Making MathJax Formulas Auto-Scale with Window Size

Su Jianlin

October 15, 2024

With the emergence and popularity of MathJax, displaying mathematical formulas on the web has gradually found a standard solution. However, MathJax (including its competitor KaTeX) is only responsible for converting LaTeX code on a webpage into mathematical formulas; it still lacks a robust method for adaptive resolution. Some mathematical articles on this site, for instance, are typeset on a PC, so they look acceptable on a desktop, but they can become quite difficult to read when viewed on a mobile phone.

After some testing, I have developed a solution that allows MathJax formulas to adaptively scale with the window size, much like images do, thereby ensuring a better display effect on mobile devices. I would like to share this with everyone.

1 Background and Idea

The origin of this problem is that even when typesetting on a PC, one occasionally encounters single-line formulas whose length exceeds the width of the webpage but are difficult to break into multiple lines. In such cases, a common solution is to manually adjust the font size of the formula using HTML code, for example:

```
<span style="font-size:90%">
    \begin{equation}A very long mathematical formula\end{equation}
</span>
```

This adjusts the size of the mathematical formula to 90% of its original size, which solves most problems. However, manual adjustment is ultimately tedious; the "90%" figure requires several manual trials to find the optimal value, and it can only adapt to a single width. A few days ago, I suddenly realized: why can't mathematical formulas have a `max-width` property to scale automatically, just like images? For example, consider the following image code:

```
<img style="width:400px; max-width:100%" src="https://example.com/test.jpg">
```

The effect achieved is that the image size stays as close to 400px as possible without exceeding the width of its parent element. After some testing, I found that MathJax formulas are text blocks, and there is no built-in method to implement a `max-width` scaling feature similar to images. However, we can use JavaScript to calculate the ratio by which a formula exceeds the width of its parent element and then automatically set the `font-size` of the formula to achieve the same effect.

2 Code Reference

First, here is the final reference solution. Replace your original MathJax configuration code with the following:

```
<script type="text/x-mathjax-config">
    MathJax.Hub.Config({
        tex2jax: {inlineMath: [['$','$'], ['\\(','\\)']]},
        TeX: {equationNumbers: {autoNumber: ["AMS"], useLabelIds: true}, extensions:
            ["AMSmath.js", "AMSsymbols.js", "extpfeil.js"]},
        "HTML-CSS": {noReflows: false, availableFonts: ["tex"], styles:
            {".MathJax_Display": {margin: "1em 0em 0.7em;", display:
            "inline-block!important;"}}},
        "CommonHTML": {noReflows: false, availableFonts: ["tex"], styles:
            {".MJXc-display": {margin: "1em 0em 0.7em;", display:
            "inline-block!important;"}}},
        "SVG": {noReflows: false, availableFonts: ["tex"], styles:
            {".MathJax_SVG_Display": {margin: "1em 0em 0.7em;", display:
            "inline-block!important;"}}}
    });
    MathJax.Hub.Queue(function() {
        document.querySelectorAll('span[id^="MathJax-Element"]').forEach(function(e) {
            parentWidth = e.parentNode.offsetWidth;
            if (e.parentNode.className.endsWith('isplay')) {
                parentWidth = e.parentNode.parentNode.offsetWidth;
            }
            if (e.offsetWidth > parentWidth) {
                e.style.fontSize = parentWidth * 100 / e.offsetWidth + '%';
            }
        });
    });
</script>
```

Compared to the configuration code mentioned in "Making MathJax Better Compatible with Google Translate and Lazy Loading", there are two key changes. First, the configuration item `{linebreaks: {automatic: true, width: "95% container"}}` has been removed. The purpose of this item was to enable automatic line breaks to improve adaptive resolution, but in practice, it isn't very useful. Many carefully written formulas look worse if they are broken automatically. Therefore, removing automatic line breaks ensures the consistency of the formulas across different widths.

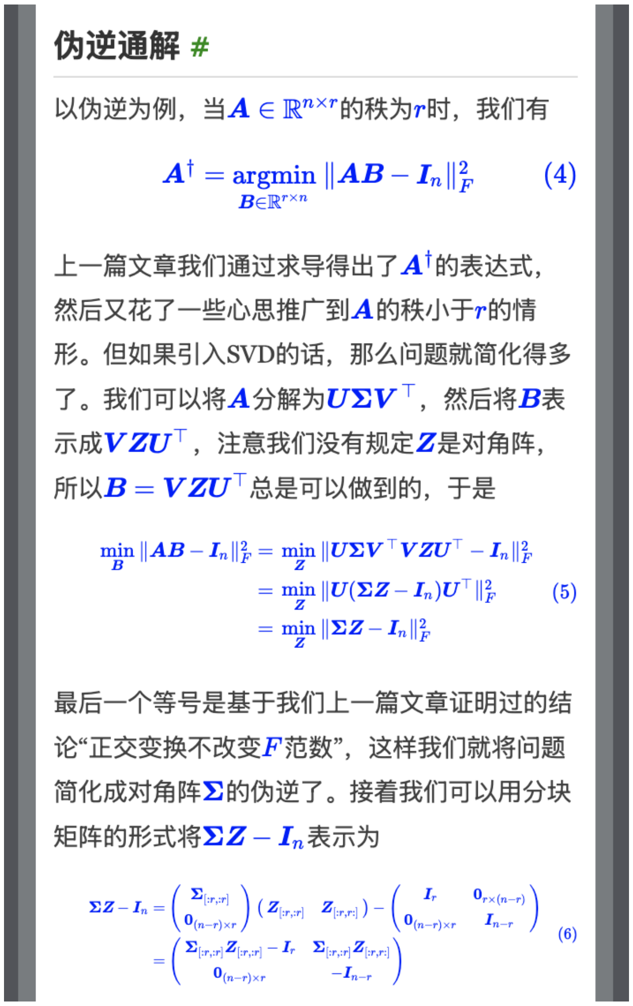
The next critical modification is the `MathJax.Hub.Queue` section. This is a function executed after the formulas have finished rendering. It first finds all `span` elements whose IDs start with "MathJax-Element" (which represent all the mathematical formulas). It then retrieves the formula's width via `e.offsetWidth` and the parent's width via `e.parentNode.offsetWidth` or `e.parentNode.parentNode.offsetWidth`. Based on these values, it calculates and sets the percentage for the `font-size` reduction.

When should `e.parentNode.offsetWidth` or `e.parentNode.parentNode.offsetWidth` be used? Mathematical formulas are divided into "inline formulas" and "display formulas." Inline formulas use the former, while display formulas use the latter. In the code structure, display formulas are nested within an extra `div` layer, so the grandparent node is the actual container. Since the class of this `div` ends with "Display" or "display," I added the condition `e.parentNode.className.endsWith('isplay')`.

These are the principles behind the reference code.

3 Summary

This article shares a method to make MathJax formulas scale automatically with the window size, which helps accommodate the narrow-screen browsing requirements of mobile devices. After this adjustment, even on small screens, mathematical formulas can be displayed exactly as they appear on a PC—though they might be a bit small for the eyes, it serves as a good emergency solution.



伪逆通解 #

以伪逆为例，当$\boldsymbol{A}\in\mathbb{R}^{n\times r}$的秩为$r$时，我们有

$$\boldsymbol{A}^{\dagger} = \mathop{\text{argmin}}_{\boldsymbol{B}\in\mathbb{R}^{r\times n}} \Vert \boldsymbol{A}\boldsymbol{B} - \boldsymbol{I}_n\Vert_F^2 \tag{4}$$

上一篇文章我们通过求导得出了$\boldsymbol{A}^{\dagger}$的表达式，然后又花了一些心思推广到$\boldsymbol{A}$的秩小于r的情形。但如果引入SVD的话，那么问题就简化得多了。我们可以将$\boldsymbol{A}$分解为$\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^{\top}$，然后将$\boldsymbol{B}$表示成$\boldsymbol{V}\boldsymbol{Z}\boldsymbol{U}^{\top}$，注意我们没有规定$\boldsymbol{Z}$是对角阵，所以$\boldsymbol{B}=\boldsymbol{V}\boldsymbol{Z}\boldsymbol{U}^{\top}$总是可以做到的，于是

$$\begin{aligned}\min_{\boldsymbol{B}} \Vert \boldsymbol{A}\boldsymbol{B} - \boldsymbol{I}_n\Vert_F^2 =&\, \min_{\boldsymbol{Z}} \Vert \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^{\top}\boldsymbol{V}\boldsymbol{Z}\boldsymbol{U}^{\top} - \boldsymbol{I}_n\Vert_F^2 \\ =&\, \min_{\boldsymbol{Z}} \Vert \boldsymbol{U}(\boldsymbol{\Sigma}\boldsymbol{Z} - \boldsymbol{I}_n)\boldsymbol{U}^{\top}\Vert_F^2 \\ =&\, \min_{\boldsymbol{Z}} \Vert \boldsymbol{\Sigma}\boldsymbol{Z} - \boldsymbol{I}_n\Vert_F^2\end{aligned} \tag{5}$$

最后一个等号是基于我们上一篇文章证明过的结论“正交变换不改变F范数”，这样我们就将问题简化成对角阵$\boldsymbol{\Sigma}$的伪逆了。接着我们可以用分块矩阵的形式将$\boldsymbol{\Sigma}\boldsymbol{Z} - \boldsymbol{I}_n$表示为

$$\begin{aligned}\boldsymbol{\Sigma}\boldsymbol{Z} - \boldsymbol{I}_n =&\, \begin{pmatrix}\boldsymbol{\Sigma}_{[:r,:r]} \\ \boldsymbol{0}_{(n-r)\times r}\end{pmatrix} \begin{pmatrix}\boldsymbol{Z}_{[:r,:r]} & \boldsymbol{Z}_{[:r,r:]}\end{pmatrix} - \begin{pmatrix}\boldsymbol{I}_r & \boldsymbol{0}_{r\times(n-r)} \\ \boldsymbol{0}_{(n-r)\times r} & \boldsymbol{I}_{n-r}\end{pmatrix} \\ =&\, \begin{pmatrix}\boldsymbol{\Sigma}_{[:r,:r]}\boldsymbol{Z}_{[:r,:r]} - \boldsymbol{I}_r & \boldsymbol{\Sigma}_{[:r,:r]}\boldsymbol{Z}_{[:r,r:]} \\ \boldsymbol{0}_{(n-r)\times r} & -\boldsymbol{I}_{n-r}\end{pmatrix}\end{aligned} \tag{6}$$

Figure 1: Narrow screen display effect after modification

When reposting, please include the original address of this article:
https://kexue.fm/archives/10474

For more detailed reposting matters, please refer to:
“Scientific Space FAQ”