Ti*k*Z tensor network diagrams

Jesse Osborne and Ian McCulloch

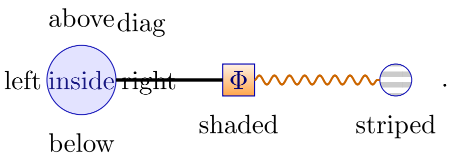
22 November 2025

Customization cheatsheet

The package exposes keys that can be configured globally via `\tnsetup{<key>=<value>}` and mixed locally inside each `tikzpicture`. Useful keys include `tensorsize`, `tensor fill`, `tensor draw`, `tensor text`, `tensor opacity`, `shade top`, `shade bottom`, `stripesize`, `stripe density`, `xscale`, `yscale`, `outer xsep`, and `outer ysep`. Line styles can be supplied directly to `\draw` (for example `[very thick]` or `[wavy]`). Labels follow TikZ's `label=<placement>:<text>` syntax and accept multiple placements on the same node.

Colors, shading, and labels

Set default colours and opacity with `\tnsetup`. Use the built-in `shadedtensor` or `striped` styles for quick decorations, and place labels above, below, left, right, or diagonally using TikZ's label anchors. The text inside the node is written between the curly braces of `\node`.



(1)

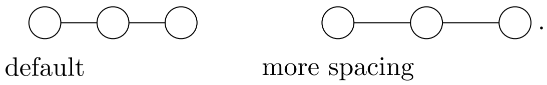
Minimal template for the diagram above:

```
\tnsetup{tensor fill=blue!12,
         tensor draw=blue!70!black,
         shade top=orange!10}
\begin{tikzpicture}[tensornetwork]
  \node[atensor,
        label=above:{above},
        label={[label distance=3pt]45:{diag}}] (A) at (0, 0) {inside};
  \node[shadedtensor] (B) at (2.2, 0) {$\Phi$};
  \node[ctensor, striped] (C) at (4.4, 0) {};
  \draw[very thick] (A) -- (B);
  \draw[wavy, orange!80!black, thick] (B) -- (C);
\end{tikzpicture}
```

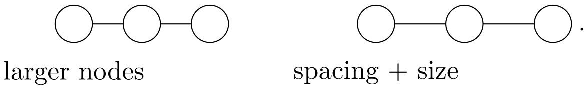
Sizing diagrams

TikZ's `scale=<number>` multiplies coordinates and line lengths (it spreads nodes apart) but leaves tensor shapes at their configured size. The package key `/tensor network/tensorsize=<length>`

directly controls the drawn radius of tensors and the padding used to place their labels. Combine them to enlarge a figure while keeping labels legible and consistent across different scalings.



(2)



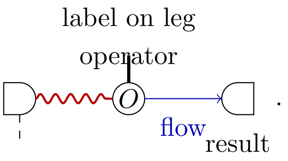
(3)

Template used above:

```
\begin{tikzpicture}[tensornetwork, scale=1.3,
                    /tensor network/tensorsize=14pt]
  \node[atensor] (A1) at (0, 0) {};
  \node[atensor] (A2) at (1, 0) {};
  \node[atensor] (A3) at (2, 0) {};
  \draw (A1) -- (A2) -- (A3);
\end{tikzpicture}
```

Line styles and annotations

All TikZ line styles work out of the box. The example below mixes dashed legs, a thick leg labelled inline, and a wavy connector for emphasis.



(4)

Template used above:

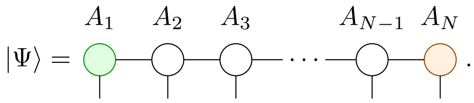
```
\begin{tikzpicture}[tensornetwork]
  \node[ltensor] (L) at (0, 0) {};
  \node[atensor, label=above:{operator}] (O) at (1.6, 0) {$O$};
  \node[rtensor, label=below:{result}] (R) at (3.2, 0) {};
  \draw[dashed] (L.south) -- +(0, -0.6);
  \draw[very thick] (O.north) -- +(0, 0.6) node[above] {label on leg};
  \draw[wavy, red!70!black, thick] (L) -- (O);
  \draw[->, blue!70!black] (O) -- (R) node[midway, below] {flow};
\end{tikzpicture}
```

Worked examples

The remaining diagrams echo the ones in the package README but include brief guidance on how to adapt them. The defaults use neutral black strokes and white fills; combine them with `\tnsetup` keys such as `tensor fill=gray!8` or local TikZ options such as `[rounded corners]` to introduce emphasis.

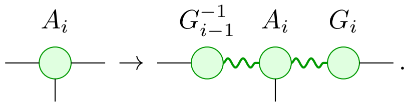
Finite MPS: the circles mark site tensors and the short legs denote physical indices. Adjust colours or leg thickness with `\tnsetup` or local `[very thick]` modifiers to highlight boundaries.

Below, the boundary tensors are tinted to show how local colour overrides look.



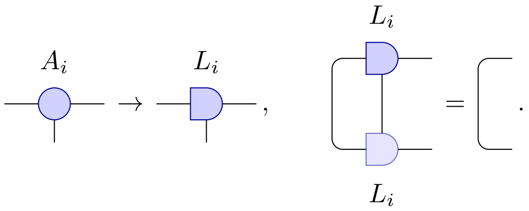
(5)

Gauge transform: swap into a new gauge by inserting invertible tensors. Here the gauge boxes use green fills and wavy internal bonds to illustrate local styling; use `\tnsetup{tensor fill=green!10, tensor draw=gr` to change defaults instead.



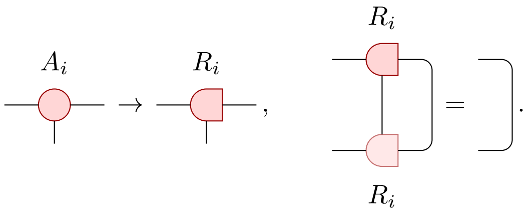
(6)

Left-orthogonal form: left-canonical tensors appear as rounded rectangles. You can also make the conjugate copy translucent via `tensor opacity=0.7` if you want to de-emphasise it in print; here the conjugate copy is drawn with a tinted, semi-transparent fill.



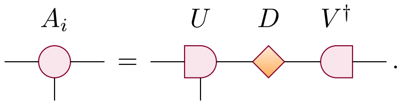
(7)

Right-orthogonal form: mirror of the above. As with all diagrams, leg lengths can be tweaked with explicit coordinates such as `--` (0.7, 0)+ if you need more room for labels. This version tints the right-canonical tensors red and leaves the conjugate copy slightly faded.



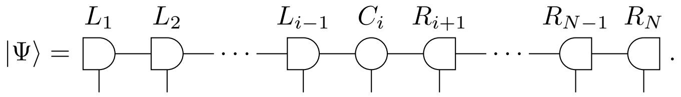
(8)

SVD: diagonal and unitary factors use distinctive node shapes. Add shading to emphasise the spectrum, e.g. `[shadedtensor]` on D; the diagram below also colours the unitaries for readability.

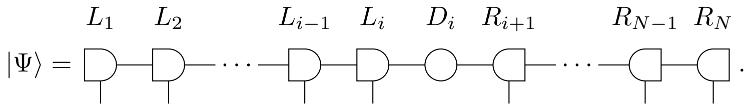


(9)

Mixed-canonical form: compare the default spacing (outer ysep close to the package default) with a roomier choice such as `outer ysep=9pt` to accommodate multi-line labels.

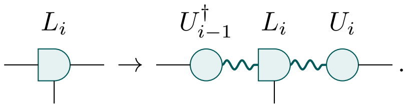


(10)



(11)

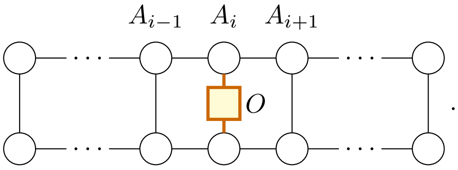
Unitary gauge transformation: use the diamond nodes for unitary tensors, optionally adding `[wavy]` legs to distinguish internal bonds. Here the unitaries are coloured teal and connected with wavy links.



(12)

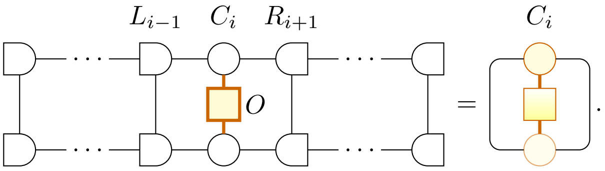
Expectation value: highlight the inserted operator O by colouring the square node (e.g. `[tensor fill=yellow!20]`) or by thickening its attached legs.

$\langle\Psi|O_i|\Psi\rangle =$



(13)

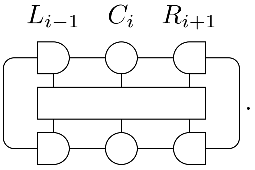
$\langle\Psi|O_i|\Psi\rangle =$



(14)

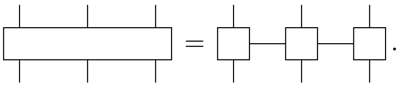
Multi-site expectation value:

$\langle\Psi|O|\Psi\rangle =$



(15)

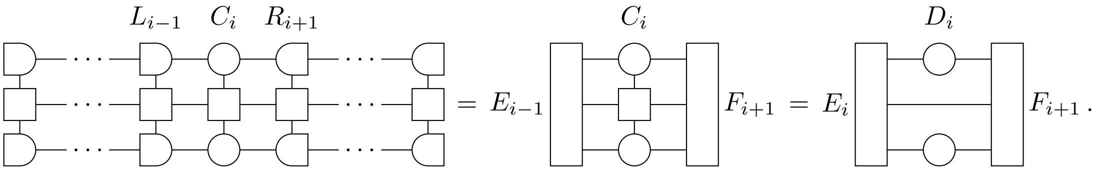
MPO:



(16)

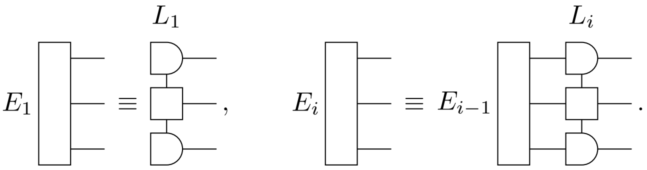
MPO expectation value: combine the ideas above. Long operators or multi-site gates can reuse the `widetensor` style (e.g. `[widetensor=2]`) to stretch horizontally.

$\langle\Psi|H|\Psi\rangle =$

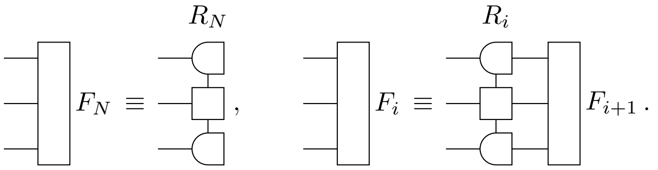


(17)

Environment tensors:

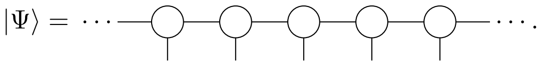


(18)



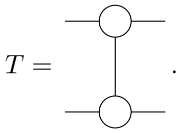
(19)

iMPS:



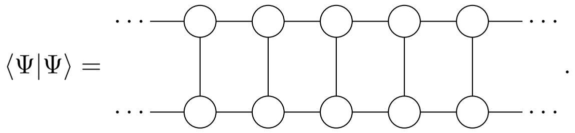
(20)

Transfer matrix:



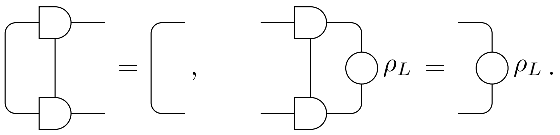
(21)

MPS norm:



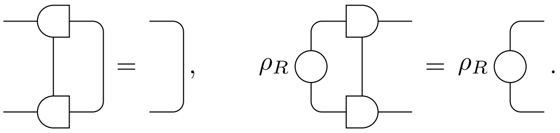
(22)

Left-orthogonal form:



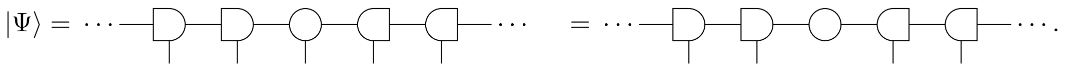
(23)

Right-orthogonal form:



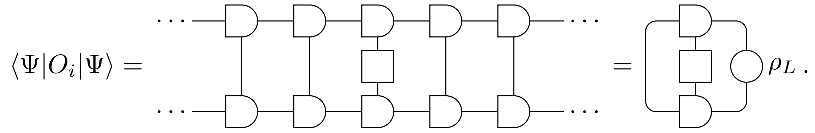
(24)

Mixed-canonical form:

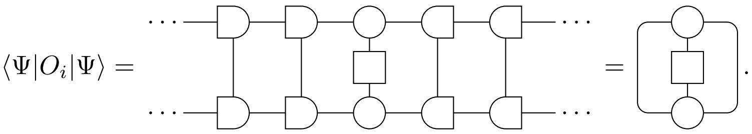


(25)

iMPS expectation value:

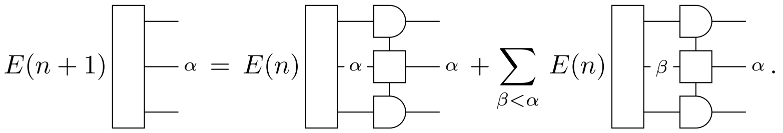


(26)



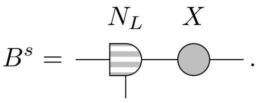
(27)

Environment tensor recursion relation:



(28)

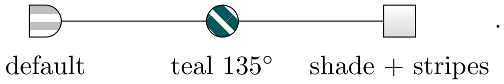
Patterns for tensors: stripes are helpful for marking environments or normalisation factors. Adjust the stripe spacing with `\tnsetup{stripesize=<length>}`, control the fill-versus-gap ratio via `stripe density` (default 0.5 for equal widths), rotate them with `/tensor network/stripe angle=<degrees>`, and recolour on a per-node basis via `[pattern color=<colour>]`. You can even combine the pattern with shading for extra emphasis.



(29)

Template for varying stripe weight, colour, and angle (using a denser 6pt stripe spacing so multiple stripes fit inside each node):

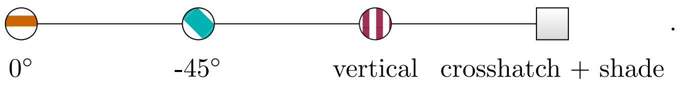
```
\tnsetup{stripesize=6pt, stripe density=0.55}
\begin{tikzpicture}[tensornetwork]
  \node[ltensor, striped] (A1) at (0, 0) {};
  \node[ctensor,
        /tensor network/stripe density=0.65,
        striped,
        /tensor network/stripe angle=135,
        pattern color=teal!70] (A2) at (2.6, 0) {};
  \node[shadedtensor, striped,
        pattern color=red!70] (A3) at (5.2, 0) {};
  \draw (A1) -- (A2) -- (A3);
\end{tikzpicture}
```



(30)

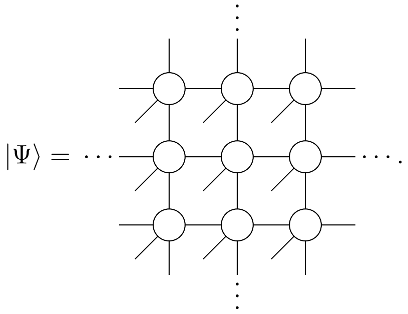
Additional pattern ideas with `patterns.meta`: change the line angle, switch to crosshatching, vary `stripe density`, or overlay a pattern on a shaded tensor. The vertical and crosshatch examples below choose equal-width strokes and gaps (50% density) with thicker lines so the pattern remains visible over shading.

```
\begin{tikzpicture}[tensornetwork, /tensor network/stripesize=10pt]
  \node[ctensor,
        /tensor network/stripe density=0.4,
        striped,
        pattern color=orange!80!black] (B1) at (0, 0) {};
  \node[ctensor,
        /tensor network/stripe density=0.7,
        striped,
        /tensor network/stripe angle=-45,
        pattern color=cyan!70!black] (B2) at (2.6, 0) {};
  \node[ctensor,
        pattern={Lines[angle=90,distance=5pt]},
        pattern color=purple!70!black,
        /tensor network/tensor opacity=0.9] (B3) at (5.2, 0) {};
  \node[shadedtensor,
        pattern={Hatch[distance=5pt,line width=1.4pt]},
        pattern color=gray!80!black] (B4) at (7.8, 0) {};
  \draw (B1) -- (B2) -- (B3) -- (B4);
\end{tikzpicture}
```



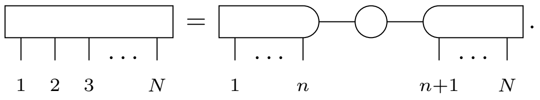
(31)

iPEPS:



(32)

Schmidt decomposition:



(33)