

Removing vertical stretch – mimicking traditional typesetting with T_EX

Kaveh Bazargan, CV Radhakrishnan

Abstract

One of T_EX's advantages over traditional typesetting systems is the mechanism to stretch horizontal and vertical glue as needed, in order to aid paragraph building and pagination. But all T_EX operators involved in day-to-day page make-up know that this inbuilt intelligence is often 'too clever' and frustrating for the user. T_EX will not do what you want it to do, and only over many years can operators gain the knowledge that allows them to make just the right change to the source code in order to coerce T_EX to produce the desired result.

Recently we have been experimenting with removing vertical stretchability, with promising results. Our approach is to round off the height of all vertical material, including floats and displayed equations, to be an integral number of the leading of the main text. One advantage is that this allows true 'grid' setting in double column text.

1 Some things we have always wanted

It is useful to look at some things we have always wanted to do in T_EX but found difficult.

1.1 More control over glue

Anyone involved with 'real world' pagination of T_EX and L^AT_EX files is aware of the frustration that T_EX's 'glue' can generate. T_EXies have long learned to accept this limitation, and developed tricks to work efficiently. However, some apparently simple tasks are still mysteriously complex to the non-T_EXie. For example, in figure 1 suppose that two lines have to move from the first to the second column. Logic would imply that two lines from the second column would have to move to the next page. But in T_EX this rarely happens. For instance the glue around the displayed equation might shrink or stretch instead.

1.2 Grid setting

One of the most frequent complaints about T_EX when setting double column text is that normally, the lines of text are not set on a grid. In other words, the baselines of one column do not align with those of the other. This is another consequence of the inherent stretchability of T_EX's glue mechanism.

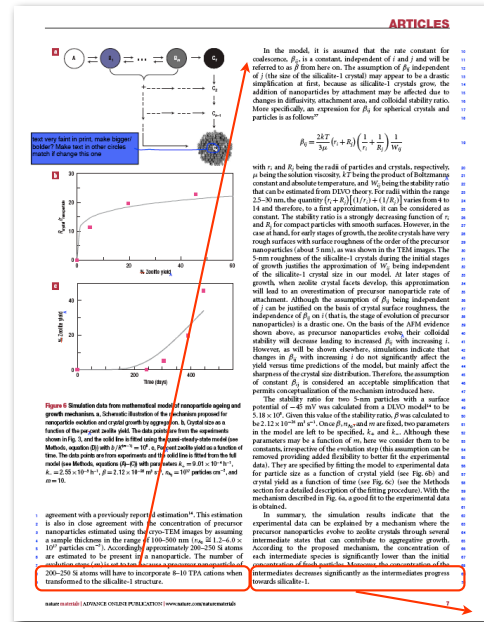


Figure 1: In T_EX documents, taking two lines over from the first column does not necessarily result in two lines from the second column moving over.

1.3 Precise control over positioning of graphics

This is another related problem. Suppose we want to move a floating graphic slightly higher or lower, without affecting pagination. For example we might want the top of a graphic appearing at the beginning of a column to be moved up a fraction in order to align with the top of the text in the next column. With the usual L^AT_EX commands this is not easy. I am mentioning this here as our macros for controlling glue allowed us to control the exact positioning of graphics too.

2 How T_EX makes up pages

Figure 2 shows T_EX's normal mechanism for setting a page. First of all the boxes are arranged in the vertical list, spaced out by the natural height of the glues assigned. Then, T_EX stretches or shrinks the glues so as to fit the boxes and make the last baseline align with the bottom of the page. As is evident, it is difficult to control the positions of the baselines using this method.

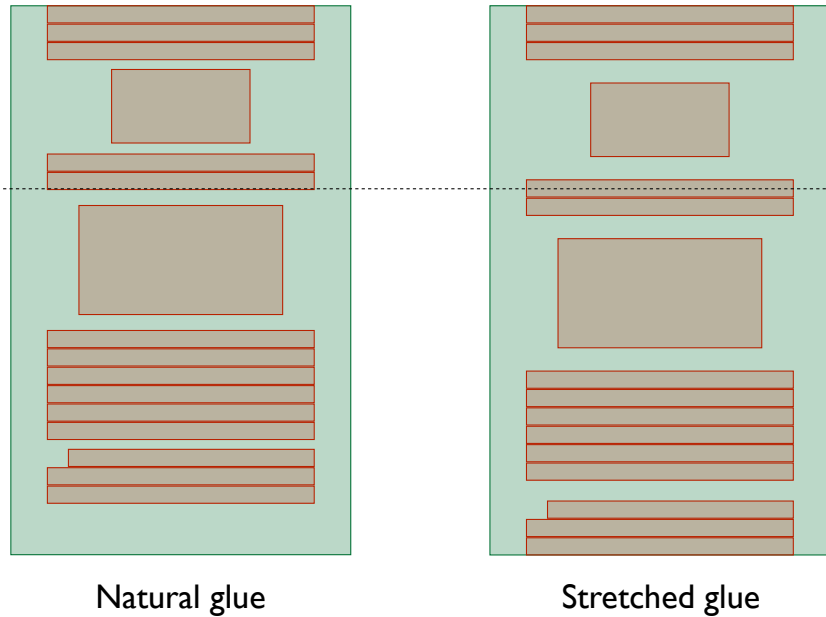


Figure 2: Stretching of glue to fit text to a page. The positions of the baselines are hard to predict when glue has stretchability.

3 Our approach to the solution

At River Valley, we have thought about and discussed extensively the methods we might use to effect grid setting. These include testing each box in the vertical list on the fly, and tweaking the vertical position. We tried to do this, both using T_EX macros, and also doing it at compiler level, using pdfT_EX. Unfortunately this approach did not work.

The more successful strategy was to try and make sure that all items in the vertical list had heights which were integral multiples of the value of `\baselineskip`. Examples of these items are:

- Floating elements (e.g. figures and tables)
- Displayed equations
- Section headers
- All skips between paragraphs, etc

Figure 3 shows some of the many glue parameters that are inserted in the vertical list and which must conform to this rule.

Let's look in more detail at how we dealt with specific issues.

3.1 Glue at the top of each column

When T_EX starts typesetting a page, it inserts a glue called `\topskip` at the top of the column. The value of this glue is derived from a complex formula involving the elements in the first line. To simplify matters, we set `\topskip` to the value of `\baselineskip` which simplified matters considerably and did not produce any problems.

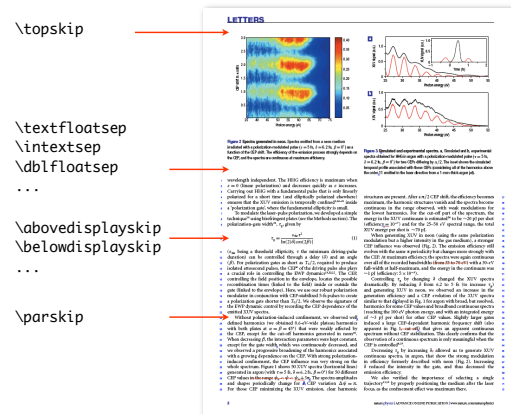


Figure 3: Some of the many vertical glues that can affect pagination. In general, baselines of the two columns are not aligned on a grid.

3.2 Stretching baseline glue

`\lineskip` and `\lineskiplimit` are two parameters that can cause big headaches in grid setting. Here is the logic: T_EX sets paragraphs by putting one line above another, normally spacing out lines by the value of `\baselineskip`. The exception comes when T_EX thinks two adjacent lines might clash. In particular, T_EX examines the depth of each line of text and the height of the succeeding line. If, when placing the usual `\baselineskip`, T_EX finds that the boundaries of the boxes containing the adjacent lines is closer than the value of `\lineskiplimit`, then the normal procedure is aborted, and a glue equal to the

Here is some text to show what happens when we have a large mathematical construct in a line of text. Here is some text to show what happens when we have a large mathematical construct in a line of text. Here is some text to show what happens when we have a large mathematical construct in a line of text. $\int_0^{\frac{\pi}{4}} \frac{1}{4} + 2 + 3$. Here is some text to show what happens when $\int_0^{\frac{\pi}{4}} \frac{1}{4} + 2 + 3$ we have a large mathematical construct in a line of text. Here is some text to show what happens when we have a large mathematical construct in a line of text.

Figure 4: Variation of leading when large inline maths is present.

value of `\lineskip` is inserted. As we can see from figure 4, this can result in variable line spacing, making grid setting impossible.

We looked at the instances where `\lineskip` had been applied. In most cases, the normal leading would have sufficed, as the oversized elements were not aligned with each other. T_EX does not know the horizontal position of the offending boxes, so in general there is no clash of text. Even when they were aligned, in most cases we could avoid the clash by reformatting the paragraph. The publications we were considering for grid setting contained only light mathematics, so we decided that we would do away with using `\lineskip` and just keep an eye out for clashing items, and deal with them on a case by case basis. So we chose the following values:

```
\lineskiplimit = -10pt
\lineskip = 0pt
```

The negative value of `\lineskiplimit` instructs T_EX not to apply `\lineskip` unless there is an overlap of more than 10pt between two adjacent lines. The value given to `\lineskip` is unimportant. Of course this might give very ugly overlapping lines, but we would pick these up while checking proofs, and we would deal with them manually. For seriously overlapping maths, we would change from inline to display math.

3.3 Dealing with floating elements.

Floating elements, such as figures or tables, generally consist of the main float, e.g. the graphic or the table, a caption, and three glue elements, one above the complete float, one below, and one separating the main element from the caption. In order to maintain grid setting, we need to control the vertical size of all these five elements. The three glue elements are set such that the total natural height is equal to an integral number of `\baselineskips`. The main element and the caption are rounded up or down, so that their heights are also an integral

number of `\baselineskips`. This is done through a `\roundoff` macro that is executed at run time.

The general macro for floats is as follows:

```
\begin{figure}
\centering
\XFigure{figure1}
{\label{fig1}Caption of figure.}
\end{figure}
```

which is similar to the normal L^AT_EX float macro. We have made the control of spacing more useful by using `keyval.sty`, and adding the following options:

```
[beforegr = ...pt]
[aftergr = ...pt]
[beforecap = ...pt]
[aftercap = ...pt]
[line = ...]
```

These options allow the main element or the caption to be moved up or down. This is done before the `\roundoff` macro is executed, so the final height of each element will still be an integral number of `\baselineskips`. The final option is a negative or positive integer, and adjusts the three glue parameters such that the complete height of the float is an integral number of lines larger or smaller. This is useful in solving pagination problems.

3.4 Displayed equations

For equations that do not break across pages, we again use `\roundoff` to make them fit the grid. The display glues such as `\abovedisplayskip` are set to fixed amounts, as before.

3.5 The one problem: Breaking displays

There is one problem we have not solved yet, namely that of automatically breaking displayed equations, and maintaining grid setting. The problem is that `\roundoff` produces one T_EX box, so T_EX's normal page breaking mechanism cannot be applied. For manuscripts with light mathematics, this is not a major problem, as the few such occurrences can be fixed manually, but it would be good to have an automated solution.

4 Results

Figures 5 and 6 shows a double column page set on a grid. The floating figure and the mathematics are all rounded off so that the text is set on a grid.

We have found that the time taken in pagination has reduced considerably after using the grid macros. When we need to move some lines from one column to the next, it results in exactly the same number of lines being taken over from the second column. The `keyval` options in floats allows us to deal easily with widows and orphans, by making a float one line longer or shorter.

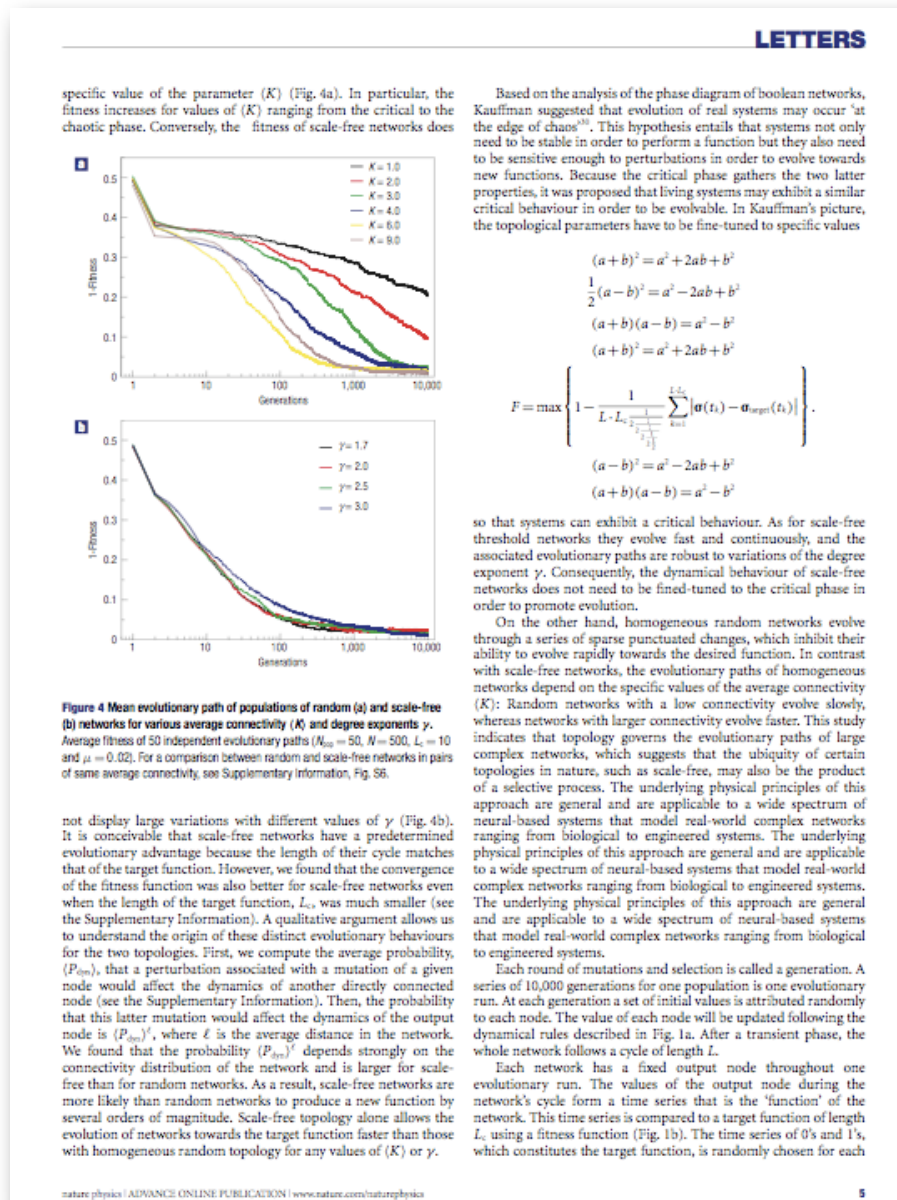


Figure 5: Example of a double column page set on a grid.

evolutionary run and is kept fixed throughout the run. At each generation we randomly choose different initial conditions for each network so that we could select for networks that are robust to variations of initial conditions.

While there is no measurable improvement of the fitness, neutral

$$a + b = x$$

mutations modify connections and weights in the connectivity

$$\frac{w}{r} = \frac{e}{r}$$

matrix of the networks within the population but do not affect the function. When an additional advantageous mutation occurs,

$$F = \max \left\{ 1 - \frac{1}{L \cdot L_c} \sum_{k=1}^{L \cdot L_c} |\sigma(t_k) - \sigma_{\text{target}}(t_k)| \right\}.$$

the number of copies of the associated mutant grows exponentially with the number of generations and becomes dominant in the population local optima with very few mutations. Many mutated networks with

$$(a + b)^2 = a^2 + 2ab + b^2 \quad (21)$$

$$\frac{1}{2}(a - b)^2 = a^2 - 2ab + b^2 \quad (22)$$

generations, the population consists of many different functions which can also have different cycle lengths (Fig. 2b). We interpret the diversity of functions in scale-free networks as the signature of the fitness landscape that allows the population to escape local optima with very few mutations. Many mutated networks with

conditions and mutations exhibit similar evolutionary paths. We find that evolutionary paths of random networks depend on rare advantageous mutative events and thus differ from one another (Fig. 3a). As for scale-free networks, the distribution of independent not only evolves faster than that of random networks but also

$$(a + b)^2 = a^2 + 2ab + b^2 \quad (27)$$

$$\frac{1}{2}(a - b)^2 = a^2 - 2ab + b^2 \quad (28)$$

$$(a + b)(a - b) = a^2 - b^2 \quad (29)$$

$$(a + b)^2 = a^2 + 2ab + b^2 \quad (30)$$

$$F = \max \left\{ 1 - \frac{1}{L \cdot L_c} \sum_{k=1}^{L \cdot L_c} |\sigma(t_k) - \sigma_{\text{target}}(t_k)| \right\} \quad (31)$$

$$(a - b)^2 = a^2 - 2ab + b^2 \quad (32)$$

$$(a + b)(a - b) = a^2 - b^2 \quad (33)$$

has the capacity to produce a wide range of heritable functions²⁷.

Functions of high fitness emerge by evolving existing functions gradually towards the target function.

$$(a + b)^2 = a^2 + 2ab + b^2 \quad (34)$$

$$\frac{1}{2}(a - b)^2 = a^2 - 2ab + b^2 \quad (35)$$

$$(a + b)^2 = a^2 + 2ab + b^2 \quad (36)$$

Functions of high fitness emerge by evolving existing functions gradually towards the target function. Point using the annealed approximation introduced by Derrida and

Figure 6: Example of a double column page with heavy maths, set on a grid.

5 Availability

We intend to release these macros in a free and open format. Our intention is to include them in a style file.

6 Acknowledgments

Han The Thanh did a lot of preliminary work in determining which route we should take. CV Rajagopal helped in writing the macros. Jagath

and Rishi did the refinements and testing.

- ▷ Kaveh Bazargan
River Valley Technologies
www.river-valley.com
- ▷ CV Radhakrishnan
River Valley Technologies
www.river-valley.com