Vol.2023-CE-168 No.9
2023/2/11

グラフィカルなカスタム・コンポーネントを用いたHTMLによる記号と図の関連づけ

佐藤 信[1]

概要：本稿では，HTML 要素を関連づけるための既提案手法により，記号と図のグラフィカルな対応づけをおこなう．SVG (Scalable Vector Graphics) 形式で表現した形状データを Associated Components に格納し，コンポーネントの動作を関連づける．HTML および CSS の文法を使用してコンポーネントの関連づけを Associated Components のカスタム属性に記述できるので，Web ページの内容に合わせたカスタマイズが容易である．円周の公式，および，自動微分を用いた線形回帰について，数式と説明図の関連づけのカスタマイズをおこなう．math augmentation による可読性の向上は，教材作成および Web ページによるオンラインでのプレゼンテーションなどに有効である．

Associating Symbols with Diagrams in HTML Using Customized Graphical Components

Makoto Satoh[1]

Abstract: This paper presents a method for associating symbols with diagrams using a proposed method for associating HTML elements. Shape data represented in SVG (Scalable Vector Graphics) format are stored in Associated Components, and then the components behaviors are associated. It is easy to customize for specific Web page contents because the association of the component behaviors are able to defined using the custom attributes of Associated Components in HTML and CSS syntax. The association of the mathematical equations and the illustration diagrams of the formula of circumference and linear regression with automatic differentiation are customized. The improvement of readability with math augmentation is effective for developing educational materials and online presentation.

1. はじめに

本稿では，HTML 要素を関連づけるための既提案手法により記号と図の関連づけをおこない，そのカスタマイズの容易性を示す[*1]．既提案手法の特徴を，次に示す．

- 数学的な内容の説明において数学記号と説明図の関連を明確にするために，グラフィカル・コンポーネントによりマイクロインタラクションをおこなう．
- Associated Components [6], [7], [8] を機能拡張することにより，カスタム SVG 要素により表現した数学記号と説明図を構成要素の動作を用いて関連づける．HTML 標準規格の文法により作成した Web ページにおいて使用可能である．

Associated Components では，HTML 要素の関連づけを HTML および CSS 標準規格の文法によりカスタム属性に記述できるので，Web ページの内容に合わせたカスタマイズが容易である．ここでは，円周の公式，および，自動微分による線形回帰について，数式と説明図の関連づけのカスタマイズをおこなう．math augmentation による可読性の向上は，教材作成およびオンラインでのプレゼンテーションなどに有効である．

[1] 岩手大学
Iwate University, Ueda, Iwate 020–8551, Japan

[*1] 提案手法の実装を公開している．
https://blue0.an.cis.iwate-u.ac.jp/AssociatedComponents/

これ以降の構成について説明する．2 節では，関連研究について述べる．記号と図の動的な関連づけ手法について 3 節において説明する．4 節では実験と検討をおこなう．最後の 5 節において本稿のまとめと今後について述べる．

2. 関連研究

2.1 数学的記述での math augmentation

コンピュータ・システムおよび応用ソフトウエアの作成において深層学習などの機械学習アルゴリズムを使用する機会が増加している．そのアルゴリズムの説明では数学的記述を用いることが多く，記述内容を分かり易くするために math augmentation をおこなうこともある．[3] では，グラフィックスのレンダリングのために使用される方程式について，数式，図および説明文の対応部分に同一色を用いることにより math augmentation をおこなう例などが示されている．図 1 は，深層学習などでよく用いられる，計算グラフを用いた自動微分の説明の一部分である．数学記号と説明文の対応部分に同一色を用いて説明を分かり易くしている．

多くの分野で研究されている情報可視化のための手法 [5] は，視覚的表現により概念的内容の理解を容易にしようとする点において，math augmentation と同様の目的をもつといえる．例えば，コンピュータ・アルゴリズムは概念的なものであるが，可視化によりそのメンタルモデルの構築が容易になることから，アルゴリズムの可視化のための多くの手法が存在する [2], [11] ．そして，深層学習の分野においても可視化のための研究が多数おこなわれている [4] ．深層学習を含む機械学習で用いられる確率的勾配降下法を分かり易く説明するために，math augmentation を有効に用いている例もある．

本稿では，数式に含まれる数学記号とその説明図を関連づけることにより math augmentation をおこなう．数学的内容の対応関係を，動的なグラフィックスを用いて明確に表現可能である．また，対話的操作に対応させた複数の異なる対応関係を提示可能である．これらの関連づけを HTML プログラムにより定義できるので，カスタマイズが容易である．

2.2 エンドユーザ・プログラミング

コンピュータおよびインターネットなどの普及により，ソフトウエアの専門家だけではなく一般の方々がアプリケーション・ソフトウエアなどのソフトウエアを使用する機会が増加している．ソフトウエアのユーザがそれぞれの目的にあわせておこなうソフトウエアのカスタマイズおよび簡単なスクリプトの作成などは，小規模なプログラミングであることからエンドユーザ・プログラミングと呼ばれる．

…

計算グラフを使用した自動微分について，次の計算をおこなう

1 入力 1 出力の計算ネットワークを例に説明する．

$$y = w\,x + b$$

x が入力，y が出力，w が重み，そして b がバイアスである．

計算グラフの構築では，上式を次のように分解する．

$$\alpha = w\,x$$

$$y = \alpha + b$$

これを計算グラフにより表現する場合には，例えば，演算をノードに対応づけ，ノード間をエッジで結合する．ノードでの演算結果はエッジを伝搬する．グラフの入力または学習パラメータなどを値を格納したノードとして表現する場合もある．

…

図 1: math augmentation の例
Fig. 1 Example of math augmentation

エンドユーザ・プログラミングの状況を調査した研究 [1] では，多くのエンドユーザ・プログラミングにおいてソフトウエア・コンポーネントを使用していることが報告されている．機能の明確な小さなコンポーネントの集合としてソフトウエアを設計することによりその構造の理解を容易にすることは，本格的なソフトウエア開発においてはもちろん重要であるが，エンドユーザ・プログラミングにおいてもコンポーネントの概念は利点が多いといえる．特に，ソフトウエアのカスタマイズにおいては，カスタマイズに関連した少数のコンポーネントのみに着目すればよいという利点がある．

本稿で用いる Associated Components は，HTML プログラムにおいて HTML 要素の関連づけを記述可能なカスタム・コンポーネントである．このコンポーネントにより可能なエンドユーザ・プログラミングを，次に示す．

レベル 1 Associated Components のクラスを継承した新たなカスタム・コンポーネントを作成する．
レベル 2 Associated Components を用いて HTML 要素の動作を関連づけた HTML プログラムを作成する．
レベル 3 **レベル 2** のプログラミングにより作成した HTML プログラムのカスタマイズをおこなう．例えば，目的にあわせて HTML 要素の関連を変更する．

これ以降では，**レベル 3** のエンドユーザ・プログラミングをおこなう．Associated Components により表現した記号および図の対応する部分を動的にハイライトするマイクロインタラクション [10] のカスタマイズをおこなう．

表 1: 追加のカスタム・イベント・ハンドラ
Table 1 Additional custom event handlers.

event handlers (abbreviations)	description
setReceiverSVGStyle (setRSVGStyle)	Set SVG styles to receivers.

3. HTML による記号と図の関連づけ

3.1 グラフィカル・コンポーネントの定義

本稿では，SVG (Scalable Vector Graphics) 形式[*2] を用いて記号および図を表現する．その形状データを，既提案の clm-associated-svg 要素に格納する．clm-associated-svg 要素は，SVG 形状を表現するための Associated Components である．clm-associated-svg 要素を用いる場合には，HTML プログラムにおいて次の記述をおこなう．

- clm-associated-svg 要素に格納する形状データの定義
- clm-associated-svg 要素の動作の関連づけの定義

これらについて，具体的に説明する．

3.2 形状データの定義

記号および図の形状データを SVG 形式で表現し，それを SVG データ定義ファイルに格納する．SVG データ定義ファイルは，HTML プログラムに指定する (リスト 1)．clm-associated-svg 要素は，SVG データ定義ファイルにある形状データを要素に格納する．SVG データ定義ファイルには，標準規格に含まれる svg 要素および path 要素の属性を指定できる．これにより，1 つの形状が複数の clm-associated-svg 要素により構成される場合に，形状の詳細な位置決め，および，形状の一部分のみを動的に変化させることなどが可能となる．svg 要素の style 属性を指定する例をリスト 2 に示す．clm-associated-svg 要素の ID を指定し，属性名と設定値を記述する．属性 d には，形状を表現する SVG パスを指定する．なお，clm-associated-svg 要素は HTMLElement を継承したカスタムコンポーネントなので，標準 HTML 要素と同様に ID 属性などの標準の属性を指定できる．

3.3 関連づけの定義

clm-associated-svg 要素では，関連づけのための機能を実現するために，それ以外の Associated Components と同じクラスを Mix-in するように設計されている．そのため，clm-associated-svg 要素は，それ以外の Associated Components と同じカスタム属性およびカスタム・イベント・ハンドラを用いて，同じ方法により要素の関連づけを定義できる．そして，clm-associated-svg 要素では，svg 要素の style 属性を動的に変更するための新しいカスタム・イベント・ハンドラが追加されている (表 1)．

なお，HTML 標準規格に定義される標準 HTML 要素，clm-associated-svg 要素およびそれ以外の Associated Components を，同じ HTML プログラムに混在可能である．clm-associated-svg 要素とそれ以外の Associated Components の関連づけも可能である．

リスト 1: SGV データ定義ファイル
Listing 1: SVG data definition file.

```
<html>
  <head>
    ...
    <script
      src="./CircumferenceOfCircle-Type1Data.js"
    ></script>
    ...
  </head>
  <body> ... </body>
</html>
```

リスト 2: SVG データの定義
Listing 2: Defining SVG data.

```
...
[
  {
    id : 'C',
    attributes :
    [
      { name: 'transform', value: ... },
      { name:'d', value: ... },
      { name:'stroke', value:' ... ' },
      { name:'stroke-opacity', value:'1.0' },
      { name:'stroke-width', value:'0.1' },
      { name:'stroke-linecap', value:'round' },
      { name:'stroke-linejoin', value:'round' },
      { name:'fill', value:' ... ' },
      { name:'fill-opacity', value:'1.0' },
    ]
  },
  ...
]
```

[*2]SVG 形式は，ベクトル・グラフィックスを表現するための形式である．線分および Bezier 曲線などを用いて形状を表現するので，自由な形状表現が可能であり，表示を拡大・縮小しても品質が劣化しない．HTML プログラムでは，標準規格に定義される SVG 要素を使用可能である．本稿では，カスタム SVG 要素を用いる．

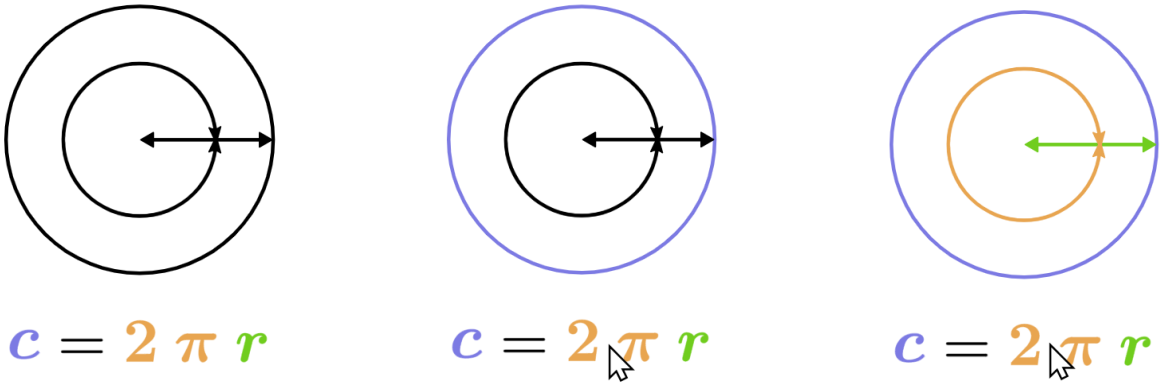

図 2: 半径を用いた円周の計算

Fig. 2 Calculating the circumference of a circle with the radius: initial state (left), cursor over (center), and click (right). Double clicking on the mathematical symbols resets the diagram to the initial state.

4. 実験と結果の検討

4.1 実装

Associated Components(clm-associated-svg 要素を含む) の実装には，JavaScript を使用した．実験に用いた HTML プログラムの作成には，HTML および CSS を使用した．なお，自動微分による線形回帰の実験では，サーバ・クライアント方式のプログラムを作成した．サーバで自動微分による線形回帰をおこない，クライアントに結果をグラフ表示した．サーバの実装には python，計算グラフによる自動微分などの数値計算には PyTorch [9] を用いた．

4.2 円周の計算

図 2 は，半径を用いた円周の公式とその説明図である．図 2 (左) は初期状態，図 2 (中央) は公式が表示されている領域にカーソルを移動することにより説明図の円周がハイライトされた状態，そして，図 2 (右) は公式のクリックにより説明図の角度 2π と半径がハイライトされた状態である．数式および説明図には，複数の clm-associated-svg 要素を用いている．各数学記号を表現する複数の要素が共通のイベント受信領域を使用するようにしている．

図 2 で用いた SVG データの定義は，3.2 節のリスト 2 である．HTML プログラムでの clm-associated-svg 要素の記述をリスト 3 に示す．id が `CircleShape` の要素は図 2 の円 (円周) であり，そのグループ id を`/Circle/`とした．id が `radiusArrowLine` などの半径の矢印を構成する要素は，`/Circle/`，`/Radius/`の 2 グループに所属する．リストにはないが，角度を示す矢印は`/Circle/`，`/Angle/`に所属する．id が `C` の要素は方程式の左辺である．`onmouseover` 属性では，この要素にカーソルオーバーする

リスト 3: 図 2 の clm-associated-svg 要素の関連づけ

Listing 3: Associating clm-associated-svg elements in **Fig. 2**.

```
...
<clm-associated-svg
  id = 'CircleShape'
  data-gId = '/Circle/' ...
></clm-associated-svg>
<clm-associated-svg
  id = 'radiusArrowLine'
  data-gId = '/Circle/ /Radius/' ...
></clm-associated-svg>
<clm-associated-svg
  id = 'radiusRightArrow'
  data-gId = '/Circle/ /Radius/' ...
></clm-associated-svg>
...
<clm-associated-svg
  id = "C"
  onmouseover = "setRId( 'CircleShape' );
    setRSVGStyle( 'stroke: ... ;' )"
  onclick = "setRId( '/Angle/' );
    setRSVGStyle( 'stroke: ... ; fill: ... ;' );
    setRId( '/Radius/' );
    setRSVGStyle( 'stroke: ... ; fill: ... ;' );"
  ondblclick = "setRId( '/Circle/' );
    setRSVGStyle( 'stroke: ... ; fill: ... ;' )"
  ...
></clm-associated-svg>
...
```

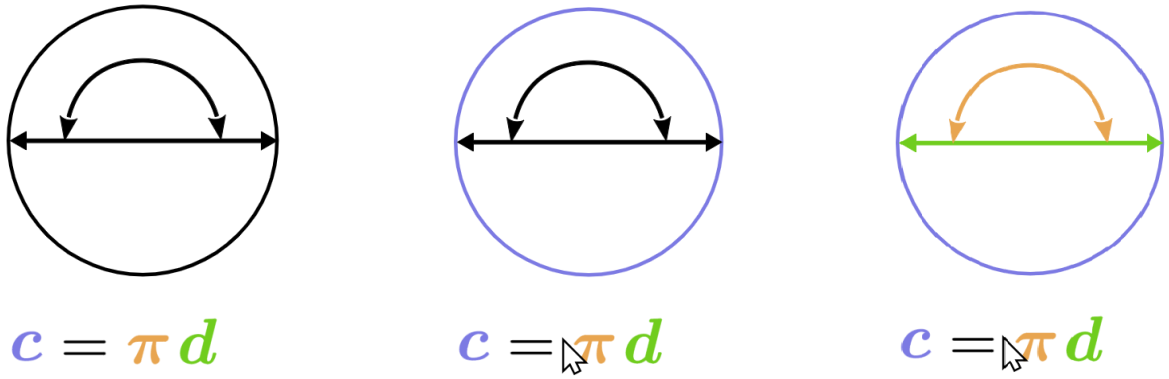

図 3: 直径を用いた円周の計算

Fig. 3 Calculating the circumference of a circle with the diameter: initial state (left), cursor over (center), and click (right). Double clicking on the mathematical symbols resets the diagram to the initial state.

と id が CircleShape の要素の stroke を青色にするように設定した．onclick 属性では，この要素のクリックによりグループ id が/Angle/の要素の stroke，fill を橙色に，そして，グループ id が/Radius/の要素の stroke，fill を緑色にするように設定した．ondblclick 属性では，この要素のダブルクリックによりグループ id が/Circle/の要素の stroke，fill を黒色にするように設定した．以上の要素の関連づけの設定により，図 2 に示すグラフィカルなマイクロインタラクションを定義できる．

図 3 は，直径を用いた円周の公式とその説明図である．ここでの数式と説明図の動作の関連づけは，図 2 と同様である．図 2 で用いたプログラムは，図 2 でのプログラムを次のようにカスタマイズすることにより作成した．

- clm-associated-svg 要素の定義の追加 (直径，角度 π，数式の右辺にある d の形状を格納)
- clm-associated-svg 要素の追加に対応した関連づけの追加および変更

カスタマイズ後の HTML プログラムを，リスト 4 に示す．図 2 では半径の形状 (直線および矢印) を 3 つの clm-associated-svg 要素で表現したが，図 3 では直径の形状 (直線および矢印) を 1 つの clm-associated-svg 要素で表現した．id が SymbolD の要素は数式の右辺の d である．この要素にカーソルを移動 (onmouseover) すると円周 (CircleShape) をハイライト，クリック (onclick) すると角度 π(/Angle/) および直径 (/Diameter/) の形状をハイライト，そして，ダブルクリック (ondblclick) すると全形状 (/Circle/) を初期状態にリセットするように定義している．なお，追加した形状の SVG データは，形状データ定義ファイル (3.2 節を参照) に追加している．

リスト 4: 図 3 の clm-associated-svg 要素の関連づけ

Listing 4: Associating clm-associated-svg elements in **Fig. 3**.

```
...
<clm-associated-svg
  id = 'CircleShape'
  data-gId = '/Circle/' ...
></clm-associated-svg>
<clm-associated-svg
  id = 'diameterArrow'
  data-gId = '/Circle/ /Diameter/' ...
></clm-associated-svg>
<clm-associated-svg
  id = 'anglePi'
  data-gId = '/Circle/ /Angle/' ...
></clm-associated-svg>
...
<clm-associated-svg
  id = "SymbolD"
  onmouseover = "setRId( 'CircleShape' );
    setRSVGStyle( 'stroke: ... ;' )"
  onclick = "setRId( '/Angle/' );
    setRSVGStyle( 'stroke: ... ; fill: ... ;' );
    setRId( '/Diameter/' );
    setRSVGStyle( 'stroke: ... ; fill: ... ;' );"
  ondblclick = "setRId( '/Circle/' );
    setRSVGStyle( 'stroke: ... ; fill: ... ;' )"
  ...
></clm-associated-svg>
...
```

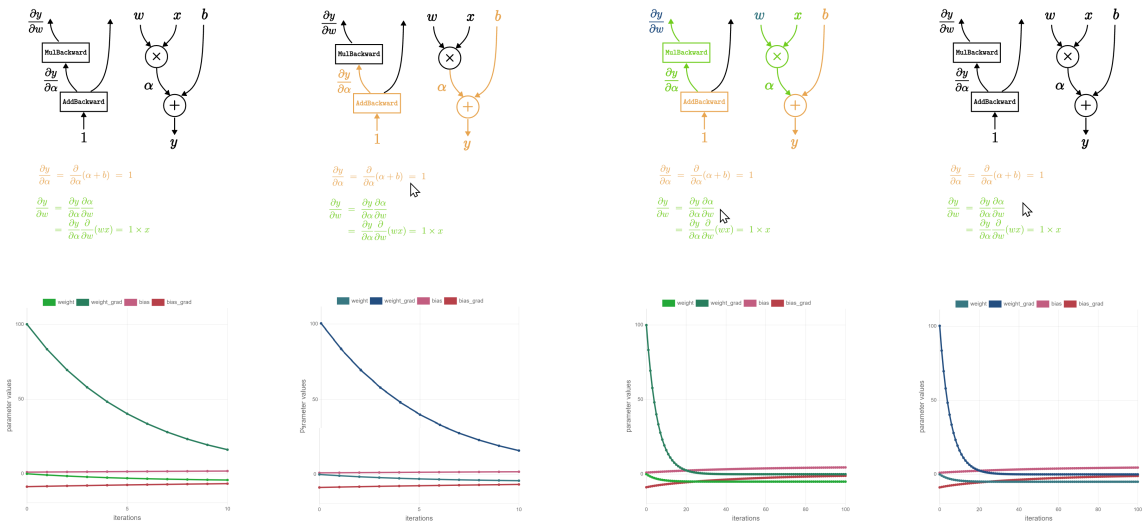

図 4: 自動微分による線形回帰

Fig. 4 Linear regression with automatic differentiation (the backward path computations using a computation graph): the initial state, click on the upper equation, click on the lower equation, and double click (from left to right in the upper row). Iterative computation of the computation graph parameters are shown in the lower row: total iteration 11 and 101 (left half and right half respectively), $\frac{\partial y}{\partial w}$ and w are highlighted in blues (the right of the each part).

リスト 5: 図 4 での clm-associated-svg 要素の関連づけ

Listing 5: Associating clm-associated-svg elements in **Fig. 4**.

```
...
<clm-associated-svg
  id = 'DYDWSymbol'
  data-gId = '/Diagram/ /DYDW/' ...
></clm-associated-svg>
<clm-associated-svg
  id = 'DYDAlphaSymbol'
  data-gId = '/Diagram/ /DYDAlpha/ /DYDW/' ...
></clm-associated-svg>
...
<clm-associated-svg
  id = "DYDAlpha-OnePath"
  onclick = "setRId( '/DYDAlpha/' );
    setRSVGStyle( 'stroke: ... ; fill: ... ;' );"
  ondblclick = "setRId( '/Diagram/' );
    setRSVGStyle( 'stroke: ... ; fill: ... ;' ); ..."
  ...
></clm-associated-svg>
<clm-associated-svg
  id = "DYDW-OnePath"
  onclick = "setRId( '/DYDW/' );
    setRSVGStyle( 'stroke: ... ; fill: ... ;' ); ..."
  ...
></clm-associated-svg>
...
```

4.3 自動微分による線形回帰

計算グラフを用いて線形回帰のパラメータを反復的に求める過程の説明を例として，記号と図の関連づけをおこなった[*3, *4]．データ集合 (x_i, y_i) が与えられたとして，次式によりパラメータ w および b を求めた (2.1 節の図 1).

$$y = wx + b \tag{1}$$

図 4 は，自動微分の計算式，計算グラフ，および，線形回帰のパラメータを求める過程のグラフである．図の上側は左から右に，初期状態，上式をクリックした状態，下式をクリックした状態，そして，ダブルクリックにより初期状態にリセットした状態である．式のクリックにより計算グラフの対応する部分がハイライトされている．図の下側は，パラメータを求める過程のグラフである．図の左右半分はそれぞれ繰り返し数 11, 101 であり，各領域の右側のグラフでは $\frac{\partial y}{\partial w}$ および w がハイライトされている．数式および計算グラフは clm-associated-svg 要素により表現している．パラメータを求める過程のグラフは，JavaScript を用いて作成している．数式と計算グラフは，Associated Components の機能により関連づけをおこない，それらとパラメータを求める過程のグラフはプロキシとしての Associated Components を用いて関連づけている．

リスト 5 は，図 4 での HTML 要素の動作の関連づけの定義である．

*3 深層学習などの機械学習では，計算グラフにより自動微分などの計算をおこなうことが多い．

*4 ここでの関連づけでは，損失の計算およびパラメータの更新についての説明は含んでいない．

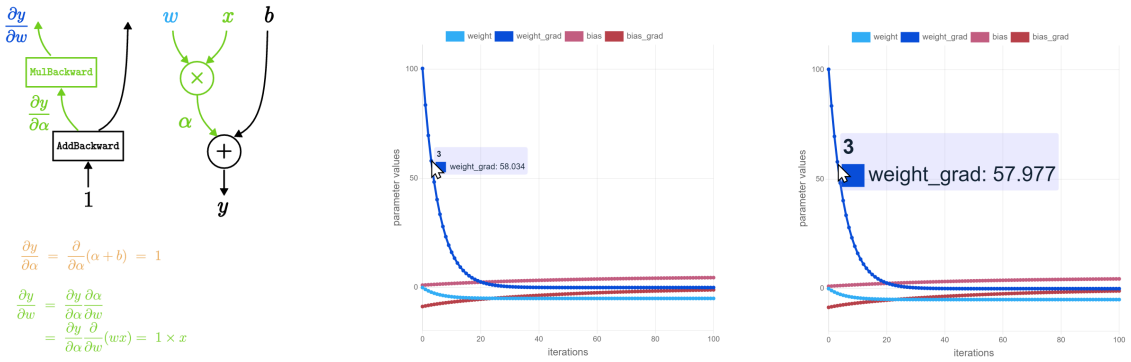

図 5: JavaScript プログラムと Associated Components の関連づけ
Fig. 5 Associating a JavaScript program with Associated Components: highlight in the light blue colors and the large font size for the tool tip.

リスト 6: ハイライトに用いる色のカスタマイズ (図 5)
Listing 6: Customizing highlight colors (**Fig. 5**).

```
...
<div
  id = 'weight'
  is = 'clm-associated-div'
  style = 'stroke: ...;'
></div>
<div
  id = 'weightGradient'
  is = 'clm-associated-div'
  style = 'stroke: ...'
></div>
...
```

リスト 7: ツールチップのカスタマイズ (図 5(中央，右))
Listing 7: costomizing tooltips (**Fig. 5**).

```
...
<div
  id = 'tooltipTitleFontSize'
  is = 'clm-associated-div'
  style = 'font-size: 50px;'
></div>
<div
  id = 'tooltipBodyFontSize'
  is = 'clm-associated-div'
  style = 'font-size: 50px;'
></div>
...
```

図 5 では，JavaScript プログラムと Associated Components を関連づけた．リスト 6 に示すように，HTML プログラムにおいてハイライトに用いる色の値を格納した div 要素としての Associated Components を定義した．それらの値を Associated Components のカスタム・イベント・ハンドラで変更することにより動的な関連づけ，および，カスタマイズを可能にした．JavaScript プログラムでは，これらの要素を参照することにより表示色を決定しグラフを作成した．また，リスト 7 のように，div 要素としての Associated Components にグラフで表示するツールチップのフォントサイズを格納し，JavaScript プログラムでそれらの値を参照することによりグラフを作成した．図 5 では，これらの div 要素の値を変更することによりカスタマイズをおこなった．

図 4, 5 の計算グラフでは，バックワードパスの右側にフォワードパスを表示している．バックワードパスのノードの `AddBackward` および `MulBackward` は，それぞれ加算および乗算のバックワードパスでの演算名である[*5]．`AddBackward` の入力である 1 は，$\frac{\partial y}{\partial y}$ の値である．なお，図ではパラメータ w の勾配を求める計算についての説明のみを示している．

4.4 検討

4.2 節では，円周の公式を例として数式と説明図を関連づけた．4.3 節では，自動微分による線形回帰について，パラメータの反復計算のための数式，説明図およびグラフの関連づけをおこなった．HTML プログラムに記述した関連づけのとおりに，図の対応部分を動的にハイライトできることを確認した[*6]．数式および図 (計算グラフ) は clm-associated-svg 要素により表現し，パラメータの更新過程のグラフは JavaScript を用いて画像として動的に生成した．Associated Components の機能を用いると，これらの関連づけを HTML プログラムのみにより記述可能であり，そして，その部分の変更のみにより関連づけのカスタ

[*5] この名前は PyTorch [9] のものである．なお，本稿の目的はグラフィカル・コンポーネントによる対応づけであるので，`AcumulateGrad` は省略した．

[*6] 動作の確認には Web ブラウザを用いた．関連づけの動作だけでなく，ブラウザの表示を拡大縮小しても品質が劣化しないことも確認できた．なお，カーソルを含んだ図を掲載するために，この原稿ではブラウザへの表示を画像として保存したものを用いている．

マイズをおこなえることを確認した．

図 2 – 5 での数式および説明図の形状には clm-associated-svg 要素を用いた．ある形状の各部分を clm-associated-svg 要素で表現する場合には，表現にあわせて形状を自由に分解できる．例えば，図 2 では，各記号に 1 つの clm-associated-svg 要素を用いてそれらを組み合わせて公式を表現した．図 3 では，必要な clm-associated-svg 要素を追加した．一方，図 4 では，偏微分の数式を，2 つの clm-associated-svg 要素で表現している．SVG グラフィックスを用いているので，複雑な形状を自由に表現可能である．clm-associated-svg 要素を用いたアプリケーションの範囲は広いといえる．

Associated Components の特徴は，既存のソフトウエアの機能は利用可能な状態で，マイクロインタラクションのためのグラフィカルな関連づけの定義をコンパクトに HTML プログラムだけで記述可能なことである．機械学習アルゴリズムなどの説明に用いられる Jupyter では機能の追加により HTML および JavaScript などを使用可能であるので，Jupyter などに Associated Components を対応させることは興味深い課題である．

5. おわりに

HTML 要素を関連づけるための既提案手法により記号と図の関連づけをおこない，そのカスタマイズの容易性を示した．具体的には，数学公式およびその説明図などの複雑な形状を Associated Components を機能拡張した clm-associated-svg 要素で表現し，それらのグラフィカルな関連づけが可能であることを示した．そして，関連づけのカスタマイズを，HTML プログラムの局所的な変更によりおこなえることを示した．数学的な内容の記述において，数式に含まれる記号，説明図，および，グラフなどの対応づけを明確にするための math augmentation をおこなうことにより，概念的な内容の把握が容易になる．教材作成およびオンラインでのプレゼンテーションなどに有効な手法であるといえる．

今後の課題には，clm-associated-svg 要素で用いる形状データの生成手法，および，Jupyter などで Associated Components を用いるための手法に関する研究などがある．

参考文献

[1] Barricelli, B. R., Cassano, F., Fogli, D. and Piccinno, A.: End-user development, end-user programming and end-user software engineering: A systematic mapping study, *Journal of Systems and Software*, Vol. 149, pp. 101 – 137 (online), DOI: https://doi.org/10.1016/j.jss.2018.11.041 (2019).

[2] Grissom, S., McNally, M. F. and Naps, T.: Algorithm Visualization in CS Education: Comparing Levels of Student Engagement, *Proceedings of the 2003 ACM Symposium on Software Visualization*, SoftVis '03, ACM, pp. 87–94 (2003).

[3] Head, A., Xie, A. and Hearst, M. A.: Math Augmentation: How Authors Enhance the Readability of Formulas Using Novel Visual Design Practices, *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*, CHI '22, ACM, (online), DOI: 10.1145/3491102.3501932 (2022).

[4] Hohman, F., Kahng, M., Pienta, R. and Chau, D. H.: Visual Analytics in Deep Learning: An Interrogative Survey for the Next Frontiers, *IEEE Transactions on Visualization and Computer Graphics*, Vol. 25, No. 8, pp. 2674–2693 (2019).

[5] Liu, S., Cui, W., Wu, Y. and Liu, M.: A Survey on Information Visualization: Recent Advances and Challenges, *Vis. Comput.*, Vol. 30, No. 12, pp. 1373–1393 (2014).

[6] 佐藤　信：Web ページ制作入門において利用可能な関連情報を提示するための GUI コンポーネント，情報処理学会研究報告，Vol. 2020-CE-154, No. 13, pp. 1–8 (2020).

[7] 佐藤　信：データ駆動型 Associated Components を用いた Ambient Web Design，情報処理学会研究報告，Vol. 2020-CG-180, No. 8, pp. 1–8 (2020).

[8] 佐藤　信：関連づけられた情報の提示のためにカスタマイズした GUI コンポーネントの設計，情報処理学会研究報告，Vol. 2020-HCI-187, No. 33, pp. 1–8 (2020).

[9] Paszke, A., Gross, S. et al.: PyTorch: An Imperative Style, High-Performance Deep Learning Library, *Advances in Neural Information Processing Systems*, Vol. 32, Curran Associates, Inc. (2019).

[10] Saffer, D.: *Microinteractions: Designing with Details*, O'Reilly Media, Inc. (2013).

[11] Shaffer, C. A., Cooper, M. L., Alon, A. J. D., Akbar, M., Stewart, M., Ponce, S. and Edwards, S. H.: Algorithm Visualization: The State of the Field, *ACM Trans. Comput. Educ.*, Vol. 10, No. 3, pp. 1–22 (2010).