⟩`

An anchor belonging to the new node *or* belonging to a subnode (the anchor name must be prefixed by the name of the node). *Note:* when a `fork anchor` key is supplied, the `anchor` key is ignored (with a *warning* message).

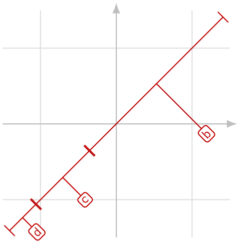


```
\begin{tikzpicture}
  \begin{sankeydiagram}
    \sankeyset{
      start style=arrow,end style=arrow,
      fill/.style={fill=yellow!10,line width=0pt,draw=yellow!10}
    }

    \sankeynode[debug]{
      name=a,quantity=40,angle=-80,
      forked={25/b,10/c,5/d},
      fork anchor=c.right,
    }

    \sankeyadvance*{b}{1cm}
    \sankeyturn*[minimum radius=2cm]{c}{-30}
    \sankeyturn*[minimum radius=2cm]{c}{30}
    \sankeyturn*[minimum radius=5mm]{d}{-90}
    \sankeyadvance{a}{1.5cm}
    \foreach \nodename in {b,c,d}{ \sankeystart{\nodename} }
    \sankeyend{a}
  \end{sankeydiagram}
\end{tikzpicture}
```

4.6.2 Fork a Sankey node

`\sankeyfork` `\sankeyfork[⟨options⟩]{⟨name⟩}{⟨quantity/name pairs⟩}`

The `\sankeyfork` macro splits the preexisting Sankey node named ⟨`name`⟩ in a list of new Sankey subnodes. The ⟨`quantity/name pairs`⟩ is a comma separated list of *quantity/name* pairs, one for each subnode from left to right. The sum of all quantities *must* be equal to the quantity of the node to fork.



```
\begin{tikzpicture}
  \begin{sankeydiagram}
    \sankeynode{name=a,quantity=40,angle=-45}
    \sankeyfork[debug]{a}{25/b,10/c,5/d}
    \path (a.left) rectangle (a.right); % create a bounding box
  \end{sankeydiagram}
\end{tikzpicture}
```

5 Miscellaneous

5.1 The `debug` layer

The options `debug` key uses the `sankeydebug` layer to draw above the `main` TikZ layer (via `\pgfsetlayers`, the `sankey` package installs four layers: `background`, `main`, `foreground`, `sankeydebug`).

The four following styles define how to display debug information:

```
\sankeyset{
  debug color/.style={/utils/exec={\colorlet{debug color}{#1}}},
  % debug color used by all debug macros
  debug color=red!75!black,
  % debug line between left and right anchors
  debug line/.style={overlay,draw=debug color,|-|},
  % debug line between center and label
  debug normal/.style={overlay,draw=debug color},
  % debug node label
  debug label/.style={
    overlay,
    draw,
    font=\ttfamily\tiny,
    text=debug color,text opacity=1,
    inner sep=.1em,
    fill=white,fill opacity=1,
    rounded corners=.1em,
    node contents={\name},
  },
}
```

5.2 The `dubins` TikZ library

The `sankey` package uses the `dubins` TikZ library (the `tikzlibrarydubins.code.tex` file) to compute Dubins paths. The documentation for this library does not yet exist.

5.3 How to duplicate a Sankey node

`\sankeynodealias` `\sankeynodealias{⟨origname⟩}{⟨clonename⟩}`

The `\sankeynodealias` macro clones the Sankey node named ⟨`origname`⟩ into the Sankey node named ⟨`clonename`⟩.

So, you can clone a Sankey node via two methods:

```
\sankeynode{name=a,quantity=10}
\sankeynode{as=a,name=b}
```

```
\sankeynode{name=a,quantity=10}
\sankeynodealias{a}{b}
```

5.4 How to define new start and end styles

Here are the definitions of the `arrow` styles:

```
\sankeyset{
  %% arrow style
  new start style={arrow}{
    (\name.left) -- ++(-10pt,0)
    -- ([xshift=-10pt/6]\name.center)
    -- ([xshift=-10pt]\name.right)
    -- (\name.right) -- cycle
  }{
    (\name.left) -- ++(-10pt,0)
    -- ([xshift=-10pt/6]\name.center)
    -- ([xshift=-10pt]\name.right)
    -- (\name.right)
  },
  new end style={arrow}{
    (\name.left) -- ([yshift=1mm]\name.left)
    -- ([xshift=10pt]\name.center)
    -- ([yshift=-1mm]\name.right) -- (\name.right) -- cycle
  }{
    (\name.left) -- ([yshift=1mm]\name.left)
    -- ([xshift=10pt]\name.center)
    -- ([yshift=-1mm]\name.right) -- (\name.right)
  },
}
```

6 Todo

- Document the `dubins` TikZ library.
- Add a tutorial.
- Add examples with cycle(s).

This manual contains three parts: User manual (p.2), Examples (p.22) and Installation & Implementation (p.43).

Part II

Examples

Contents

7 List of Figures

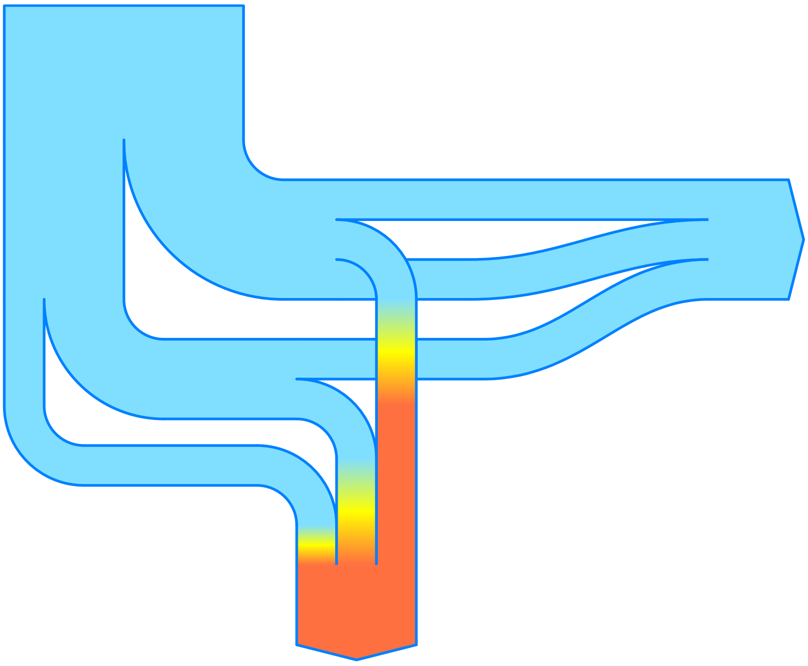


Figure 1: Simple example

8 Simple example

See figure 1. The `sankey-example1.tex` file contains the following code and is an attachement of the current PDF document.

```
\begin{tikzpicture}
  \begin{sankeydiagram}%[debug]
    \colorlet{cold}[rgb]{cyan!75!blue!50!white}
    \colorlet{hot}[rgb]{red!50!orange!75!white}
    \sankeyset{
      ratio=90pt/6,minimum radius=15pt,
      start style=simple,end style=simple,
      draw/.style={
        draw=blue!50!cyan,
        line width=1pt,line cap=round,line join=round,
      },
      cold/.style={
        fill/.style={
          draw=cold,line width=0pt,fill=cold,
        },
      },
      cold to hot/.style={
        fill/.style={
          fill=none,top color=cold,
          bottom color=hot,middle color=yellow,
        },
      },
      hot/.style={
        fill/.style={
          draw=hot,line width=0pt,fill=hot,
        },
      },
    }

    \sankeyset{cold}
    \sankeynodestart{name=p0,at={100,0},angle=-90,quantity=6}
    \sankeyadvance{p0}{50pt}
    \sankeyfork{p0}{3/p1,3/p2}
    \sankeyturnleft{p1}{90}
    \sankeyadvance{p1}{20pt}
    \sankeyadvance{p2}{60pt}
    \sankeyfork{p2}{2/p3,1/p4}
    \sankeyturnleft{p3}{90}
    \sankeyadvance{p3}{50pt}
    \sankeyfork{p3}{1/p5,1/p6}
    \sankeyadvance{p5}{70pt}
    \sankeyfork{p1}{1/p7,1/p8,1/p9}
    \sankeyadvance{p7}{50pt}
    \sankeyadvance{p9}{50pt}
    \sankeyadvance{p4}{40pt}
    \sankeyturnleft{p4}{90}
    \sankeyadvance{p4}{65pt}
    \sankeyadvance{p7}{40pt}
    \sankeynode{
      name=p11,at={[shift={(50pt,-15pt)}]p7},quantity=3,
      forked={1/p7a,1/p9a,1/p5a},
    }
    \sankeyoutin{p7}{p7a}
    \sankeyoutin{p9}{p9a}
    \sankeyoutin{p5}{p5a}
    \sankeyadvance{p11}{30pt}
    \sankeyend{p11}
    \sankeyturnright{p8}{90}
    \sankeyturnright{p6}{90}
    \sankeyturnright{p4}{90}
    \sankeyset{hot}
    \sankeyadvance[cold to hot]{p8}{40pt}
    \sankeynode{
      name=p10,at={[shift={(-15pt,-60pt)}]p8},angle=-90,quantity=3,
      forked={1/p8a,1/p6a,1/p4a},
    }
    \sankeyoutin[cold to hot]{p4}{p4a}
    \sankeyoutin[cold to hot]{p6}{p6a}
    \sankeyoutin{p8}{p8a}
    \sankeyadvance{p10}{30pt}
    \sankeyend{p10}
  \end{sankeydiagram}
\end{tikzpicture}
```

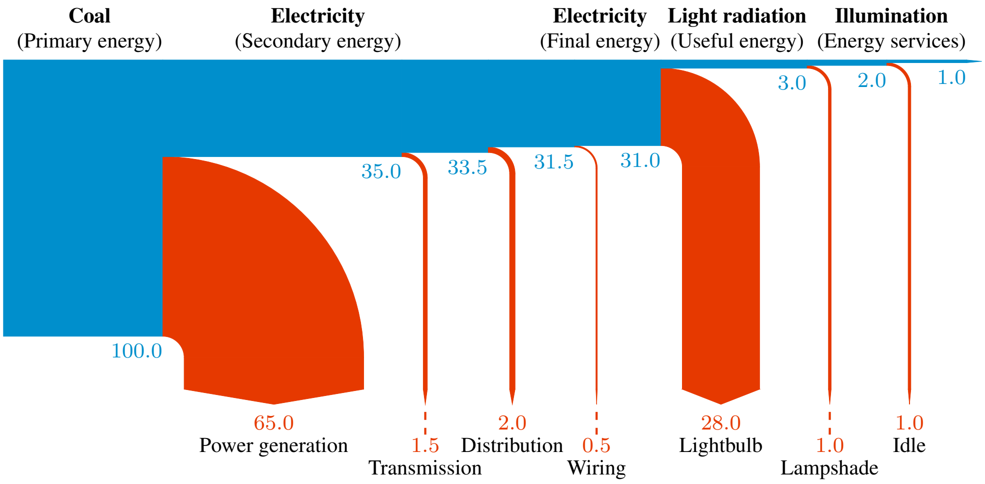


Figure 2: Energy diagram

9 Energy diagram

This example comes from IB Pysics Blog by Kyu Won Shim.

See figure 2. The `sankey-example-energy.tex` file contains the following code and is an attachement of the current PDF document.

```
\begin{tikzpicture}
  % font choice
  \renewcommand\rmdefault{txr}\rmfamily\footnotesize
  \sisetup{
    round-mode=places,
    round-precision=1,
    add-decimal-zero,
    round-pad=true,
  }
  \begin{sankeydiagram}
    \colorlet{energy}{blue!30!cyan!80!black}
    \colorlet{lost energy}{red!50!orange!90!black}
    \sankeyset{
      ratio=13em/100,
      minimum radius=1em,
      start style=simple,end style=simple,
      draw/.style={draw=none,line width=0},
      energy/.style={
        fill/.style={
          draw=energy,
          line width=0,
          fill=energy,
        }
      },
      lost energy/.style={
        fill/.style={
          draw=lost energy,
          line width=0,
          fill=lost energy,
        }
      }
    }

    \newcommand\abovelabel[2]{ % valname, label
      \node[anchor=south east,align=center,inner xsep=0] at (#1.left) {#2};
    }

    \newcommand\energylabel[1]{ % valname
      \node[anchor=north east,text=energy,inner xsep=0] at (#1.right)
      {\num{\sankeygetnodeqty{#1}}};
    }
```

```
\newcommand\lostenergylabel[2]{ % valname, label
  \node[anchor=north,text=lost energy] at ([yshift=-2.5mm]#1.center)
  (value)
  {\num{\sankeygetnodeqty{#1}}};
  \node[anchor=north,inner sep=0,align=center] at (value.south) {#2};
}

\newcommand\lostenergylabelbottom[2]{ % valname, label
  \draw[draw=lost energy,dashed,thick]
  ([yshift=-3mm]#1.center) coordinate (#1) -- ([yshift=-3mm]#1.center);
  \lostenergylabel{#1}{#2}
}

\sankeynode{name=Co,quantity=100.0}
\path (Co.right) ++(0,-7mm) coordinate (c);

\newcommand\turnandstop[2]{ % valname, label
  \begingroup
  \sankeyset{lost energy}
  \sankeyturnright{#1}{90}
  \sankeynode{as=#1,name=#1-stop,at={#1 |- c}}
  \sankeyoutin{#1}{#1-stop}
  \sankeynode{as=#1-stop,name=#1}
  \sankeyend{#1}
  \lostenergylabel{#1}{#2}
  \endgroup
}

\newcommand\turnandstopbottom[2]{ % valname, label
  \begingroup
  \sankeyset{lost energy}
  \sankeyturnright{#1}{90}
  \sankeynode{as=#1,name=#1-stop,at={#1 |- c}}
  \sankeyoutin{#1}{#1-stop}
  \sankeynode{as=#1-stop,name=#1}
  \sankeyend{#1}
  \lostenergylabelbottom{#1}{#2}
  \endgroup
}

\def\hshift{6.25em}

\sankeyadvance[energy]{Co}{1.2*\hshift}
\abovelabel{Co}{\textbf{Coal}\\(Primary energy)}
\energylabel{Co}
\sankeyfork{Co}{35/El1,65/Pg}
\turnandstop{Pg}{Power generation}

\sankeyadvance[energy]{El1}{1.8*\hshift}
\abovelabel{El1}{\textbf{Electricity}\\(Secondary energy)}
\energylabel{El1}
\sankeyfork{El1}{33.5/El2,1.5/Tr}
\turnandstopbottom{Tr}{Transmission}

\sankeyadvance[energy]{El2}{.65*\hshift}
\energylabel{El2}
\sankeyfork{El2}{31.5/El3,2.0/Di}
\turnandstop{Di}{Distribution}

\sankeyadvance[energy]{El3}{.65*\hshift}
\energylabel{El3}
\sankeyfork{El3}{31.0/El4,0.5/Wi}
\turnandstopbottom{Wi}{Wiring}

\sankeyadvance[energy]{El4}{.65*\hshift}
\abovelabel{El4}{\textbf{Electricity}\\(Final energy)}
\energylabel{El4}
\sankeyfork{El4}{3.0/Lr1,28.0/Lb}
```

```
    \turnandstop{Lb}{Lightbulb}

    \sankeyadvance[energy]{Lr1}{1.1*\hshift}
    \abovelabel{Lr1}{\textbf{Light radiation}\\(Useful energy)}
    \energylabel{Lr1}
    \sankeyfork{Lr1}{2.0/Lr2,1.0/Ls}
    \turnandstopbottom{Ls}{Lampshade}

    \sankeyadvance[energy]{Lr2}{.6*\hshift}
    \energylabel{Lr2}
    \sankeyfork{Lr2}{1.0/Il,1.0/Id}
    \turnandstop{Id}{Idle}

    \sankeyadvance[energy]{Il}{.6*\hshift}
    \abovelabel{Il}{\textbf{Illumination}\\(Energy services)}
    \energylabel{Il}
    \sankeyend[energy]{Il}
  \end{sankeydiagram}
\end{tikzpicture}
```

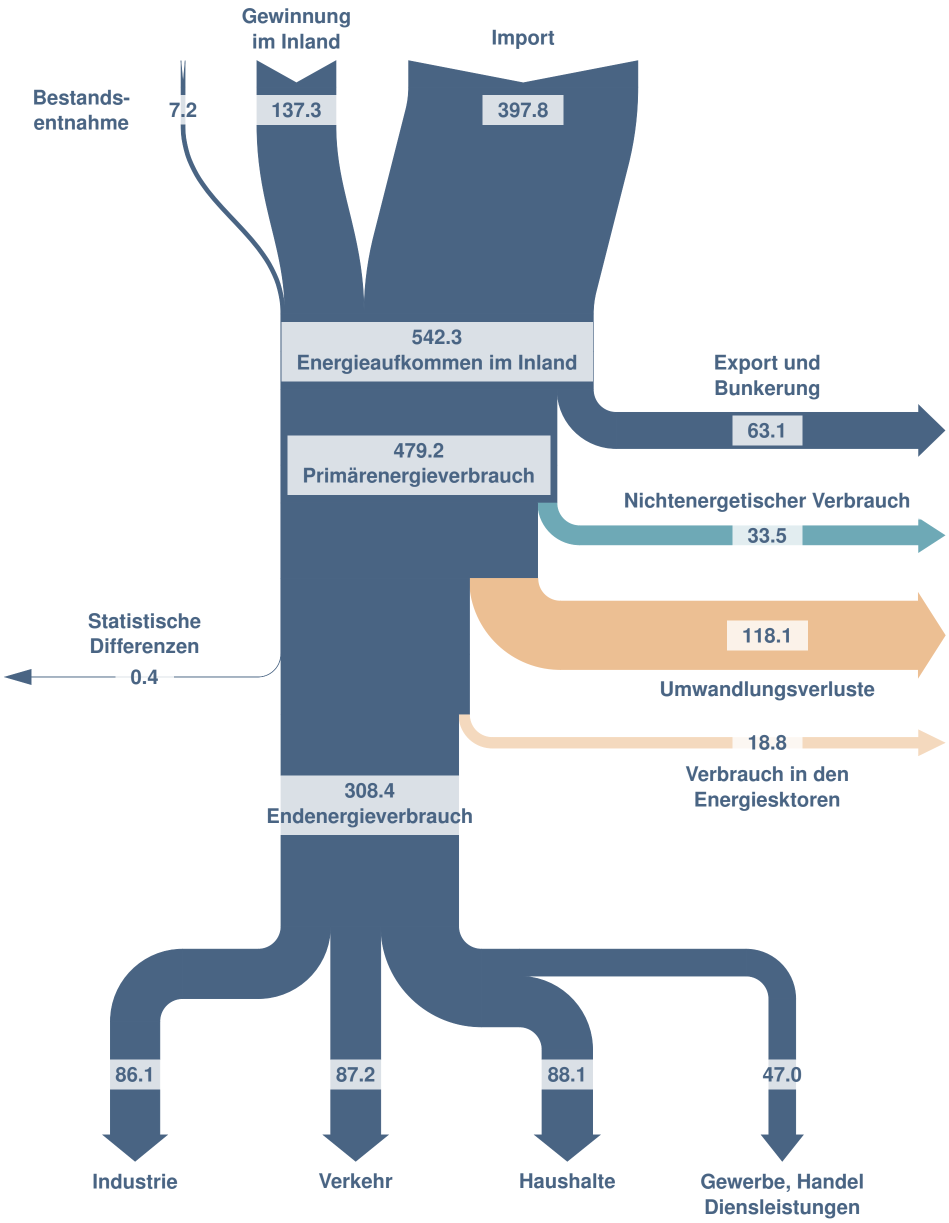


Figure 3: Example from TeX.se question

10 Example from question on TeX.se

This example came from this question on TeX.StackExchange.

See figure 3. The `sankey-example2.tex` file contains the following code and is an attachement of the current PDF document.

```
\begin{tikzpicture}
  \renewcommand*\sfdefault{txss}

  \begin{sankeydiagram}%[debug]
    \sankeyset{
      ratio=4cm/524.3,
      minimum radius=3mm,
      start style=arrow,
      end style=arrow,
      fill/.style={
        line width=0pt,
        fill=cyan!50!blue!50!black,
        draw=cyan!50!blue!50!black,
      },
      draw/.style={draw=none},
      every node/.style={angle=-90},
    }
```

```
\sankeynodestart{name=B,at={-.5,0},quantity=7.2}
\coordinate[below=1mm of B.center] (B label);
\sankeyadvance{B}{5mm}
\sankeynodestart{name=GI,at={1,0},quantity=137.3}
\coordinate[below=1mm of GI.center] (GI label);
\sankeyadvance{GI}{5mm}
\sankeynodestart{name=I,at={4,0},quantity=397.8}
\coordinate[below=1mm of I.center] (I label);
\sankeynode{
  name=EI,at={2.86,-3},quantity=542.3,
  forked={397.8/Ia,137.3/GIa,7.2/Ba}
}
\sankeydubins[minimum radius=1.2cm]{I}{Ia}
\sankeyoutin{GI}{GIa}
\sankeyoutin{B}{Ba}
\sankeyadvance{EI}{5mm}
\coordinate (EI label) at (EI);
\sankeyadvance{EI}{5mm}
\sankeyfork{EI}{63.1/EB,479.2/P}

\sankeyturnleft{EB}{90}
\sankeyadvance{EB}{4cm}
\coordinate (EB label) at ($(EB)!.5!(EB-old)$);
\sankeyend{EB}

\sankeyadvance{P}{10mm}
\coordinate (P label) at (P);
\sankeyadvance{P}{5mm}

\sankeyfork{P}{33.5/NV,445.7/P-NV}

{
  \colorlet{NV color}{cyan!80!lime!50!gray}
  \sankeyset{fill/.append style={fill=NV color,draw=NV color}}
  \sankeyturnleft{NV}{90}
  \sankeynode{as=NV,name=NV2,at=NV -| EB}
  \sankeyoutin{NV}{NV2}
  \coordinate (NV label) at (NV -| EB label);
  \sankeyend{NV2}
}

\sankeyadvance{P-NV}{10mm}
\sankeyfork{P-NV}{118.1/U,327.6/P-NV-U}

{
  \sankeyset{
    fill/.style={fill=orange!70!gray!50,draw=orange!70!gray!50}
  }
  \sankeyturnleft{U}{90}
  \sankeynode{as=U,name=U2,at=U -| EB}
  \sankeyoutin{U}{U2}
  \coordinate (U label) at (U -| EB label);
  \sankeyend{U2}
}

\sankeyadvance{P-NV-U}{10mm}
\sankeyfork{P-NV-U}{327.2/P-NV-U-SD,0.4/SD}

{
  \sankeyturnright{SD}{90}
  \sankeyadvance{SD}{15mm}
  \coordinate (SD label) at (SD);
  \sankeyadvance{SD}{15mm}
  \sankeyend{SD}
}

\sankeyadvance{P-NV-U-SD}{8mm}
\sankeyfork{P-NV-U-SD}{18.8/VE,308.4/E}
```

```
  {
    \sankeyset{fill/.append style={orange!70!gray!30}}
    \sankeyturnleft{VE}{90}
    \sankeynode{as=VE,name=VE2,at=VE -| EB}
    \sankeyoutin{VE}{VE2}
    \coordinate (VE label) at (VE -| EB label);
    \sankeyend{VE2}
  }

  \sankeyadvance{E}{8mm}
  \coordinate (E label) at (E);
  \sankeyadvance{E}{20mm}
  \sankeyfork{E}{135.1/H+GHD,87.2/V,86.1/In}

  \sankeyturnright{In}{90}
  \sankeyadvance{In}{10mm}
  \sankeyturnleft{In}{90}
  \sankeyadvance{In}{5mm}
  \coordinate (In label)  at (In);
  \sankeyadvance{In}{10mm}
  \sankeyend{In}

  \sankeynode{as=V,name=V2,at=V|-In label}
  \sankeyoutin{V}{V2}
  \coordinate (V label) at (V2);
  \sankeyadvance{V2}{10mm}
  \sankeyend{V2}

  \sankeyturnleft{H+GHD}{90}
  \sankeyadvance{H+GHD}{5mm}
  \sankeyfork{H+GHD}{47.0/GHD,88.1/H}

  \sankeyturnright{H}{90}
  \sankeynode{as=H,name=H2,at=H|-In label}
  \sankeyoutin{H}{H2}
  \coordinate (H label) at (H2);
  \sankeyadvance{H2}{10mm}
  \sankeyend{H2}

  \sankeyadvance{GHD}{30mm}
  \sankeyturnright{GHD}{90}
  \sankeynode{as=GHD,name=GHD2,at=GHD|-In label}
  \sankeyoutin{GHD}{GHD2}
  \coordinate (GHD label) at (GHD2);
  \sankeyadvance{GHD2}{10mm}
  \sankeyend{GHD2}
\end{sankeydiagram}

% labels
\tikzset{
  label/.style={
    fill=white,fill opacity=.8,text opacity=1,
    inner sep=1mm,
    text=cyan!50!blue!50!black,
    inner xsep=2mm,
    font=\sffamily\bfseries\footnotesize,
    align=center,
  },
}
\node[label,anchor=north] (B label) at (B label) {7.2};
\node[label,left=1mm of B label] {Bestands-\\entnahme};
\node[label,anchor=north] at (GI label) {137.3};
\node[label,above=5mm of GI label] {Gewinnung\\im Inland};
\node[label,anchor=north] at (I label) {397.8};
\node[label,above=5mm of I label] {Import};

\node[label] at (EI label) {542.3\\Energieaufkommen im Inland};
```

```
  \node[label,anchor=center] (EB label) at (EB label) {63.1};
  \node[label,above=1mm of EB label] {Export und\\Bunkerung};

  \node[label] at (P label) {479.2\\Primärenergieverbrauch};

  \node[label,anchor=center] (NV label) at (NV label) {33.5};
  \node[label,above=0mm of NV label] {Nichtenergetischer Verbrauch};

  \node[label,anchor=center] (U label) at (U label) {118.1};
  \node[label,below=3mm of U label] {Umwandlungsverluste};

  \node[label,anchor=center] (SD label) at (SD label) {0.4};
  \node[label,above=0mm of SD label] {Statistische\\Differenzen};

  \node[label,anchor=center] (VE label) at (VE label) {18.8};
  \node[label,below=0mm of VE label] {Verbrauch in den\\Energiesktoren};

  \node[label,anchor=north] (E label) at (E label)
  {308.4\\Endenergieverbrauch};

  \node[label,anchor=north] (In label) at (In label) {86.1};
  \node[label,anchor=north,below=1cm of In label] {Industrie};

  \node[label,anchor=north] (V label) at (V label) {87.2};
  \node[label,anchor=north,below=1cm of V label] {Verkehr};

  \node[label,anchor=north] (H label) at (H label) {88.1};
  \node[label,anchor=north,below=1cm of H label] {Haushalte};

  \node[label,anchor=north] (GHD label) at (GHD label) {47.0};
  \node[label,anchor=north,below=1cm of GHD label]
  {Gewerbe, Handel\\Diensleistungen};
\end{tikzpicture}
```

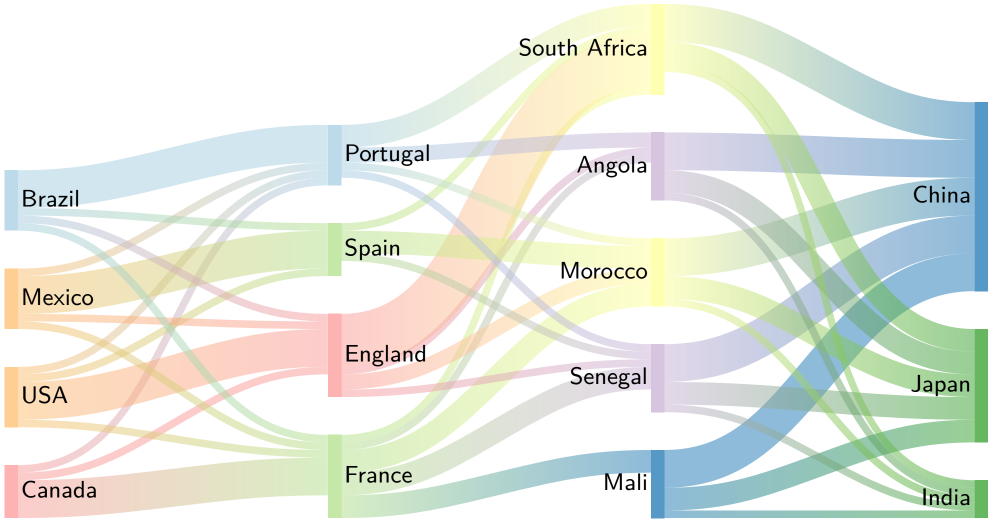


Figure 4: Reproduction of an example from Google Charts documentation

11 Reproduction of an example from Google Charts documentation

This example is a reproduction of an example of Google Charts Documentation[4].

See figure 4. The `sankey-example3.tex` file contains the following code and is an attachement of the current PDF document.

```
\begin{tikzpicture}
  \begin{sankeydiagram}%[debug]
    \sffamily
    \sankeyset{
      ratio=1cm/10,
      outin steps=2,
      draw/.style={draw=none,line width=0pt},
      color/.style={fill/.style={fill=#1,fill opacity=.75}},
      shade/.style 2 args={fill/.style={left color=#1,
          right color=#2,fill opacity=.5}},
      % colors
      @define HTML color/.code args={#1/#2}{\definecolor{#1}{HTML}{#2}},
      @define HTML color/.list={
        cyan/a6cee3,lime/b2df8a,red/fb9a99,orange/fdbf6f,
        violet/cab2d6,yellow/ffff99,blue/1f78b4,green/33a02c
      },
      % colors of countries
      @let country color/.code args={#1/#2}{\colorlet{#1}[rgb]{#2}},
      @let country color/.list={
        CA/red,US/orange,MX/orange,BR/cyan,FR/lime,GB/red,
        SP/lime,PT/cyan,ML/blue,SN/violet,MA/yellow,
        AO/violet,ZA/yellow,IN/green,JP/green,CN/blue
      },
    }
    \def\vdist{5mm}
    \def\hwidth{.5em}
    \def\hdist{4.1cm}

    \sankeynode{name=CA,quantity=7}
    \sankeynode{name=US,quantity=8,at={[yshift=\vdist]CA.left},anchor=right}
    \sankeynode{name=MX,quantity=8,at={[yshift=\vdist]US.left},anchor=right}
    \sankeynode{name=BR,quantity=8,at={[yshift=\vdist]MX.left},anchor=right}

    \foreach \country in {CA,US,MX,BR}{
      \sankeyadvance[color=\country]{\country}{\hwidth}
    }
```

---

[4] https://developers.google.com/chart/interactive/docs/gallery/sankey

```
\sankeyfork{CA}{1/CA-to-PT,1/CA-to-GB,5/CA-to-FR}
\sankeyfork{US}{1/US-to-PT,1/US-to-SP,5/US-to-GB,1/US-to-FR}
\sankeyfork{MX}{1/MX-to-PT,5/MX-to-SP,1/MX-to-GB,1/MX-to-FR}
\sankeyfork{BR}{5/BR-to-PT,1/BR-to-SP,1/BR-to-GB,1/BR-to-FR}

\sankeynode{name=FR,quantity=11,
  at={[xshift=\hdist]CA.right},anchor=right}
\sankeynode{name=GB,quantity=11,
  at={[yshift=\vdist]FR.left},anchor=right}
\sankeynode{name=SP,quantity=7,
  at={[yshift=\vdist]GB.left},anchor=right}
\sankeynode{name=PT,quantity=8,
  at={[yshift=\vdist]SP.left},anchor=right}

\sankeyfork{FR}
{1/FR-from-BR,1/FR-from-MX,1/FR-from-US,5/FR-from-CA,3/FR-from-00}
\sankeyfork{GB}
{1/GB-from-BR,1/GB-from-MX,5/GB-from-US,1/GB-from-CA,3/GB-from-00}
\sankeyfork{SP}{1/SP-from-BR,5/SP-from-MX,1/SP-from-US}
\sankeyfork{PT}{5/PT-from-BR,1/PT-from-MX,1/PT-from-US,1/PT-from-CA}

\foreach \country in {FR,GB,SP,PT}{
  \sankeyadvance[color=\country]{\country}{\hwidth}
}

\sankeyfork{FR}{1/FR-to-ZA,1/FR-to-AO,3/FR-to-MA,3/FR-to-SN,3/FR-to-ML}
\sankeyfork{GB}{7/GB-to-ZA,1/GB-to-AO,2/GB-to-MA,1/GB-to-SN}
\sankeyfork{SP}{1/SP-to-ZA,3/SP-to-MA,1/SP-to-SN,2/SP-to-00}
\sankeyfork{PT}{3/PT-to-ZA,2/PT-to-AO,1/PT-to-MA,1/PT-to-SN,1/PT-to-00}

\sankeynode{name=ML,quantity=9,
  at={[xshift=\hdist]FR.right},anchor=right}
\sankeynode{name=SN,quantity=9,
  at={[yshift=\vdist]ML.left},anchor=right}
\sankeynode{name=MA,quantity=9,
  at={[yshift=\vdist]SN.left},anchor=right}
\sankeynode{name=AO,quantity=9,
  at={[yshift=\vdist]MA.left},anchor=right}
\sankeynode{name=ZA,quantity=12,
  at={[yshift=\vdist]AO.left},anchor=right}

\sankeyfork{ML}{3/ML-from-FR,6/Mail-from-00}
\sankeyfork{SN}
{1/SN-from-PT,1/SN-from-SP,1/SN-from-GB,3/SN-from-FR,3/SN-from-00}
\sankeyfork{MA}{1/MA-from-PT,3/MA-from-SP,2/MA-from-GB,3/MA-from-FR}
\sankeyfork{AO}{2/AO-from-PT,1/AO-from-GB,1/AO-from-FR,5/AO-from-00}
\sankeyfork{ZA}{3/ZA-from-PT,1/ZA-from-SP,7/ZA-from-GB,1/ZA-from-FR}

\foreach \country in {ML,SN,MA,AO,ZA}{
  \sankeyadvance[color=\country]{\country}{\hwidth}
}

\sankeyfork{ML}{5/ML-to-CN,3/ML-to-JP,1/ML-to-IN}
\sankeyfork{SN}{5/SN-to-CN,3/SN-to-JP,1/SN-to-IN}
\sankeyfork{MA}{5/MA-to-CN,3/MA-to-JP,1/MA-to-IN}
\sankeyfork{AO}{5/AO-to-CN,3/AO-to-JP,1/AO-to-IN}
\sankeyfork{ZA}{5/ZA-to-CN,3/ZA-to-JP,1/ZA-to-IN,3/ZA-to-00}

\sankeynode{name=IN,quantity=5,
  at={[xshift=\hdist]ML.right},anchor=right}
\sankeynode{name=JP,quantity=15,
  at={[yshift=\vdist]IN.left},anchor=right}
\sankeynode{name=CN,quantity=25,
  at={[yshift=\vdist]JP.left},anchor=right}

\sankeyfork{IN}
{1/IN-from-ZA,1/IN-from-AO,1/IN-from-MA,1/IN-from-SN,1/IN-from-ML}
\sankeyfork{JP}
{3/JP-from-ZA,3/JP-from-AO,3/JP-from-MA,3/JP-from-SN,3/JP-from-ML}
```

```
    \sankeyfork{CN}
    {5/CN-from-ZA,5/CN-from-AO,5/CN-from-MA,5/CN-from-SN,5/CN-from-ML}

    \foreach \country in {IN,JP,CN}{
      \sankeyadvance[color=\country]{\country}{\hwidth}
    }

    \foreach \startcountry/\countries in {
      CA/{PT,GB,FR},    US/{PT,SP,GB,FR},    MX/{PT,SP,GB,FR},
      BR/{PT,SP,GB,FR}, FR/{ML,SN,MA,AO,ZA}, GB/{SN,MA,AO,ZA},
      SP/{SN,MA,ZA},    PT/{SN,MA,AO,ZA},    ML/{IN,JP,CN},
      SN/{IN,JP,CN},    MA/{IN,JP,CN},       AO/{IN,JP,CN},
      ZA/{IN,JP,CN}}
    {
      \foreach \endcountry in \countries {
        \sankeyoutin[shade={\startcountry}{\endcountry}]
        {\startcountry-to-\endcountry}{\endcountry-from-\startcountry}
      }
    }

    \foreach \country/\countryname in {CA/Canada, US/USA, MX/Mexico,
      BR/Brazil, FR/France, GB/England, SP/Spain, PT/Portugal}
    {
      \node[anchor=west,inner sep=.1em,font=\small]
      at (\country) {\countryname\vphantom{Ag}};
    }

    \foreach \country/\countryname in {
      ML/Mali, SN/Senegal, MA/Morocco, AO/Angola,
      ZA/South Africa, IN/India, JP/Japan, CN/China}
    {
      \node[anchor=east,inner sep=.1em,font=\small]
      at (\country-old) {\countryname\vphantom{Ag}};
    }
  \end{sankeydiagram}
\end{tikzpicture}
```

11.1 Variation

Here is a variation of the previous example using the `rotate` key.

See figure 5 on the next page. The `sankey-example3-variation.tex` file contains the following code and is an attachement of the current PDF document.

```
\begin{tikzpicture}[rotate=-90]
  \begin{sankeydiagram}[rotate=-90]
    \sffamily
    \sankeyset{
      ratio=1.7cm/10,
      outin steps=2,
      start style=arrow,
      end style=simple,
      draw/.style={draw=white,line width=.4pt},
      color/.style={fill/.style={fill=#1,fill opacity=.75}},
      shade/.style 2 args={fill/.style={
          fill=none,line width=0,
          top color=#1,bottom color=#2,
          middle color=#1!50!#2!50!white,
          fill opacity=.75}},
      % colors
      @define HTML color/.code args={#1/#2}{\definecolor{#1}{HTML}{#2}},
      @define HTML color/.list={
        cyan/a6cee3,lime/b2df8a,red/fb9a99,orange/fdbf6f,
        violet/cab2d6,yellow/ffff99,blue/1f78b4,green/33a02c
      },
      % colors of countries
      @let country color/.code args={#1/#2}{\colorlet{#1}[rgb]{#2}},
      @let country color/.list={
        CA/red,US/orange,MX/lime,BR/violet,FR/yellow,GB/blue,
```

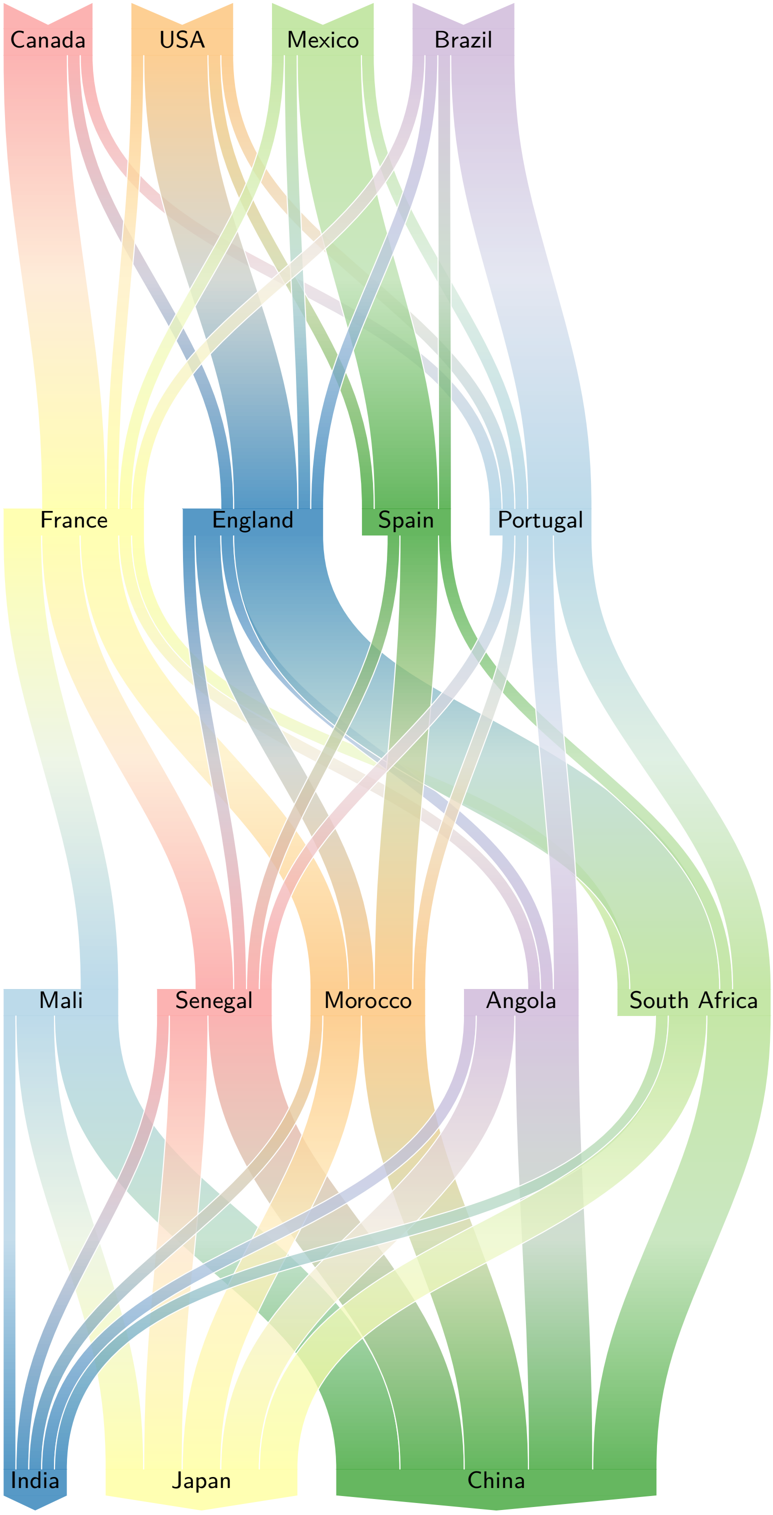


Figure 5: Reproduction of an example from Google Charts documentation – variation using the `rotate` key.

```
    SP/green,PT/cyan,ML/cyan,SN/red,MA/orange,
    AO/violet,ZA/lime,IN/blue,JP/yellow,CN/green
  },
}
\def\vdist{5mm}
\def\hwidth{1em}
\def\hdist{6cm}

\sankeynode{name=CA,quantity=7}
\sankeynode{name=US,quantity=8,at={[yshift=\vdist]CA.left},anchor=right}
\sankeynode{name=MX,quantity=8,at={[yshift=\vdist]US.left},anchor=right}
\sankeynode{name=BR,quantity=8,at={[yshift=\vdist]MX.left},anchor=right}

\foreach \country in {CA,US,MX,BR}{
  \sankeystart[color=\country]{\country}
  \sankeyadvance[color=\country]{\country}{\hwidth}
}

\sankeyfork{CA}{1/CA-to-PT,1/CA-to-GB,5/CA-to-FR}
\sankeyfork{US}{1/US-to-PT,1/US-to-SP,5/US-to-GB,1/US-to-FR}
\sankeyfork{MX}{1/MX-to-PT,5/MX-to-SP,1/MX-to-GB,1/MX-to-FR}
\sankeyfork{BR}{5/BR-to-PT,1/BR-to-SP,1/BR-to-GB,1/BR-to-FR}

\sankeynode{name=FR,quantity=11,
  at={[xshift=\hdist]CA.right},anchor=right}
\sankeynode{name=GB,quantity=11,
  at={[yshift=\vdist]FR.left},anchor=right}
\sankeynode{name=SP,quantity=7,
  at={[yshift=\vdist]GB.left},anchor=right}
\sankeynode{name=PT,quantity=8,
  at={[yshift=\vdist]SP.left},anchor=right}

\sankeyfork{FR}
{1/FR-from-BR,1/FR-from-MX,1/FR-from-US,5/FR-from-CA,3/FR-from-00}
\sankeyfork{GB}
{1/GB-from-BR,1/GB-from-MX,5/GB-from-US,1/GB-from-CA,3/GB-from-00}
\sankeyfork{SP}{1/SP-from-BR,5/SP-from-MX,1/SP-from-US}
\sankeyfork{PT}{5/PT-from-BR,1/PT-from-MX,1/PT-from-US,1/PT-from-CA}

\foreach \country in {FR,GB,SP,PT}{
  \sankeyadvance[color=\country]{\country}{\hwidth}
}

\sankeyfork{FR}{1/FR-to-ZA,1/FR-to-AO,3/FR-to-MA,3/FR-to-SN,3/FR-to-ML}
\sankeyfork{GB}{7/GB-to-ZA,1/GB-to-AO,2/GB-to-MA,1/GB-to-SN}
\sankeyfork{SP}{1/SP-to-ZA,3/SP-to-MA,1/SP-to-SN,2/SP-to-00}
\sankeyfork{PT}{3/PT-to-ZA,2/PT-to-AO,1/PT-to-MA,1/PT-to-SN,1/PT-to-00}

\sankeynode{name=ML,quantity=9,
  at={[xshift=\hdist]FR.right},anchor=right}
\sankeynode{name=SN,quantity=9,
  at={[yshift=\vdist]ML.left},anchor=right}
\sankeynode{name=MA,quantity=9,
  at={[yshift=\vdist]SN.left},anchor=right}
\sankeynode{name=AO,quantity=9,
  at={[yshift=\vdist]MA.left},anchor=right}
\sankeynode{name=ZA,quantity=12,
  at={[yshift=\vdist]AO.left},anchor=right}

\sankeyfork{ML}{3/ML-from-FR,6/Mail-from-00}
\sankeyfork{SN}
{1/SN-from-PT,1/SN-from-SP,1/SN-from-GB,3/SN-from-FR,3/SN-from-00}
\sankeyfork{MA}{1/MA-from-PT,3/MA-from-SP,2/MA-from-GB,3/MA-from-FR}
\sankeyfork{AO}{2/AO-from-PT,1/AO-from-GB,1/AO-from-FR,5/AO-from-00}
\sankeyfork{ZA}{3/ZA-from-PT,1/ZA-from-SP,7/ZA-from-GB,1/ZA-from-FR}

\foreach \country in {ML,SN,MA,AO,ZA}{
  \sankeyadvance[color=\country]{\country}{\hwidth}
}
```

```
    \sankeyfork{ML}{5/ML-to-CN,3/ML-to-JP,1/ML-to-IN}
    \sankeyfork{SN}{5/SN-to-CN,3/SN-to-JP,1/SN-to-IN}
    \sankeyfork{MA}{5/MA-to-CN,3/MA-to-JP,1/MA-to-IN}
    \sankeyfork{AO}{5/AO-to-CN,3/AO-to-JP,1/AO-to-IN}
    \sankeyfork{ZA}{5/ZA-to-CN,3/ZA-to-JP,1/ZA-to-IN,3/ZA-to-00}

    \sankeynode{name=IN,quantity=5,
      at={[xshift=\hdist]ML.right},anchor=right}
    \sankeynode{name=JP,quantity=15,
      at={[yshift=\vdist]IN.left},anchor=right}
    \sankeynode{name=CN,quantity=25,
      at={[yshift=\vdist]JP.left},anchor=right}

    \sankeyfork{IN}
    {1/IN-from-ZA,1/IN-from-AO,1/IN-from-MA,1/IN-from-SN,1/IN-from-ML}
    \sankeyfork{JP}
    {3/JP-from-ZA,3/JP-from-AO,3/JP-from-MA,3/JP-from-SN,3/JP-from-ML}
    \sankeyfork{CN}
    {5/CN-from-ZA,5/CN-from-AO,5/CN-from-MA,5/CN-from-SN,5/CN-from-ML}

    \foreach \country in {IN,JP,CN}{
      \sankeyadvance[color=\country]{\country}{\hwidth}
      \sankeyend[color=\country]{\country}
    }

    \foreach \startcountry/\countries in {
      CA/{PT,GB,FR},    US/{PT,SP,GB,FR},    MX/{PT,SP,GB,FR},
      BR/{PT,SP,GB,FR}, FR/{ML,SN,MA,AO,ZA}, GB/{SN,MA,AO,ZA},
      SP/{SN,MA,ZA},    PT/{SN,MA,AO,ZA},    ML/{IN,JP,CN},
      SN/{IN,JP,CN},    MA/{IN,JP,CN},       AO/{IN,JP,CN},
      ZA/{IN,JP,CN}}
    {
      \foreach \endcountry in \countries {
        \sankeyoutin[shade={\startcountry}{\endcountry}]
        {\startcountry-to-\endcountry}{\endcountry-from-\startcountry}
      }
    }

    \foreach \country/\countryname in {CA/Canada, US/USA, MX/Mexico,
      BR/Brazil, FR/France, GB/England, SP/Spain, PT/Portugal}
    {
      \node[anchor=south,inner sep=.1em,font=\small]
      at (\country) {\countryname\vphantom{Ag}};
    }

    \foreach \country/\countryname in {
      ML/Mali, SN/Senegal, MA/Morocco, AO/Angola,
      ZA/South Africa, IN/India, JP/Japan, CN/China}
    {
      \node[anchor=south,inner sep=.1em,font=\small]
      at (\country) {\countryname\vphantom{Ag}};
    }
  \end{sankeydiagram}
\end{tikzpicture}
```

12 Very nice example – Nadieh Bremer creation

Graphic designer Nadieh Bremer created this very nice Sankey diagram[5] for Adyen's second half 2018 report to shareholders. It is coded here with her kind permission.

You can change the data values between lines 27 and 52.

See figure 6 on the following page. The `sankey-example4.tex` file contains the following code and is an attachement of the current PDF document.

```
 1 \begin{tikzpicture}
 2   \renewcommand*\sfdefault{txss}
 3   \sffamily
 4   \sisetup{
 5     detect-all=true,
 6     group-separator={,},
 7     group-minimum-digits=4,
 8   }
 9   % storage of labels
10   \newcommand\LabSet[2]{% node name, label
11     \expandafter\edef\csname#1@Lab\endcsname{#2}}
12   \newcommand\Lab[1]{% node name
13     \csname#1@Lab\endcsname}
14   % storage of quantities
15   \newcommand\QtySet[2]{% node name, quantity
16     \expandafter\edef\csname#1@Qty\endcsname{\fpeval{#2}}}
17   \newcommand\Qty[1]{% node name
18     \csname#1@Qty\endcsname}
19   % all nodes with their name, label and quantity
20   \sankeyset{
21     def data/.code args={#1/#2/#3}{% node name/label/values
22       \LabSet{#1}{#2}
23       \QtySet{#1}{#3}
24       \typeout{#1: \Qty{#1}€ (\Lab{#1})}
25     },
26     def data/.list={
27       {Pf/Processing\\fees/71713},
28       {Sog/Sales of\\good/4547},
29       {Sf/Settlement\\fees/842075},
30       {Os/Other\\services/37532},
31       {R/Revenues/\Qty{Pf}+\Qty{Sog}+\Qty{Sf}+\Qty{Os}},
32       {Coi/Cost of Inventory/5151},
33       {Ciffi/Cost insecure from financial institutions/758234},
34       {Nr/Net revenue/\Qty{R}-\Qty{Coi}-\Qty{Ciffi}},
35       {Aadotaifa/Amortization and\\depreciation of tangible and\\
36         intangible fixed assets/4688},
37       {Ssapc/Social securities and\\pension costs/7860},
38       {Was/Wages and salaries/35627},
39       {Ooe/Other operating expenses/37346},
40       {Nr2/-/\Qty{Nr}-\Qty{Aadotaifa}-\Qty{Ssapc}-\Qty{Was}-\Qty{Ooe}},
41       {Oi/Other income/47},
42       {Ibiiieait/Income before interest income,\\interest expense and
43         income taxes/\Qty{Nr2}+\Qty{Oi}},
44       {Fe/Finance expense/561},
45       {Ofr/Other financial results/2533},
46       {Ibiiieait2/-/\Qty{Ibiiieait}-\Qty{Fe}-\Qty{Ofr}},
47       {Fi/Finance income/204},
48       {Ibit/Income before income taxes/\Qty{Ibiiieait2}+\Qty{Fi}},
49       {It/Income taxes/21134},
50       {Niftp/Net income for the period/\Qty{Ibit}-\Qty{It}},
51       {Octa/Other currency\\translation adjustments/785},
52       {Tci/Total comprehensive income/\Qty{Niftp}+\Qty{Octa}}
53     },
54   }
55 
56   \definecolor{mygreen}{RGB}{9,192,82}
57   \tikzset{
58     cost node/.style={
```

---

[5] https://www.visualcinnamon.com/portfolio/adyen-report-2019/

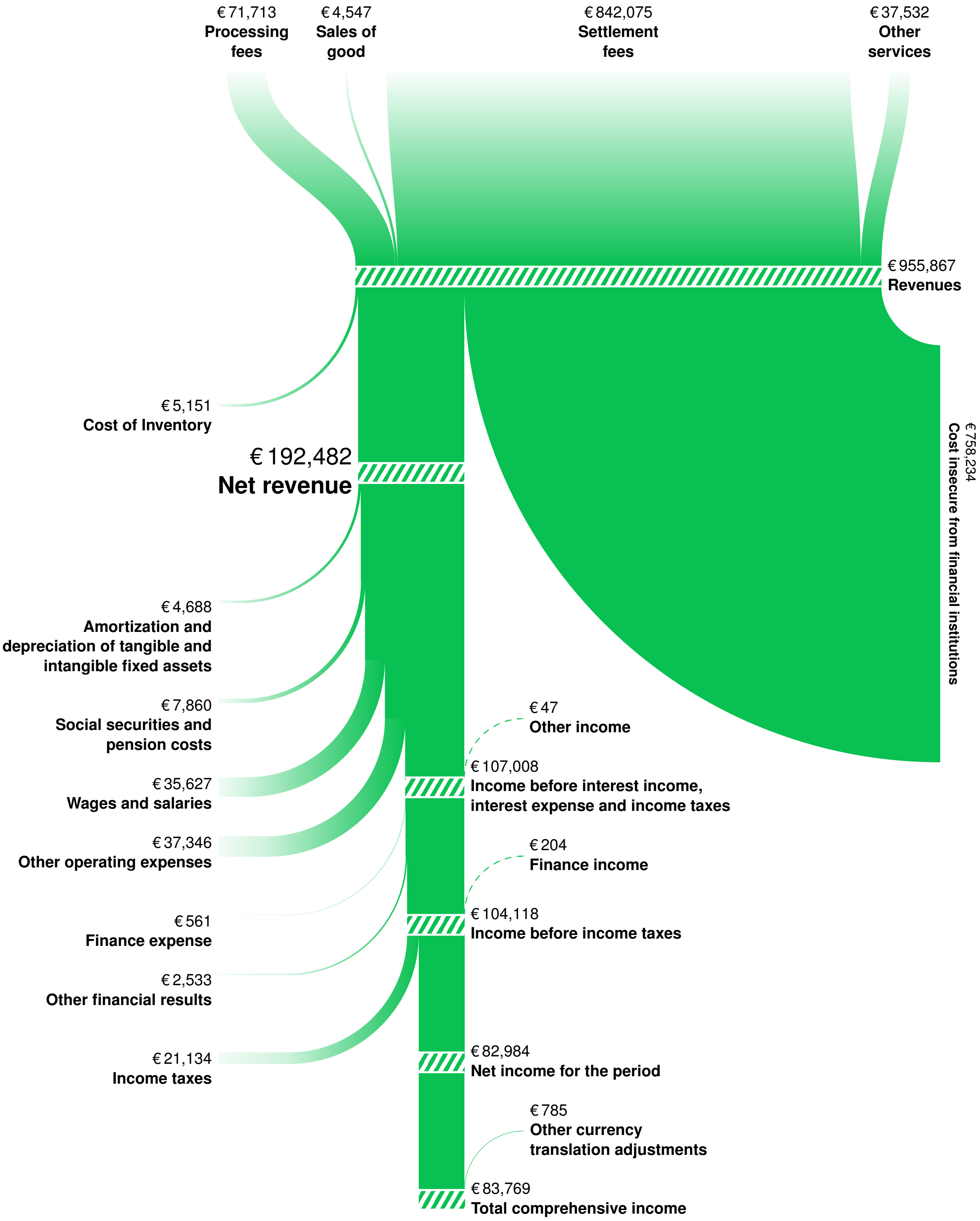


Figure 6: Very nice example – Nadieh Bremer's creation (from Adyen's Shareholder Report)

```
    overlay,
    align=flush center,
    node font=\footnotesize\sffamily\bfseries,
    inner sep=0,
    node contents={%
      {\mdseries€\,\num{\Qty{#1}}}\\
      \Lab{#1}\vphantom{g}%
    },
  },
  white hash/.style={
    draw=none,fill=none,
    pattern={Lines[angle=60,line width=2pt,distance=4pt]},
    pattern color=white,
  },
  line sep/.style={draw=white,line width=1pt},
  left label/.style={left=#1,align=flush right,anchor=north east},
  right label/.style={right=#1,align=flush left},
  right label hashed/.style={
    right=1mm of $(#1.left)!.5!(#1-old.left)$,align=flush left,
  },
  left label hashed/.style={
    left=1mm of $(#1.right)!.5!(#1-old.right)$,align=flush right,
  },
}

\newcommand\turnandstop[1]{
  \sankeyturn[green to greenwhite]{#1}{-90}
  \sankeynode{as=#1,name=#1-e,at={#1 -| Coi}}
  \sankeyoutin[greenwhite to white]{#1}{#1-e}
  \node[cost node=#1,left label={1mm of #1-e.right}];
}

\begin{sankeydiagram}%[debug]
  \sankeyset{
    ratio=28em/1000000,
    minimum radius=2cm,
    start style=none,
    every node/.style={angle=-90},
    % default fill and draw styles
    fill/.style={
      line width=0pt,
      fill=mygreen,
    },
    draw/.style={draw=none},
    % specific fill and draw styles
    green to greenwhite/.style={
      fill/.style={
        line width=0pt,
        right color=mygreen,
        left color=mygreen!20!white,
      }
    },
    greenwhite to white/.style={
      fill/.style={
        line width=0pt,
        right color=mygreen!20!white,
        left color=mygreen!5!white,
      }
    },
    dashed/.style={draw/.style={draw=mygreen,dashed}},
  }

  \coordinate (top) at (0,2em);

  \sankeynodestart{name=Pf,quantity=\Qty{Pf}}
  \node[cost node=Pf,above=.5em of Pf.center];

  \sankeynodestart{name=Sog,quantity=\Qty{Sog},
    at={[xshift=4em]Pf.left},anchor=right}
  \node[cost node=Sog,above=.5em of Sog.center];

  \sankeynodestart{name=Sf,quantity=\Qty{Sf},
    at={[xshift=2em]Sog.left},anchor=right}
  \node[cost node=Sf,above=.5em of Sf.center];

  \sankeynodestart{name=Os,quantity=\Qty{Os},
    at={[xshift=2em]Sf.left},anchor=right}
  \node[cost node=Os,above=.5em of Os.center];

  \sankeynode{
    name=R,quantity=\Qty{R},at={[yshift=-10em]Sf.center},
    forked={\Qty{Os}/Os-a,\Qty{Sf}/Sf-a,\Qty{Sog}/Sog-a,\Qty{Pf}/Pf-a},
  }

  \foreach \nodename in {Pf,Sog,Sf,Os}{
    \sankeyoutin[fill/.style={top color=white,bottom color=mygreen}]
    {\nodename}{\nodename-a}
  }

  \sankeyadvance{R}{1em}
  \node[cost node=R,right label hashed=R];

  \sankeyfork{R}{\Qty{Ciffi}/Ciffi,\Qty{Nr}/Nr,\Qty{Coi}/Coi}

  \sankeyturnleft[minimum radius=1.cm]{Ciffi}{90}
  \node[cost node=Ciffi,at={([shift={(1mm,0)}]Ciffi.center)},rotate=-90,
  anchor=south,align=flush left,node font=\scriptsize\sffamily\bfseries];

  \sankeyturnright[green to greenwhite]{Coi}{90}
  \sankeyadvance[greenwhite to white]{Coi}{1em}
  \node[cost node=Coi,left={1mm of [yshift=.75ex]Coi.left},
  align=flush right,anchor=north east,overlay];

  \sankeyadvance{Nr}{9em}
  \sankeyadvance{Nr}{1em}
  \node[cost node=Nr,left label hashed=Nr,
  node font=\large\sffamily\bfseries];

  \sankeyfork{Nr}{\Qty{Nr2}/Nr2,\Qty{Ooe}/Ooe,
    \Qty{Was}/Was,\Qty{Ssapc}/Ssapc,\Qty{Aadotaifa}/Aadotaifa}

  \turnandstop{Aadotaifa}

  \sankeyadvance{Ssapc}{5em}
  \turnandstop{Ssapc}

  \sankeyadvance{Was}{9em}
  \turnandstop{Was}

  \sankeyadvance{Ooe}{12em}
  \turnandstop{Ooe}

  \sankeyadvance{Nr2}{15em}
  \sankeynode{name=Ibiiieait,quantity=\Qty{Ibiiieait},
    anchor=right,at={Nr2.right},
    forked={\Qty{Oi}/Oi,\Qty{Nr2}/Nr2-e}}

  \sankeyturnleftbackward[minimum radius=1cm,dashed]{Oi}{90}
  \node[cost node=Oi,right label=1mm of Oi.left];

  \sankeyadvance{Ibiiieait}{1em}
  \node[cost node=Ibiiieait,right label hashed=Ibiiieait];

  \sankeyfork{Ibiiieait}
  {\Qty{Ibiiieait2}/Ibiiieait2,\Qty{Ofr}/Ofr,\Qty{Fe}/Fe}

  \turnandstop{Fe}

  \sankeyadvance{Ofr}{3em}
  \turnandstop{Ofr}

  \sankeyadvance{Ibiiieait2}{6em}
  \sankeynode{name=Ibit,quantity={\Qty{Ibiiieait2}+\Qty{Fi}},
    anchor=right,at={Ibiiieait2.right},
    forked={\Qty{Fi}/Fi,\Qty{Ibiiieait2}/Ibiiieait2-e}}

  \sankeyturnleftbackward[minimum radius=1cm,dashed]{Fi}{90}
  \node[cost node=Fi,right label=1mm of Fi.left];

  \sankeyadvance{Ibit}{1em}
  \node[cost node=Ibit,right label hashed=Ibit];

  \sankeyfork{Ibit}{\Qty{Niftp}/Niftp,\Qty{It}/It}

  \turnandstop{It}

  \sankeyadvance{Niftp}{6em}

  \sankeyadvance{Niftp}{1em}
  \node[cost node=Niftp,right label hashed=Niftp];

  \sankeynode{name=Tci,quantity=\Qty{Niftp}+\Qty{Octa},
    anchor=right,at={[yshift=-6em]Niftp.right},
    forked={\Qty{Octa}/Octa,\Qty{Niftp}/Niftp-e}}
  \sankeyoutin{Niftp}{Niftp-e}

  \sankeyturnleftbackward[minimum radius=1cm]{Octa}{90}
  \node[cost node=Octa,right label=1mm of Octa.left];

  \sankeyadvance{Tci}{1em}
  \node[cost node=Tci,right label hashed=Tci];

  \newcommand\hashband[1]{
    \draw[line sep] (#1-old.right) -- (#1-old.left);
    \draw[line sep] (#1.right) -- (#1.left);
    \path[white hash] (#1-old.right) rectangle (#1.left);
  }

  \foreach \nodename in {R,Nr,Nr,Ibiiieait,Ibit,Niftp,Tci}{
    \hashband{\nodename}
  }

\end{sankeydiagram}
\end{tikzpicture}
```

This manual contains three parts: User manual (p.2), Examples (p.22) and Installation & Implementation (p.43).

Part III

Installation & Implementation

Contents

13 Compiling sankey

To produce the `sankey` package:

```
pdflatex sankey.ins # or 'latex sankey.ins'
```

To finish the installation you have to move the `tikzlibrarydubins.code.tex` and `sankey.sty` files into a directory searched by LaTeX.

To compile the `sankey` documentation (the `sankey.pdf` file):

```
pdflatex sankey.dtx
makeindex -s gind.ist -o sankey.ind sankey.idx
pdflatex sankey.dtx
makeindex -s gind.ist -o sankey.ind sankey.idx
pdflatex sankey.dtx
pdflatex sankey.dtx
```

14 The `sankey.sty` file

Poorly commented source code...

Version information:

```
\NeedsTeXFormat{LaTeX2e}[2015/10/01]
\ProvidesPackage{sankey}[2025/01/10 v3.0.2 to draw Sankey diagrams]
```

All required packages and TikZ libraries:

```
\RequirePackage{xparse}
\RequirePackage{etoolbox}
\RequirePackage{xfp}
\RequirePackage{tikz}
\usetikzlibrary{
  calc,
  decorations.markings,
  dubins
}
```

Declarations of PGF layers (to debug Sankey diagrams):

```
%% add a new layer to debug sankey diagrams
\pgfdeclarelayer{background}
\pgfdeclarelayer{foreground}
\pgfdeclarelayer{sankeydebug}
\pgfsetlayers{background,main,foreground,sankeydebug}
```

14.1 Fields

`\snk@newfield`

The `sankeynewfield` macro defines setter and getter macros for *key/value* pairs. It requires five parameters: the *def* macro used to store a new value, the *setter* macro name, the *getter* macro name, the *cs name* used by the new field (including a `#1` parameter – the *key*) and the *error message* (used by the getter macro if the key is not defined).

The *setter* macro requires two parameters: the key and the value. The *getter* macro requires one parameter: the key.

```
\def\snk@newfield#1#2#3#4#5{
  % setter
  \def#2##1##2{\expandafter#1\csname #4\endcsname{##2}}
  % getter
  \def#3##1{%
    \ifcsdef{#4}{%
      \csname#4\endcsname%
    }{%
      \PackageError{sankey}{#5}{unknown key with \string#3}%
    }%
  }
}
```

14.1.1 Definition of *global* and *expanded* fields (using `\xdef`)

`\snk@setnodeqty`
`\sankeygetnodeqty`

The setter and getter macros to store and retrieve the *quantity* field associated with each Sankey node (the key is the name of the Sankey node).

```
\snk@newfield\xdef\snk@setnodeqty\sankeygetnodeqty%
{@snk@node@qty@#1}{Unknown sankey node '#1'}
```

`\snk@setnodeorient`
`\sankeygetnodeorient`

The setter and getter macros to store and retrieve the *angle* (or orientation) field associated with each Sankey node (the key is the name of the Sankey node).

```
\snk@newfield\xdef\snk@setnodeorient\sankeygetnodeorient%
{@snk@node@orient@#1}{Unknown sankey node '#1'}
```

14.1.2 Definitions of *local* fields (using `\def`)

`\snk@setstartfill`
`\snk@getstartfill`
`\snk@setstartdraw`
`\snk@getstartdraw`

The setter and getter macros to store and retrieve the starting fill/draw paths (the key is the style name).

```
\snk@newfield\def\snk@setstartfill\snk@getstartfill%
{@snk@start@fill@#1}{Unknown sankey start fill path #1}

\snk@newfield\def\snk@setstartdraw\snk@getstartdraw%
{@snk@start@draw@#1}{Unknown sankey start draw path #1}
```

`\snk@setendfill`
`\snk@getendfill`
`\snk@setenddraw`
`\snk@getenddraw`

The setter and getter macros to store and retrieve the ending fill/draw paths (the key is the style name).

```
\snk@newfield\def\snk@setendfill\snk@getendfill%
{@snk@end@fill@#1}{Unknown sankey end fill path #1}

\snk@newfield\def\snk@setenddraw\snk@getenddraw%
{@snk@end@draw@#1}{Unknown sankey end draw path #1}
```

14.1.3 Check if a sankey node is defined

`\snk@ifnodedefined`

The `\snk@ifnodedefined` macro checks if a Sankey node is defined by checking if its name is associated to a *quantity*.

```
\newcommand\snk@ifnodedefined[3]{%
  \ifcsdef{@snk@node@qty@#1}{#2}{#3}%
}
```

`\snk@errorifnotdefined`

The `\snk@errorifnotdefined` macro generares an error message if the Sankye node is not defined.

```
\newcommand\snk@errorifnotdefined[1]{%
  \snk@ifnodedefined{#1}{}%
  {\PackageError{sankey}{Unknown sankey node '#1'}{}}%
}
```

14.2 The `sankey node` shape

A `sankey node` is defined as a TikZ node with a particular *shape*: its width is null and its height matches the associated *quantity*. This shape requires only three anchors: `center`, `left` and `right`. These threee anchors are sufficient to use the `sankey` package. But the `fit` library needs anchors defined by rectangular node.

```
\pgfdeclareshape{sankey node}{
  \inheritsavedanchors[from=rectangle]
  \inheritanchor[from=rectangle]{center}
  \inheritanchorborder[from=rectangle]
  \anchor{left}{\pgf@process{\northeast}}
  \anchor{right}{\pgf@process{\southwest}}
  % compatibility with 'fit' library
  \inheritanchor[from=rectangle]{west}
  \inheritanchor[from=rectangle]{east}
  \inheritanchor[from=rectangle]{north}
  \inheritanchor[from=rectangle]{south}
  \inheritanchor[from=rectangle]{north west}
  \inheritanchor[from=rectangle]{south east}
  \inheritanchor[from=rectangle]{north east}
  \inheritanchor[from=rectangle]{south west}
}
```

14.3 Keys

`\sankeyset`

The `sankey` package uses `pgfkeys` to set options via *key=value* pairs using the `/sankey` path (for Sankey diagram options) and using the `/sankey/node parameters` path (for Sankey node parameters).

The `\sankeyset` macro processes its parameter as a list of comma separated pairs of the form *key=value* with `/sankey` as default path.

```
\pgfkeys{/sankey/.is family}
\NewDocumentCommand\sankeyset{m}{\pgfkeys{sankey,#1}}
```

14.3.1 Keys to define the scale

`\snk@totalqty`
`\snk@totallen`

The `ratio quantity`, `ratio length` and `ratio` keys define the ratio between *flow quantity* and *graphic length* (the scale). The `\snk@totalqty` and `\snk@totallen` macros store the values. All *quantities* are processed by `\fpeval` and all *lengths* are processed by `pgfmath`.

```
\sankeyset{
  ratio quantity/.code={\edef\snk@totalqty{\fpeval{#1}}},
  ratio quantity/.value required,
  ratio length/.code={
    \pgfmathsetmacro\snk@totallen{#1}
    \edef\snk@totallen{\snk@totallen pt}
  },
  ratio length/.value required,
  ratio/.style args={#1/#2}{
    ratio length=#1,
    ratio quantity=#2,
  },
  ratio/.value required,
}
```

14.3.2 Rotate offset

`\snk@rotate`

The `rotate` key stores an offset angle applied to all Sankey nodes. This is useful when using the `rotate` option within a `tikzpicture` or a `scope`. This TikZ option is only applied to coordinates and not to TikZ nodes (remember that Sankey nodes are TikZ nodes). It's up to the author to keep the `rotate` option of the `tikzpicture` and that of the `sankeydiagram` synchronous.

```
\sankeyset{
  rotate/.code={\edef\snk@rotate{\fpeval{#1}}},
  rotate/.value required,
}
```

14.3.3 Minimum radius

`\snk@minradius`

The `minimum radius` key processes the minimum radius of curvature by `pgfmath` then stores it in the `\snk@minradius` macro.

```
\sankeyset{
  minimum radius/.code={
    \pgfmathsetmacro\snk@minradius{#1}
    \edef\snk@minradius{\snk@minradius pt}
  },
  minimum radius/.value required,
}
```

14.3.4 Outin step

`\snk@stepoutin`

The `outin` key stores its value in the `\snk@stepoutin` macro.

```
\sankeyset{
  outin steps/.estore in=\snk@stepoutin,
  outon steps/.value required,
}
```

14.3.5 Sankey debug

The `debug` key drives the `sankey debug` toggle.

```
\newtoggle{sankey debug}
\sankeyset{
  debug/.is choice,
  debug/true/.code={\toggletrue{sankey debug}},
  debug/false/.code={\togglefalse{sankey debug}},
  debug/.default=true,
}
```

14.3.6 Start and end styles

`\snk@startstyle`
`\snk@endstyle`

The `start style` and `end style` keys are choices.

The `new start style` and `new end style` keys add new option to these choices. They use the `startfill`, `startdraw` fields or the `endfill` and `enddraw` fields (using the style *name* as key) to store the fill and draw paths then create a new option to install the new start (`\snk@startstyle`) or end (`\snk@endstyle`) style.

```
\sankeyset{
  start style/.is choice,
  end style/.is choice,
  % to define new start and end styles
  new start style/.code n args={3}{% name, fill path, draw path
    \snk@setstartfill{#1}{#2}
    \snk@setstartdraw{#1}{#3}
    \sankeyset{start style/#1/.code={\def\snk@startstyle{#1}}}
  },
  new end style/.code n args={3}{% name, fill path, draw path
    \snk@setendfill{#1}{#2}
    \snk@setenddraw{#1}{#3}
    \sankeyset{end style/#1/.code={\def\snk@endstyle{#1}}}
  },
}
```

14.3.7 Initial parameters

The `@initial options` style defines default values for options of Sankey diagram. The `every diagram` style (initially empty) allows the user to choose its own default values.

The `@initial options` and the `every diagram` styles are applied (in this order) at the beginning of each Sankey diagram.

```
\sankeyset{
  debug color/.style={/utils/exec={\colorlet{debug color}{#1}}},
  @initial options/.style={
    ratio=1cm/10,
    minimum radius=5mm,%
    outin steps=10,
    debug=false,
    start style=none,
    end style=none,
    rotate=0,
    % default fill/draw styles,
    fill/.style={line width=0pt,fill=white},
    draw/.style={draw=black,line width=.4pt},
    % debug color used by all debug macros
    debug color=red!75!black,
    % debug line between left and right anchors
    debug line/.style={overlay,draw=debug color,|-|},
    % debug line between center and label
    debug normal/.style={overlay,draw=debug color},
    % debug node label
    debug label/.style={
      overlay,
      draw,
      font=\ttfamily\tiny,
      text=debug color,text opacity=1,
      inner sep=.1em,
      fill=white,fill opacity=1,
      rounded corners=.1em,
      node contents={\name},
    },
    every node/.style={},
  },
  every diagram/.style={},
}

```

14.3.8 Sankey node parameters

The `/sankey/node parameters` family defines all parameters during creation of Sankey node.

```
\sankeyset{node parameters/.is family}
```

`\name`
`\qty`
`\orient`
`\pos`

The `name`, `quantity`, `angle` and `at` keys use the `\name`, `\qty`, `\orient` and `\pos` macros to store the *name*, the *quantity*, the *orientation* (or *angle*) and the *position* of a Sankey node during its creation.

The *quantity* is processed via `\fpeval`. The *orientation* is normalized.

```
\sankeyset{node parameters,
  name/.estore in=\name,
  name/.value required,
  quantity/.code={\edef\qty{\fpeval{#1}}},
  quantity/.value required,
  angle/.code={\edef\orient{\snk@normalize@angle{#1}}},
  angle/.value required,
  at/.code={\snk@getpos\pos{#1}},
  at/.value required,
}
```

The `as` key is just a shortcut to define the four current node parameters by copying them from an existing Sankey node.

```
\sankeyset{node parameters,
  as/.style={
    name=#1,
    quantity=\sankeygetnodeqty{#1},
    angle=\sankeygetnodeorient{#1},
    at={#1.center},
  },
  as/.value required,
}
```

`\snk@anchor`

The `anchor` key stores in the `\snk@anchor` macro the anchor name to use to create the new current Sankey node

```
\sankeyset{node parameters,
  anchor/.is choice,
  anchor/left/.code={\def\snk@anchor{left}},
  anchor/right/.code={\def\snk@anchor{right}},
  anchor/center/.code={\def\snk@anchor{center}},
  anchor/.value required,
}
```

The `start` and `end` keys drive the `sankey node start` and `sankey node end` toggles.

```
\newtoggle{sankey node start}
\newtoggle{sankey node end}
\sankeyset{node parameters,
  start/.is choice,
  start/true/.code={\toggletrue{sankey node start}},
  start/false/.code={\togglefalse{sankey node start}},
  start/.default=true,
  %
  end/.is choice,
  end/true/.code={\toggletrue{sankey node end}},
  end/false/.code={\togglefalse{sankey node end}},
  end/.default=true,
}
```

`\snk@listofforks`
`\snk@forkanchor`

The `forked` and `fork anchor` keys store their value in the `\snk@listofforks` and `\snk@forkanchor` macros.

```
\sankeyset{node parameters,
  forked/.estore in=\snk@listofforks,
  forked/.value required,
  fork anchor/.estore in=\snk@forkanchor,
  fork anchor/.value required,
}
```

The `@initial parameters` style initialises all Sankey node parameters at the start of the creation of a new Sankey node.

```
\sankeyset{node parameters,
  @initial parameters/.style={
    start=false,
    end=false,
    forked=,
    fork anchor=,
    anchor=center,
    at={0,0},
    angle=0,
  },
}
```

14.3.9 Internal Tikz style

To apply this style with Tikz, use absolute key name (`/sankey/@sankey node`). This style is used to create the Tikz node assoicated to a Sankey node.

```
\sankeyset{
  % sankey node TikZ style
  @sankey node/.style n args={3}{% name, pos, anchor
    shape=sankey node,
    inner sep=0,
    minimum height={\sankeyqtytolen{\sankeygetnodeqty{#1}}},
    minimum width=0,
    draw=none,
    line width=0pt,
    fill=none,
    node contents={},
    rotate=\sankeygetnodeorient{#1}+\snk@rotate,
    at={(#2)},
    name=#1,
    anchor=#3,
  },
}
```

14.4 Internal macros

`\snk@getpos` The `\snk@getpos` macro extracts the position of a TikZ node[6].

```
\def\snk@getpos#1#2{%
  \tikz@scan@one@point\pgfutil@firstofone(#2)\relax%
  \edef#1{\the\pgf@x,\the\pgf@y}%
}
```

`\snk@modulo` The `\snk@modulo` macro evaluates #1 modulo #2 using `\fpeval`.

```
\def\snk@modulo#1#2{\fpeval{#1-(floor((#1)/(#2),0)*#2)}}
```

`\snk@normalize@angle` The `\snk@normalize@angle` macro normalizes #1 (an angle) between -180 and 180 (using `\fpeval`).

```
\def\snk@normalize@angle#1{%
  \fpeval{\snk@modulo{(#1)+180}{360}-180}%
}
```

`\snk@show@debug` The `snk@show@debug` macro draws debug information of the Sankey node named #1 but only if the `sankey debug` toogle is true. Everything is drawn on the `sankeydebug` layer. It uses the `/sankey/debug line`, `/sankey/debug normal` and `/sankey/debug label` Tikz styles.

```
\def\snk@show@debug#1{% node name
  \iftoggle{sankey debug}{
    \begingroup
    \edef\name{#1}
    \edef\qty{\sankeygetnodeqty{\name}}
    \edef\orient{\sankeygetnodeorient{\name}}
    \begin{pgfonlayer}{sankeydebug}
      \path[/sankey/debug line] (\name.left) -- (\name.right);
      \pgfmathsetmacro{\snk@len}{\sankeyqtytolen{\qty}/3}
      \path[/sankey/debug normal] (\name.center)
      -- ($(\name.center)!\snk@len pt!90:(\name.right)$)
      node[/sankey/debug label,rotate=\orient+90+\snk@rotate,anchor=north];
    \end{pgfonlayer}
    \endgroup
  }{}
}
```

---

[6] Thanks to Andrew Stacey https://tex.stackexchange.com/a/33765/14500

`\snk@makeforkednode`
`\snk@tot`
`\snk@subnodeqty`
`\snk@subnodename`
`\snk@added@values`

The `\snk@makeforkednode` forks a Sankey node.

```
\def\snk@makeforkednode{
  \begingroup
  \ifdefempty{\snk@listofforks}{
    \ifdefempty{\snk@forkanchor}{}{
      \PackageWarning{sankey}%
      {Can't use 'fork anchor' key without 'forked' key}
    }
  }{
    \def\snk@tot{0}
    \def\snk@added@values{}
    \sankeyset{
      @add forked node/.code args={##1/##2}{
        \coordinate (##2) at ($(\name.left)%
        !\fpeval{(\snk@tot+.5*(##1))/\qty}%
        !(\name.right)$);
        \edef\snk@orient{\orient}
        \sankeynode[debug=false]
        {name=##2,quantity=##1,at=##2,angle=\snk@orient}
        \edef\snk@tot{\fpeval{\snk@tot+##1}}
        \edef\snk@added@values{\snk@added@values+##1}
      },
      @add forked node/.list/.expand once=\snk@listofforks,
    }
    \edef\snk@diff{\fpeval{abs(\qty-\snk@tot)}}
    \ifnumequal{\snk@diff}{0}{}{
      \PackageWarning{sankey}%
      {^^J*** Warning: bad sankey fork: %
        \qty\space!=\space\snk@added@values(=\snk@tot)%
        ^^J\snk@listofforks}
    }
    \ifdefempty{\snk@forkanchor}{
      \edef\snk@forkanchor{\name.\snk@anchor}
    }{}
    \snk@getpos\snk@c{$(\snk@forkanchor) - (\pos)$}
    \sankeynode{as=\name,at={$(\name) - (\snk@c)$}}
    \foreach \snk@subnodeqty/\snk@subnodename in \snk@listofforks {
      \sankeynode{as=\snk@subnodename,at={$(\snk@subnodename) - (\snk@c)$}}
    }
  }
  \endgroup
}
```

`\snk@makenode`

The `\snk@makenode` macro creates a new Sankey node named `\name` with `\qty` quantity, oriented at `\orient` degrees (but modified by the `\snk@rotate` angle offset), anchored by its `\anchor` (or its *center* by default) at `\pos` position.

```
\def\snk@makenode{
  \begingroup
  \snk@setnodeqty{\name}{\qty}
  \edef\orient{\snk@normalize@angle{\orient}}
  \snk@setnodeorient{\name}{\orient}
  \ifundef{\snk@anchor}{\def\snk@anchor{center}}{}
  \node[/sankey/@sankey node={\name}{\pos}{\snk@anchor}];
  \endgroup
}
```

`\snk@filldrawstart`

The `\snk@filldrawstart` macro fills (with the `/sankey/fill` TikZ style) then draws (with the `/sankey/draw` TikZ style) a start of flow using paths from style `\snk@startstyle` on the Sankey node named `\name`.

```
\def\snk@filldrawstart{
  \begin{scope}[shift={(\name)},rotate=\orient]
    \path[/sankey/fill] \snk@getstartfill{\snk@startstyle};
    \path[/sankey/draw] \snk@getstartdraw{\snk@startstyle};
  \end{scope}
}
```

`\snk@filldrawend`

The `\snk@filldrawend` macro fills (with the `/sankey/fill` TikZ style) then draws (with the `/sankey/draw` TikZ style) a end of flow using paths from style `\snk@endstyle` on the Sankey node named `\name`.

```
\def\snk@filldrawend{
  \begin{scope}[shift={(\name)},rotate=\sankeygetnodeorient{\name}]
    \path[/sankey/fill] \snk@getendfill{\snk@endstyle};
    \path[/sankey/draw] \snk@getenddraw{\snk@endstyle};
  \end{scope}
}
```

`\snk@checkquantities`
`\snk@qtyi`
`\snk@qtyii`

The `\snk@checkquantities` compares quantities from Sankey nodes `#1` and `#2` and emits an error message if they differ (`#3` is the name of the macro which requested the verification).

```
\def\snk@checkquantities#1#2#3{
  \begingroup
  \edef\snk@qtyi{\sankeygetnodeqty{#1}}
  \edef\snk@qtyii{\sankeygetnodeqty{#2}}
  \ifdefstrequal{\snk@qtyi}{\snk@qtyii}{}{
    \PackageError{sankey}%
    {^^J*** \string#3: quantities differ between %
      #1 (\snk@qtyi) and #2 (\snk@qtyii)%
      ^^J}%
    {The quantities of the two Sankey nodes must be equal.}
  }
  \endgroup
}
```

14.5 User macros

The user macros are globally defined as internal (with `snk@` prefix) and locally defined in the `sankeydiagram` environment as user macros (without the `snk@` prefix).

`\snk@sankeydubins`

The `\snk@sankeydubins` macro links two Sankey nodes via a Dubins path. First, it computes the Dubins path between centers (left and right radii are the same) and stores the result in `sankey`. Then it uses the stored result to fill and draw the lane (left border and right borders use Dubins paths with asymmetric radii).

```
\NewDocumentCommand\snk@sankeydubins{O{}mm}{% options, sn, en
  \snk@errorifnotdefined{#2}
  \snk@errorifnotdefined{#3}
  \snk@checkquantities{#2}{#3}{\sankeydubins}
  \begingroup
  \sankeyset{#1}
  \pgfmathsetmacro\qty{\sankeygetnodeqty{#2}}
  \dubinspathset{
    sankey/.style={
      start point=#2.center,start angle=\sankeygetnodeorient{#2},
      end point=#3.center,end angle=\sankeygetnodeorient{#3},
      minimum radius=\snk@minradius + .5 * \sankeyqtytolen{\qty} pt,
    },
  }
  \dubinspathcalc{sankey,store=sankey}
  \dubinspathset{
    left border/.style={
      sankey, use store=sankey,
      left and right minimum radii={\snk@minradius}
      and {\snk@minradius + \sankeyqtytolen{\qty} pt},
    },
    right border/.style={
      sankey, use store=sankey,
      left and right minimum radii=
      {\snk@minradius + \sankeyqtytolen{\qty} pt}
      and {\snk@minradius},
    },
  }
  % fill the region
  \path[/sankey/fill] (#2.left)  \dubinspath{left border}
  -- (#3.left) -- (#3.right) \dubinspath{right border,reverse}
  -- (#2.right) -- cycle;
  % draw left and right borders
  \path[/sankey/draw] (#2.left) \dubinspath{left border}
  (#2.right) \dubinspath{right border};
  \endgroup
}
```

`\snk@sankeyoutin`

The `\snk@sankeyoutin` macro links two Sankey nodes via a Bézier curve. First, to simulate constant width, it creates `\snk@stepoutin` intermediate Sankey nodes along the Bézier curve. Then, the lane is filled and drawn linking all the intermediate Sankey node via smaller Bézier curves.

```
\NewDocumentCommand\snk@sankeyoutin{O{}mm}{% options, sn, en
  \snk@errorifnotdefined{#2}
  \snk@errorifnotdefined{#3}
  \snk@checkquantities{#2}{#3}{\sankeyoutin}
  \begingroup
  \sankeyset{#1}
  \edef\qty{\sankeygetnodeqty{#2}}
  \pgfmathsetmacro\snk@len{\sankeyqtytolen{\qty}/2}
  \edef\snk@step{\fpeval{1/\snk@stepoutin}}
  \edef\snk@laststep{\inteval{\snk@stepoutin-1}}
  \path[overlay,decorate,decoration={
      markings,
      mark=between positions \snk@step and {\fpeval{1-.5*\snk@step}}
      step \snk@step with {
        \edef\snk@outinmidptname{%
          snk@outinmidpt-%
          \pgfkeysvalueof{/pgf/decoration/mark info/sequence number}%
        }
        \path
        (0,0) coordinate(\snk@outinmidptname)
        (0,-\snk@len pt) coordinate (\snk@outinmidptname-r)
        (0,\snk@len pt) coordinate (\snk@outinmidptname-l)
        ;
      }
    }]
  (#2.center)
  to[out=\sankeygetnodeorient{#2},in=\sankeygetnodeorient{#3}+180]
  (#3.center);
  \foreach \snk@ptnum in {1,...,\snk@laststep}{
    \edef\snk@outinmidptname{snk@outinmidpt-\snk@ptnum}
    \dbp@anglebetween\snk@outinmidptangle%
    {\snk@outinmidptname-r}{\snk@outinmidptname-l}
    \sankeynode[debug=false]{%
      name=\snk@outinmidptname,
      quantity=\qty,
      angle=\snk@outinmidptangle-90,%
      at=\snk@outinmidptname%
    }
  }
  \sankeynode[debug=false]{%
    name={snk@outinmidpt-0},
    quantity=\sankeygetnodeqty{#2},
    angle=\sankeygetnodeorient{#2},%
    at={#2}%
  }
  \sankeynode[debug=false]{%
    name={snk@outinmidpt-\snk@stepoutin},
    quantity={\sankeygetnodeqty{#3}},
    angle={\sankeygetnodeorient{#3}},%
    at={#3}%
  }
  \path[/sankey/fill,looseness=1]
  (snk@outinmidpt-0.left)
  \foreach \snk@curpt
  [remember=\snk@curpt as \snk@prevpt (initially 0)]
  in {1,...,\snk@stepoutin}{
    to[out=\sankeygetnodeorient{snk@outinmidpt-\snk@prevpt},
      in=\sankeygetnodeorient{snk@outinmidpt-\snk@curpt}+180]
    (snk@outinmidpt-\snk@curpt.left)
  }
  --
  (snk@outinmidpt-\snk@stepoutin.right)
  \foreach \snk@curpt
  [remember=\snk@curpt as \snk@prevpt (initially \snk@stepoutin)]
  in {\snk@laststep,...,0}{
    to[out=\sankeygetnodeorient{snk@outinmidpt-\snk@prevpt}+180,
      in=\sankeygetnodeorient{snk@outinmidpt-\snk@curpt}]
    (snk@outinmidpt-\snk@curpt.right)
  }
  -- cycle;
  \path[/sankey/draw,looseness=1]
  (snk@outinmidpt-0.left)
  \foreach \snk@curpt
  [remember=\snk@curpt as \snk@prevpt (initially 0)]
  in {1,...,\snk@stepoutin}{
    to[out=\sankeygetnodeorient{snk@outinmidpt-\snk@prevpt},
      in=\sankeygetnodeorient{snk@outinmidpt-\snk@curpt}+180]
    (snk@outinmidpt-\snk@curpt.left)
  }
  (snk@outinmidpt-\snk@stepoutin.right)
  \foreach \snk@curpt
  [remember=\snk@curpt as \snk@prevpt (initially \snk@stepoutin)]
  in {\snk@laststep,...,0}{
    to[out=\sankeygetnodeorient{snk@outinmidpt-\snk@prevpt}+180,
      in=\sankeygetnodeorient{snk@outinmidpt-\snk@curpt}]
    (snk@outinmidpt-\snk@curpt.right)
  };
  \endgroup
}
```

`\snk@sankeynodealias` The `\snk@sankeynodealias` macro clones the Sankey node named `#1` into a Sankey node named `#2`.

```
\NewDocumentCommand\snk@sankeynodealias{mm}{%name, alias
  \snk@errorifnotdefined{#1}
  \path[late options={name=#1,alias=#2}];
  \snk@setnodeqty{#2}{\sankeygetnodeqty{#1}}
  \snk@setnodeorient{#2}{\sankeygetnodeorient{#1}}
}
```

`\snk@sankeynode` The `\snk@sankeynode` macro creates the new Sankey node named `#2`.

```
\NewDocumentCommand\snk@sankeynode{O{}m}{% options, node parameters
  \begingroup
  \sankeyset{#1}
  \sankeyset{node parameters,@initial parameters,/sankey/every node,#2}
  \snk@makenode{}
  \snk@makeforkednode{}
  \iftoggle{sankey node start}{\snk@filldrawstart}{}
  \iftoggle{sankey node end}{\snk@filldrawend}{}
  \snk@show@debug{\name}
  \endgroup
}
```

`\snk@sankeystart` The `\snk@sankeystart` macro fills and draws a starting lane attached to the Sankey node named `#2`.

```
\NewDocumentCommand\snk@sankeystart{O{}m}{% options, name
  \snk@errorifnotdefined{#2}
  \begingroup
  \sankeyset{#1}
  \edef\name{#2}
  \edef\orient{\sankeygetnodeorient{#2}}
  \edef\qty{\sankeygetnodeqty{#2}}
  \snk@filldrawstart
  \endgroup
}
```

`\snk@sankeynodestart` The `\snk@sankeynodestart` macro creates the new Sankey node named `#2` then fills and draws a starting lane attached to this new Sankey node.

```
\NewDocumentCommand\snk@sankeynodestart{O{}m}{% option, node parameters
  \sankeynode[#1]{start,#2}
}
```

`\snk@sankeyend` The `\snk@sankeyend` macro fills and draws an ending lane attached to the Sankey node named `#2`.

```
\NewDocumentCommand\snk@sankeyend{O{}m}{%options, name
  \snk@errorifnotdefined{#2}
  \begingroup
  \sankeyset{#1}
  \edef\name{#2}
  \edef\orient{\sankeygetnodeorient{#2}}
  \edef\qty{\sankeygetnodeqty{#2}}
  \snk@filldrawend
  \endgroup
}
```

`\snk@sankeynodeend` The `\snk@sankeynodeend` macro creates the new Sankey node named `#2` then fills and draws an ending lane attached to this new Sankey node.

```
\NewDocumentCommand\snk@sankeynodeend{O{}m}{% options, node parameters
  \sankeynode[#1]{end,#2}
}
```

`\snk@init@move` The `\snk@init@move` macro applies options et clones the current node before its moving.

```
\def\snk@init@move#1#2{% params: options, name
  \sankeyset{#1}
  \edef\name{#2}
  \edef\snk@oldname{#2-old}
  \sankeynodealias{\name}{\snk@oldname}
  \edef\qty{\sankeygetnodeqty{\name}}
}
```

`\snk@sankeyadvance` The `\snk@sankeyadvance` macro moves toward (or backward if *starred* calls – `#1`) the Sankey node named `#3`. `#4` is a distance. The previous position is keeped by a Sankey node named `#3-old`.

```
\NewDocumentCommand\snk@sankeyadvance{sO{}mm}{%
  % params: *(reverse), options, name, distance
  \snk@errorifnotdefined{#3}
  \begingroup
  \snk@init@move{#2}{#3}
  \IfBooleanTF{#1}{
    % move backward
    \sankeynode{
      at={$(\snk@oldname.center)!#4!90:(\snk@oldname.left)$},
      angle=\sankeygetnodeorient{\snk@oldname},
      quantity=\sankeygetnodeqty{\snk@oldname},
      name=\name,
    }
    \path[/sankey/fill]
    (\name.left) -- (\snk@oldname.left)
    -- (\snk@oldname.right) -- (\name.right) -- cycle;
    \path[/sankey/draw]
    (\name.left) -- (\snk@oldname.left)
    (\snk@oldname.right) -- (\name.right);
  }{
    % move forward
    \sankeynode{
      at={$(\snk@oldname.center)!#4!-90:(\snk@oldname.left)$},
      angle=\sankeygetnodeorient{\snk@oldname},
      quantity=\sankeygetnodeqty{\snk@oldname},
      name=\name,
    }
    \path[/sankey/fill]
    (\snk@oldname.left) -- (\name.left)
    -- (\name.right) -- (\snk@oldname.right) -- cycle;
    \path[/sankey/draw]
    (\snk@oldname.left) -- (\name.left)
    (\name.right) -- (\snk@oldname.right);
  }
  \snk@show@debug{\name}
  \endgroup
}
```

`\snk@sankeyturnright` The `sankeyturnright` macro moves forward the Sankey node named `#3` by turning right. The angle is `#4` (the starred version uses the opposite of `#4`). If the angle is *negative*, the macro calls the `\sankeyturnrightbackward` macro to move backward else the macro fills/draws the lane between the previous position and the new position. The previous position is keeped by a Sankey node named `#3-old`.

```
\NewDocumentCommand\snk@sankeyturnright{sO{}mm}{
  % *(reverse), options, name, angle
  \snk@errorifnotdefined{#3}
  \begingroup
  \IfBooleanTF{#1}
  {\edef\snk@angle{\fpeval{-1*#4}}}
  {\edef\snk@angle{\fpeval{1*#4}}}
  \edef\snk@anglesign{\fpeval{sign(\snk@angle)}}
  \ifnumgreater{\snk@anglesign}{-1}{
    \snk@init@move{#2}{#3}
    \edef\orient{\snk@normalize@angle{\sankeygetnodeorient{\name}-\snk@angle}}
    \snk@getpos\snk@c{$(\name.right)!-\snk@minradius!(\name.left)$}
    \snk@getpos\pos{$(\snk@c)!1!-\snk@angle:(\name.center)$}
    \snk@makenode{}
    % fill the region
    \path[/sankey/fill] let
    \p1=(\snk@oldname.left),\p2=(\snk@oldname.right),
    \p3=(\name.left),\p4=(\name.right),
    \n1={\sankeyqtytolen{\qty}},
    \n{maxr}={\snk@minradius+\n1},
    \n{minr}={\snk@minradius}
    in
    (\p1) arc(\orient+\snk@angle+90:\orient+90:\n{maxr}) -- (\p3) --
    (\p4) arc(\orient+90:\orient+\snk@angle+90:\n{minr}) -- (\p2) -- cycle;
    % draw left and right borders
    \path[/sankey/draw] let
    \p1=(\snk@oldname.left),\p2=(\snk@oldname.right),
    \p3=(\name.left),\p4=(\name.right),
    \n1={\sankeyqtytolen{\qty}},
    \n{maxr}={\snk@minradius+\n1},
    \n{minr}={\snk@minradius}
    in
    (\p1) arc(\orient+\snk@angle+90:\orient+90:\n{maxr})
    (\p4) arc(\orient+90:\orient+\snk@angle+90:\n{minr});
    \snk@show@debug{\name}
  }{
    \sankeyturnrightbackward[#2]{#3}{-1*\snk@angle}
  }
  \endgroup
}
```

`\snk@sankeyturnrightbackward` The `\snk@sankeyturnrightbackward` macro moves backward the Sankey node named `#3` by turning right. The angle is `#4` (the starred version uses the opposite of `#4`). If the angle is *negative*, the macro calls the `\sankeyturnright` macro to move forward else the macro fills/draws the lane between the previous position and the new position. The previous position is keeped by a Sankey node named `#3-old`.

```
\NewDocumentCommand\snk@sankeyturnrightbackward{sO{}mm}{
  % *(reverse), options, name, angle
  \snk@errorifnotdefined{#3}
  \begingroup
  \IfBooleanTF{#1}
  {\edef\snk@angle{\fpeval{-1*#4}}}
  {\edef\snk@angle{\fpeval{1*#4}}}
  \edef\snk@anglesign{\fpeval{sign(\snk@angle)}}
  \ifnumgreater{\snk@anglesign}{-1}{
    \snk@init@move{#2}{#3}
    \edef\orient{\snk@normalize@angle{\sankeygetnodeorient{\name}+\snk@angle}}
    \snk@getpos\snk@c{$(\name.right)!-\snk@minradius!(\name.left)$}
    \snk@getpos\pos{$(\snk@c)!1!\snk@angle:(\name.center)$}
    \snk@makenode{}
    % fill the region
    \path[/sankey/fill] let
    \p1=(\name.left),\p2=(\name.right),
    \p3=(\snk@oldname.left),\p4=(\snk@oldname.right),
    \n1={\sankeyqtytolen{\qty}},
    \n{maxr}={\snk@minradius+\n1}, \n{minr}={\snk@minradius}
    in
    (\p1) arc(\orient+90:\orient-\snk@angle+90:\n{maxr}) -- (\p3) --
    (\p4) arc(\orient-\snk@angle+90:\orient+90:\n{minr}) -- (\p2) -- cycle;
    % draw left and right borders
    \path[/sankey/draw] let
    \p1=(\name.left),\p2=(\name.right),
    \p3=(\snk@oldname.left),\p4=(\snk@oldname.right),
    \n1={\sankeyqtytolen{\qty}},
    \n{maxr}={\snk@minradius+\n1}, \n{minr}={\snk@minradius}
    in
    (\p1) arc(\orient+90:\orient-\snk@angle+90:\n{maxr})
    (\p4) arc(\orient-\snk@angle+90:\orient+90:\n{minr});
    \snk@show@debug{\name}
  }{
    \sankeyturnright[#2]{#3}{-1*\snk@angle}
  }
  \endgroup
}
```

`\snk@sankeyturnleft` The `\snk@sankeyturnleft` macro moves forward the Sankey node named `#3` by turning left. The angle is `#4` (the starred version uses the opposite of `#4`). If the angle is *negative*, the macro calls the `\sankeyturnleftbackward` macro to move backward else the macro fills/draws the lane between the previous position and the new position. The previous position is keeped by a Sankey node named `#3-old`.

```
\NewDocumentCommand\snk@sankeyturnleft{sO{}mm}{
  % *(reverse), options, name, angle
  \snk@errorifnotdefined{#3}
  \begingroup
  \IfBooleanTF{#1}
  {\edef\snk@angle{\fpeval{-1*#4}}}
  {\edef\snk@angle{\fpeval{1*#4}}}
  \edef\snk@anglesign{\fpeval{sign(\snk@angle)}}
  \ifnumgreater{\snk@anglesign}{-1}{
    \snk@init@move{#2}{#3}
    \edef\orient{\snk@normalize@angle{\sankeygetnodeorient{\name}+\snk@angle}}
    \snk@getpos\snk@c{$(\name.left)!-\snk@minradius!(\name.right)$}
    \snk@getpos\pos{$(\snk@c)!1!\snk@angle:(\name.center)$}
    \snk@makenode{}
    % fill the region
    \path[/sankey/fill] let
    \p1=(\snk@oldname.left),\p2=(\snk@oldname.right),
    \p3=(\name.left),\p4=(\name.right),
    \n1={\sankeyqtytolen{\qty}},
    \n{maxr}={\snk@minradius+\n1}, \n{minr}={\snk@minradius}
    in
    (\p1) arc(\orient-\snk@angle-90:\orient-90:\n{minr}) -- (\p3) --
    (\p4) arc(\orient-90:\orient-\snk@angle-90:\n{maxr}) -- (\p2) -- cycle;
    % draw left and right borders
    \path[/sankey/draw] let
    \p1=(\snk@oldname.left),\p2=(\snk@oldname.right),
    \p3=(\name.left),\p4=(\name.right),
    \n1={\sankeyqtytolen{\qty}},
    \n{maxr}={\snk@minradius+\n1}, \n{minr}={\snk@minradius}
    in
    (\p1) arc(\orient-\snk@angle-90:\orient-90:\n{minr})
    (\p4) arc(\orient-90:\orient-\snk@angle-90:\n{maxr});
    \snk@show@debug{\name}
  }{
    \sankeyturnleftbackward[#2]{#3}{-1*\snk@angle}
  }
  \endgroup
}
```

`\snk@sankeyturnleftbackward` The `\snk@sankeyturnleftbackward` macro moves backward the Sankey node named `#3` by turning left. The angle is `#4` (the starred version uses the opposite of `#4`). If the angle is *negative*, the macro calls the `\sankeyturnleft` macro to move forward else the macro fills/draws the lane between the previous position and the new position. The previous position is keeped by a Sankey node named `#3-old`.

```
\NewDocumentCommand\snk@sankeyturnleftbackward{sO{}mm}{
  % *(reverse), options, name, angle
  \snk@errorifnotdefined{#3}
  \begingroup
  \IfBooleanTF{#1}
  {\edef\snk@angle{\fpeval{-1*#4}}}
  {\edef\snk@angle{\fpeval{1*#4}}}
  \edef\snk@anglesign{\fpeval{sign(\snk@angle)}}
  \ifnumgreater{\snk@anglesign}{-1}{
    \snk@init@move{#2}{#3}
    \edef\orient{\snk@normalize@angle{\sankeygetnodeorient{\name}-\snk@angle}}
    \snk@getpos\snk@c{$(\name.left)!-\snk@minradius!(\name.right)$}
    \snk@getpos\pos{$(\snk@c)!1!-\snk@angle:(\name.center)$}
    \snk@makenode{}
    % fill the region
    \path[/sankey/fill] let
    \p1=(\name.left),\p2=(\name.right),
    \p3=(\snk@oldname.left),\p4=(\snk@oldname.right),
    \n1={\sankeyqtytolen{\qty}},
    \n{maxr}={\snk@minradius+\n1}, \n{minr}={\snk@minradius}
    in
    (\p1) arc(\orient-90:\orient+\snk@angle-90:\n{minr}) -- (\p3) --
    (\p4) arc(\orient+\snk@angle-90:\orient-90:\n{maxr}) -- (\p2) -- cycle;
    % draw left and right borders
    \path[/sankey/draw] let
    \p1=(\name.left),\p2=(\name.right),
    \p3=(\snk@oldname.left),\p4=(\snk@oldname.right),
    \n1={\sankeyqtytolen{\qty}},
    \n{maxr}={\snk@minradius+\n1}, \n{minr}={\snk@minradius}
    in
    (\p1) arc(\orient-90:\orient+\snk@angle-90:\n{minr})
    (\p4) arc(\orient+\snk@angle-90:\orient-90:\n{maxr});
    \snk@show@debug{\name}
  }{
    \sankeyturnleft[#2]{#3}{-1*\snk@angle}
  }
  \endgroup
}
```

`\snk@sankeyturn` The `\snk@sankeyturn` macro moves toward (or backward if *starred* calls – `#1`) the Sankey node named `#3` by turning left (angle `#4` is positive) or right (angle `#4` is negative). The previous position is keeped by a Sankey node named `#3-old`.

```
\NewDocumentCommand\snk@sankeyturn{sO{}mm}{%
  % *(reverse), options, name, angle
  \snk@errorifnotdefined{#3}
  \begingroup
  \edef\snk@anglesign{\fpeval{sign(#4)}}
  \IfBooleanTF{#1}{
    \ifnumgreater{\snk@anglesign}{-1}
    {\sankeyturnleftbackward[#2]{#3}{#4}}
    {\sankeyturnrightbackward[#2]{#3}{-1*#4}}
  }{
    \ifnumgreater{\snk@anglesign}{-1}
    {\sankeyturnleft[#2]{#3}{#4}}
    {\sankeyturnright[#2]{#3}{-1*#4}}
  }
  \endgroup
}
```

`\snk@sankeyfork` The `\snk@sankeyfork` macro forks the Sankey node named `#2` to the list of subnodes given by `#3`. The subnodes are cloned to take into account the `debug` option.

```
\NewDocumentCommand\snk@sankeyfork{O{}mm}{%options, name, list of forks
  \snk@errorifnotdefined{#2}
  \begingroup
  \sankeyset{#1}
  \sankeynode[debug=false]{as={#2},forked={#3}}
  \foreach \qty/\snk@subnodename in {#3}{\sankeynode{as={\snk@subnodename}}}
  \endgroup
}
```

`\snk@sankeyqtytolen` The `\snk@sankeyqtytolen` macro converts quantity to length using `\fpeval` and the ratio determined by `\snk@totalqty` and `\snk@totallen`.

```
\NewExpandableDocumentCommand\snk@sankeyqtytolen{m}{%
  \fpeval{(#1)/\snk@totalqty*\snk@totallen}%
}
```

14.6 The `sankeydiagram` environment

sankeydiagram (*env.*)

The `sankeydiagram` environment allows the creation of Sankey diagrams.

\sankeyadvance
\sankeydubins
\sankeyend
\sankeyfork
\sankeynodealias
\sankeynodeend
\sankeynodestart
\sankeynode
\sankeyoutin
\sankeyqtytolen
\sankeystart
\sankeyturnleftbackward
\sankeyturnleft
\sankeyturnrightbackward
\sankeyturnright
\sankeyturn

It defines locally all the macros used by a Sankey diagram.

Then it applies the `@initial options` and `every diagram` styles (in this order) and applies all the keys provided in its optional argument.

```
\NewDocumentEnvironment{sankeydiagram}{O{}}{
  \NewCommandCopy{\sankeyadvance}{\snk@sankeyadvance}
  \NewCommandCopy{\sankeydubins}{\snk@sankeydubins}
  \NewCommandCopy{\sankeyend}{\snk@sankeyend}
  \NewCommandCopy{\sankeyfork}{\snk@sankeyfork}
  \NewCommandCopy{\sankeynodealias}{\snk@sankeynodealias}
  \NewCommandCopy{\sankeynodeend}{\snk@sankeynodeend}
  \NewCommandCopy{\sankeynodestart}{\snk@sankeynodestart}
  \NewCommandCopy{\sankeynode}{\snk@sankeynode}
  \NewCommandCopy{\sankeyoutin}{\snk@sankeyoutin}
  \NewCommandCopy{\sankeyqtytolen}{\snk@sankeyqtytolen}
  \NewCommandCopy{\sankeystart}{\snk@sankeystart}
  \NewCommandCopy{\sankeyturnleftbackward}{\snk@sankeyturnleftbackward}
  \NewCommandCopy{\sankeyturnleft}{\snk@sankeyturnleft}
  \NewCommandCopy{\sankeyturnrightbackward}{\snk@sankeyturnrightbackward}
  \NewCommandCopy{\sankeyturnright}{\snk@sankeyturnright}
  \NewCommandCopy{\sankeyturn}{\snk@sankeyturn}
  \sankeyset{
    @initial options,
    every diagram,
    % user values
    #1}
}
{} % empty but mandatory ! :-)
```

14.7 Predefined start and end styles

The `none` style.

```
\sankeyset{
  new start style={none}{}{},
  new end style={none}{}{},
}
```

The `simple` style.

```
\sankeyset{
  new start style={simple}{
    (\name.left) -- ([xshift=-.5\pgflinewidth]\name.left)
    -- ([xshift=-.5\pgflinewidth]\name.right) -- (\name.right) -- cycle
  }{
    (\name.left) -- ([xshift=-.5\pgflinewidth]\name.left)
    -- ([xshift=-.5\pgflinewidth]\name.right) -- (\name.right)
  },
  new end style={simple}{
    (\name.left) -- ([xshift=2mm]\name.center)
    -- (\name.right) -- cycle
  }{
    (\name.left)  -- ([xshift=2mm]\name.center) -- (\name.right)
  },
}
```

The `arrow` style.

```
\sankeyset{
  new start style={arrow}{
    (\name.left) -- ++(-10pt,0) -- ([xshift=-10pt/6]\name.center)
    -- ([xshift=-10pt]\name.right) -- (\name.right) -- cycle
  }{
    (\name.left) -- ++(-10pt,0) -- ([xshift=-10pt/6]\name.center)
    -- ([xshift=-10pt]\name.right) -- (\name.right)
  },
  new end style={arrow}{
    (\name.left) -- ([yshift=1mm]\name.left)
    -- ([xshift=10pt]\name.center) -- ([yshift=-1mm]\name.right)
    -- (\name.right) -- cycle
  }{
    (\name.left) -- ([yshift=1mm]\name.left)
    -- ([xshift=10pt]\name.center) -- ([yshift=-1mm]\name.right)
    -- (\name.right)
  },
}
```

15 tikzlibrarydubins.code.tex

Not yet documented nor commented...

`\tikzlibrarydubins@version`
`\tikzlibrarydubins@date`

```
\def\tikzlibrarydubins@version{v3.0.2}
\def\tikzlibrarydubins@date{2025/01/10}
```

```
\usetikzlibrary{calc}
\RequirePackage{etoolbox}
\RequirePackage{xfp}

\newbool{dubinspathreverse}
```

`\ifpgfmathcond`

```
\def\ifpgfmathcond#1{%
  \pgfmathparse{(#1)?1:0}%
  \ifnumequal{\pgfmathresult}{1}%
}
```

`\dbp@getxy`

```
\def\dbp@getxy#1#2#3{%
  \tikz@scan@one@point\pgfutil@firstofone(#3)\relax%
  \edef#1{\the\pgf@x}%
  \edef#2{\the\pgf@y}%
}
```

`\dbp@anglebetween`

```
\def\dbp@anglebetween#1#2#3{%macro, s, t
  \dbp@getxy\dbp@ax\dbp@ay{#2}
  \dbp@getxy\dbp@bx\dbp@by{#3}
  \pgfmathsetmacro#1{atan2(\dbp@by-\dbp@ay,\dbp@bx-\dbp@ax)}
}
```

`\dbp@distancebetween`

```
\def\dbp@distancebetween#1#2#3{%macro, s, t
  \dbp@getxy\dbp@ax\dbp@ay{#2}
  \dbp@getxy\dbp@bx\dbp@by{#3}
  \edef#1{\fpeval{sqrt(%
      (\dbp@bx-\dbp@ax)*(\dbp@bx-\dbp@ax)%
      +(\dbp@by-\dbp@ay)*(\dbp@by-\dbp@ay)%
      )}}%
}
```

`\dbp@rsr`

```
\newcommand\dbp@rsr{% s, sa, t, ta, as, len, at, r
  let
  \p{tr}=([shift={(\dbp@angb-90:\dbp@radius pt)}]\dbp@b),
  \n1={\dbp@anga+90},
  \n2={\dbp@angb+90},
  \n3={\n2+\dbp@lastangle}
  in
  arc(\n1:\n1-\dbp@firstangle:\dbp@rradius pt)
  -- ([shift={(\p{tr})}]\n3:\dbp@rradius pt)
  arc(\n3:\n2:\dbp@rradius pt)
}
```

\dbp@lsl

```
\newcommand\dbp@lsl{% s, sa, t, ta, as, len, at, r
  let
  \p{tl}=([shift={(\dbp@angb+90:\dbp@radius pt)}]\dbp@b),
  \n1={\dbp@anga-90},\n2={\n1+\dbp@firstangle},
  \n3={\dbp@angb-90},\n4={\n3-\dbp@lastangle}
  in
  arc(\n1:\n2:\dbp@lradius pt)
  -- ([shift={(\p{tl})}]\n4:\dbp@lradius pt)
  arc(\n4:\n3:\dbp@lradius pt)
}
```

\dbp@rsl

```
\newcommand\dbp@rsl{% s, sa, t, ta, as, len, at, r
  let
  \p{tl}=([shift={(\dbp@angb+90:\dbp@radius pt)}]\dbp@b),
  \n1={\dbp@anga+90},\n2={\n1-\dbp@firstangle},
  \n3={\dbp@angb-90},\n4={\n3-\dbp@lastangle}
  in
  arc(\n1:\n2:\dbp@rradius pt)
  -- ([shift={(\p{tl})}]\n4:\dbp@lradius pt)
  arc(\n4:\n3:\dbp@lradius pt)
}
```

\dbp@lsr

```
\newcommand\dbp@lsr{% s, sa, t, ta, as, len, at, r
  let
  \p{tr}=([shift={(\dbp@angb-90:\dbp@radius pt)}]\dbp@b),
  \n1={\dbp@anga-90},\n2={\n1+\dbp@firstangle},
  \n3={\dbp@angb+90},\n4={\n3+\dbp@lastangle}
  in
  arc(\n1:\n2:\dbp@lradius pt)
  -- ([shift={(\p{tr})}]\n4:\dbp@rradius pt)
  arc(\n4:\n3:\dbp@rradius pt)
}
```

\dbp@lrl

```
\newcommand\dbp@lrl{% s, sa, t, ta, as, ai, at, r
  let
  \n1={\dbp@anga-90},\n2={\n1+\dbp@firstangle},
  \n3={\dbp@angb-90},\n4={\n3-\dbp@lastangle}
  in
  arc(\n1:\n2:\dbp@lradius pt)
  arc(\n2+180:\n2+180-\dbp@midparam:\dbp@rradius pt)
  arc(\n4:\n3:\dbp@lradius pt)
}
```

\dbp@rlr

```
\newcommand\dbp@rlr{% s, sa, t, ta, as, ai, at, r
  let
  \n1={\dbp@anga+90},\n2={\n1-\dbp@firstangle},
  \n3={\dbp@angb+90},\n4={\n3+\dbp@lastangle}
  in
  arc(\n1:\n2:\dbp@rradius pt)
  arc(\n2+180:\n2+180+\dbp@midparam:\dbp@lradius pt)
  arc(\n4:\n3:\dbp@rradius pt)
}
```

\dbp@rev@lsl

```
\newcommand\dbp@rev@lsl{\dbp@rsr}
```

\dbp@rev@rsr

```
\newcommand\dbp@rev@rsr{\dbp@lsl}
```

\dbp@rev@lsr

```
\newcommand\dbp@rev@lsr{\dbp@lsr}
```

\dbp@rev@rsl

```
\newcommand\dbp@rev@rsl{\dbp@rsl}
```

\dbp@rev@lrl

```
\newcommand\dbp@rev@lrl{\dbp@rlr}
```

\dbp@rev@rlr

```
\newcommand\dbp@rev@rlr{\dbp@lrl}
```

\dubinspath

```
\newcommand\dubinspath[1]{%
  \pgfextra{
    \dubinspathset{#1}
    \ifbool{dubinspathreverse}{
      \edef\dbp@newa{\dbp@b}
      \edef\dbp@newb{\dbp@a}
      \pgfmathsetmacro\dbp@newanga{180+\dbp@angb}
      \pgfmathsetmacro\dbp@newangb{180+\dbp@anga}
      \edef\dbp@newfirstangle{\dbp@lastangle}
      \edef\dbp@newlastangle{\dbp@firstangle}
      \edef\dbp@newmethod{rev@\dbp@method}
      \edef\dbp@newlradius{\dbp@rradius}
      \edef\dbp@newrradius{\dbp@lradius}
      \dubinspathset{
        start point=\dbp@newa,
        end point=\dbp@newb,
        start angle=\dbp@newanga,
        end angle=\dbp@newangb,
        first angle=\dbp@newfirstangle,
        last angle=\dbp@newlastangle,
        left and right minimum radii=\dbp@newlradius pt and \dbp@newrradius pt,
        method=\dbp@newmethod,
      }
    }{}
  }
  \csname dbp@\dbp@method\endcsname%
}
```

\dbp@store
\dbp@get

```
\def\dbp@store#1#2{%
  \expandafter\xdef\csname dbp@store@#1@#2\endcsname%
  {\csname dbp@#2\endcsname}%
}
\def\dbp@get#1#2{%
  \csname dbp@store@#1@#2\endcsname%
}
```

\dbp@setparams

```
\def\dbp@setparams#1#2#3#4#5{%
  % method, length, fisrt angle, middle param, last angle
  \edef\dbp@method{#1}
  \edef\dbp@length{#2}
  \edef\dbp@firstangle{#3}
  \edef\dbp@middleparam{#4}
  \edef\dbp@lastangle{#5}
  \ifdef{\dbp@storename}{
    \foreach \p in {method,length,firstangle,middleparam,lastangle}{
      \dbp@store{\dbp@storename}{\p}
    }
  }{}
}
```

\dbp@updateparams

```
\def\dbp@updateparams#1#2#3#4#5{
  \ifpgfmathcond{#2<\dbp@length}{
    \dbp@setparams{#1}{#2}{#3}{#4}{#5}
  }{}
}
```

```
\tikzset{
  dubins path/.is family,
  dubins path,
  start point/.store in=\dbp@a,
  start angle/.store in=\dbp@anga,
  end point/.store in=\dbp@b,
  end angle/.store in=\dbp@angb,
  store/.store in=\dbp@storename,
  use store/.style={
    method=\dbp@get{#1}{method},
    first angle=\dbp@get{#1}{firstangle},
    last angle=\dbp@get{#1}{lastangle},
    middle param=\dbp@get{#1}{middleparam},
  },
  minimum radius/.code={
    \pgfmathsetmacro\dbp@radius{#1}
    \pgfmathsetmacro\dbp@rradius{#1}
    \pgfmathsetmacro\dbp@lradius{#1}
  },
  left and right minimum radii/.code args={#1 and #2}{
    \pgfmathsetmacro\dbp@lradius{#1}
    \pgfmathsetmacro\dbp@rradius{#2}
    \pgfmathsetmacro\dbp@radius{(\dbp@lradius + \dbp@rradius)/2}
  },
  method/.store in=\dbp@method,
  first angle/.store in=\dbp@firstangle,
  last angle/.store in=\dbp@lastangle,
  middle param/.store in=\dbp@midparam,
  reverse/.is if=dubinspathreverse,
}
```

\dubinspathset

```
\newcommand\dubinspathset[1]{\tikzset{dubins path,#1}}
```

\dubinspathcalc

```
\newcommand\dubinspathcalc[1]{%
  \begingroup
  \dubinspathset{#1}
  \tikzset{
    declare function={
      angtodist(\dbp@a,\dbp@r)={abs(\dbp@a)*.01745329*\dbp@r};
      modangr(\dbp@a,\dbp@b)={
        (
        Mod(\dbp@a,360)<Mod(\dbp@b,360)
        ?
        Mod(\dbp@a,360)
        :
        Mod(\dbp@a,360)-360)+\dbp@b-Mod(\dbp@b,360
        )
      };
      modangl(\dbp@a,\dbp@b)={
        (
        Mod(\dbp@a,360)<Mod(\dbp@b,360)
        ?
        Mod(\dbp@a,360)+360
        :
        Mod(\dbp@a,360))+(\dbp@b)-Mod(\dbp@b,360)
      };
    },
  }

  \pgfmathsetmacro\dbp@radius{\dbp@rradius}
  \pgfmathsetmacro\dbp@anga{mod((\dbp@anga)+180,360)-180}
  \pgfmathsetmacro\dbp@angb{mod((\dbp@angb)+180,360)-180}
  \path
  let
  \p{a}=(\dbp@a),
  \p{b}=(\dbp@b),
  \p{ar}=($(\p{a}) + (\dbp@anga-90:\dbp@radius pt)$),
  \p{al}=($(\p{a}) + (\dbp@anga+90:\dbp@radius pt)$),
  \p{br}=($(\p{b}) + (\dbp@angb-90:\dbp@radius pt)$),
  \p{bl}=($(\p{b}) + (\dbp@angb+90:\dbp@radius pt)$)
  in \pgfextra{
    \pgfinterruptpath

    % RSR (ar and br)
    \dbp@anglebetween\dbp@rsrarbr{\p{ar}}{\p{br}}
    \dbp@distancebetween\dbp@rsrdarbr{\p{ar}}{\p{br}}
    \pgfmathsetmacro\dbp@rsrangone{Mod(\dbp@anga-\dbp@rsrarbr,360)}
    \pgfmathsetmacro\dbp@rsrangtwo{Mod(\dbp@rsrarbr-\dbp@angb,360)}
    \pgfmathsetmacro\dbp@rsrlen{\dbp@rsrdarbr}
    \pgfmathsetmacro\dbp@rsrdist{
      angtodist(\dbp@rsrangone,\dbp@radius)
      +\dbp@rsrlen
      +angtodist(\dbp@rsrangtwo,\dbp@radius)
    }
    \dbp@setparams%
    {rsr}{\dbp@rsrdist}{\dbp@rsrangone}{\dbp@rsrlen}{\dbp@rsrangtwo}

    % LSL (al and bl)
    \dbp@anglebetween\dbp@lslalbl{\p{al}}{\p{bl}}
    \dbp@distancebetween\dbp@lsldalbl{\p{al}}{\p{bl}}
    \pgfmathsetmacro{\dbp@lslangone}{mod(\dbp@lslalbl-\dbp@anga+720,360)}
    \pgfmathsetmacro{\dbp@lslangtwo}{mod(\dbp@angb-\dbp@lslalbl+720,360)}
    \pgfmathsetmacro\dbp@lsllen{\dbp@lsldalbl}
    \pgfmathsetmacro\dbp@lsldist{
      angtodist(\dbp@lslangone,\dbp@radius)
      +\dbp@lsllen
      +angtodist(\dbp@lslangtwo,\dbp@radius)
    }
    \dbp@updateparams%
    {lsl}{\dbp@lsldist}{\dbp@lslangone}{\dbp@lsllen}{\dbp@lslangtwo}

    % RSL (ar and bl)
    \dbp@distancebetween\dbp@rsldarbl{\p{ar}}{\p{bl}}
    \pgfmathtruncatemacro\dbp@rslok{(\dbp@rsldarbl>=2*\dbp@radius)?1:0}
    \ifnumequal{\dbp@rslok}{1}{
      \dbp@anglebetween\dbp@rslarbl{\p{ar}}{\p{bl}}
      \pgfmathsetmacro\dbp@rslanglesup{
        asin(\dbp@radius/\dbp@rsldarbl*2)}
      \pgfmathsetmacro\dbp@rslangone
      {Mod(\dbp@anga-\dbp@rslarbl+\dbp@rslanglesup,360)}
      \pgfmathsetmacro\dbp@rslangtwo
      {Mod(\dbp@angb-\dbp@rslarbl+\dbp@rslanglesup,360)}
      \pgfmathsetmacro\dbp@rsllen{veclen(\dbp@rsldarbl,\dbp@radius)}
      \pgfmathsetmacro\dbp@rsldist{
        angtodist(\dbp@rslangone,\dbp@radius)
        +\dbp@rsllen
        +angtodist(\dbp@rslangtwo,\dbp@radius)
      }

      \dbp@updateparams%
      {rsl}{\dbp@rsldist}{\dbp@rslangone}{\dbp@rsllen}{\dbp@rslangtwo}
    }{}

    % LSR (al and br)
    \dbp@distancebetween\dbp@lsrdalbr{\p{al}}{\p{br}}
    \pgfmathtruncatemacro\dbp@lsrok{(\dbp@lsrdalbr>=2*\dbp@radius)?1:0}
    \ifnumequal{\dbp@lsrok}{1}{
      \dbp@anglebetween\dbp@lsralbr{\p{al}}{\p{br}}
      \pgfmathsetmacro\dbp@lsranglesup{
        asin(\dbp@radius/\dbp@lsrdalbr*2)}
      \pgfmathsetmacro\dbp@lsrangone
      {Mod(\dbp@lsralbr+\dbp@lsranglesup-\dbp@anga,360)}
      \pgfmathsetmacro\dbp@lsrangtwo
      {Mod(\dbp@lsralbr+\dbp@lsranglesup-\dbp@angb,360)}
      \pgfmathsetmacro\dbp@lsrlen{veclen(\dbp@lsrdalbr,\dbp@radius)}
      \pgfmathsetmacro\dbp@lsrdist{
        angtodist(\dbp@lsrangone,\dbp@radius)
        +\dbp@lsrlen
        +angtodist(\dbp@lsrangtwo,\dbp@radius)
      }
      \dbp@updateparams%
      {lsr}{\dbp@lsrdist}{\dbp@lsrangone}{\dbp@lsrlen}{\dbp@lsrangtwo}
    }{}

```

```
    % LRL (al and bl)
    \dbp@distancebetween\dbp@lrldalbl{\p{al}}{\p{bl}}
    \pgfmathtruncatemacro\dbp@lrlok{(\dbp@lrldalbl<=4*\dbp@radius)?1:0}
    \ifnumequal{\dbp@lrlok}{1}{
      \dbp@anglebetween\dbp@lrlalbl{\p{al}}{\p{bl}}
      \pgfmathsetmacro\dbp@lrlangsup{acos(\dbp@lrldalbl/\dbp@radius/4)}
      \pgfmathsetmacro\dbp@lrlangone{
        modangl(\dbp@lrlalbl+\dbp@lrlangsup,\dbp@anga-90)-(\dbp@anga-90)}
      \pgfmathsetmacro\dbp@lrlangtwo{%
        (\dbp@angb-90)-modangr(\dbp@lrlalbl+180-\dbp@lrlangsup,\dbp@angb-90)}
      \pgfmathsetmacro\dbp@lrlangthree{360-2*(90-\dbp@lrlangsup)}
      \pgfmathsetmacro\dbp@lrldist{
        angtodist(\dbp@lrlangone,\dbp@radius)
        +angtodist(\dbp@lrlangthree,\dbp@radius)
        +angtodist(\dbp@lrlangtwo,\dbp@radius)
      }
      \dbp@updateparams%
      {lrl}{\dbp@lrldist}{\dbp@lrlangone}{\dbp@lrlangthree}{\dbp@lrlangtwo}
    }{}

    % RLR (ar and br)
    \dbp@distancebetween\dbp@rlrdarbr{\p{ar}}{\p{br}}
    \pgfmathtruncatemacro\dbp@rlrok{(\dbp@rlrdarbr<=4*\dbp@radius)?1:0}
    \ifnumequal{\dbp@rlrok}{1}{
      \dbp@anglebetween\dbp@rlrarbr{\p{ar}}{\p{br}}
      \pgfmathsetmacro\dbp@rlrangsup{acos(\dbp@rlrdarbr/\dbp@radius/4)}
      \pgfmathsetmacro\dbp@rlrangone{
        (\dbp@anga+90)-modangr(\dbp@rlrarbr-\dbp@rlrangsup,\dbp@anga+90)}
      \pgfmathsetmacro\dbp@rlrangtwo{%
        modangl(\dbp@rlrarbr+180+\dbp@rlrangsup,\dbp@angb+90)-(\dbp@angb+90)}
      \pgfmathsetmacro\dbp@rlrangthree{360-2*(90-\dbp@rlrangsup)}
      \pgfmathsetmacro\dbp@rlrdist{
        angtodist(\dbp@rlrangone,\dbp@radius)
        +angtodist(\dbp@rlrangthree,\dbp@radius)
        +angtodist(\dbp@rlrangtwo,\dbp@radius)
      }
      \dbp@updateparams%
      {rlr}{\dbp@rlrdist}{\dbp@rlrangone}{\dbp@rlrangthree}{\dbp@rlrangtwo}
    }{}

    \endpgfinterruptpath
  };
  \endgroup
}
```

16 Change History

v1.0 – 2016/03/06
- General: First version 44

v2.0 – 2021/01/27
- General: First public version (on CTAN) 44

v3.0 – 2021/03/14
- General: Add keys to fork a Sankey node during its creation 19
  - Add the reproduction of an example from Google Charts documentation 32
  - Better naming rule 44
  - Fix bad names in second example 28
  - Use `.ins` and `.dtx` files 44
- `\snk@sankeyturn`: Simplification by using new *turn* macros 58
- `\snk@sankeyturnleft`: New macro . . 57
- `\snk@sankeyturnleftbackward`: New macro 57
- `\snk@sankeyturnright`: New macro . 56
- `\snk@sankeyturnrightbackward`: New macro 56

v3.0.1 – 2022/02/04
- General: Add instructions for compiling and installing the package 43
  - Fix options used by the new version of `siunitx` package . . . 43

v3.0.2 – 2025/01/10
- `sankeydiagram`: Simplification of the environment by copying internal macros 59

v3.0.2 – 2025/10/10
- General: Fixed extra parentheses in some code examples 2
  - Update for new version of `ltxdoc` class 2
- `\snk@init@move`: Factorization of the node move initialization code . . 55

17 Index

Italic numbers refer page where the corresponding entry is described; Underlined numbers refer to the code line of the definition; Roman numbers refer to the code lines where the entry is used.