数学記号と図のグラフィカルな関連づけを用いた自動微分による反復的線形回帰の可視化 *

○佐藤 信 (岩手大学理工学部)

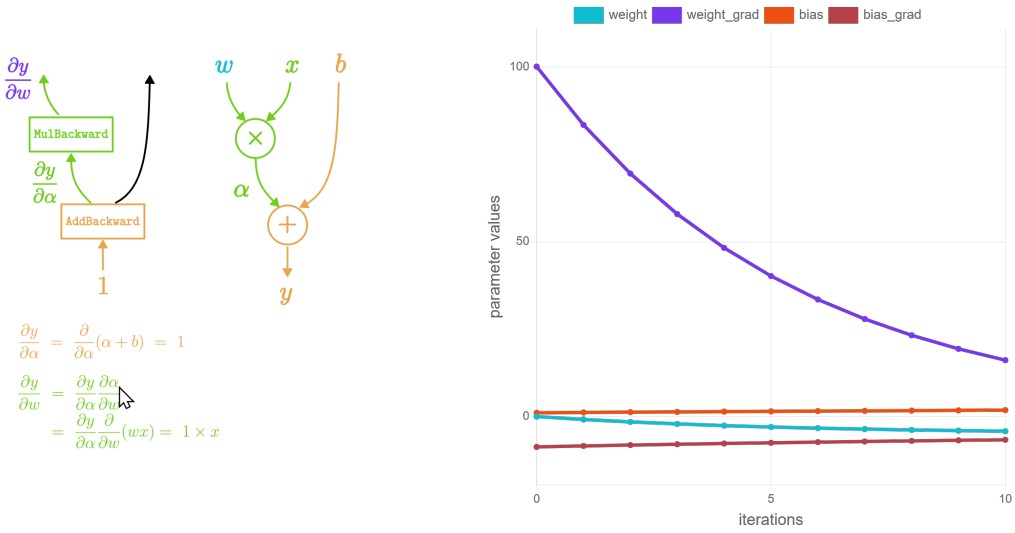

Figure 1: Visualizing iterative linear regression: the computation graphs of automatic differentiation (upper left), the mathematical equations (lower left), and the plots of weight and bias, and their gradients (right). The corresponding parts are highlighted in the same colors (blues).

概 要

本稿では，math augmentation のための既提案手法を用いて，反復計算による線形回帰でのパラメータの計算の可視化をおこなう．学習のための計算において用いられる計算グラフについて，数学記号，計算グラフの説明図，および，計算過程での勾配の変化をグラフィカルに関連づけることにより，自動微分を用いた計算の過程を分かり易く可視化する．

1 はじめに

本稿では，math augmentation のための既提案手法[1] [1] を用いて，反復的線形回帰のパラメータの学習過程を可視化する．既提案手法の特徴を，次に示す．

- 数学的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 [2], [3], [4] の機能を拡張することにより，数学記号と説明図の構成要素を表現するカスタム SVG 要素の動作を関連づける．HTML 標準規格の文法により作成した Web ページにおいて，提案手法を使用可能である．

ここでは，既提案のグラフィカル・コンポーネントを用いることにより，反復的な線形回帰での計算グラフによる自動微分 [5], [6] の仕組みの可視化をおこなう．

これ以降の構成について，簡単に説明する．2 節では，関連研究との比較をおこなう．そして，グラフィカル・コンポーネントを用いた可視化のための関連づけ手法について 3 節において説明する．4 節では，実験結果を示し検討をおこなう．そして最後に，5 節で本稿のまとめと今後について述べる．

*Visualizing Iterative Linear Regression Using Automatic Differentiation with Graphical Association between Mathematical Symbols and Diagrams

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

2 関連研究

2.1 数学的記述での math augmentation

応用ソフトウエアの作成において，高度な数学計算を含むアルゴリズムを用いる機会が増加している．特に，深層学習を含む機械学習において用いられるアルゴリズムの説明では数学的記述が用いられることが多い．そのような記述内容を分かり易くする手法には，math augmentation がある．[7] 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光の方程式について，数式，図および説明文の対応する部分が分かり易いよう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 2 ([1] より引用) 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について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．

本稿の可視化では，パラメータの学習において用いられる計算グラフについて，自動微分で用いる偏微分などの数式，計算グラフの図，および，学習の過程での勾配の変化を示すグラフ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．特徴は，[1] で提案のグラフィカル・コンポーネントを用いることにより，math augmentation をおこなうためのカスタマイズした HTML コンポーネントの動作の関連の対応づけを HTML プログラムのみで記述可能なことである．

2.2 機械学習の可視化

情報を視覚的に表現することは情報の分析および理解を容易する．そのため，可視化に関する研究が多数おこなわれている [8] ．コンピュータ・ソフトウエアの作成において用いられるコンピュータ・アルゴリズムは概念的なものであるが，メンタルモデルの構築および仕組みの理解を容易にするために，アルゴリズムを可視化するための多くの手法が発表されている [9] ．

そして，機械学習の分野においても可視化を目的とした研究がおこなわれている [10] ．特に，深層学習の分野は発展が著しく，深層学習を対象とした可視化手法が多く提案されている [11] [12] ．それらにおいて数学的記述が含まれる場合には，math augmentation が有効である．math augmentation は，数学的表現の対応関係を明確にするための可視化手法であるといえる．深層学習モデルなどの学習で用いられる確率的勾配降下法の動作の説明において math augmentation をおこなった例もある．

...

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．

...

Figure 2: An example of math augmentation.

機械学習を対象とした可視化手法の目的は，次のように分類できる．

- 機械学習モデルの性能などの分析[2]
- 機械学習アルゴリズムなどの手法の仕組みの説明

深層学習の各構成要素については数学的性質が明確であるが，それらを組み合わせた深層ネットワークをデータ集合により学習した学習モデルについては，明らかになってきている事柄も多いが未知の部分も多いといえる．学習モデルの概要を捉える手段として，可視化による分析は有効であるといえる．

本稿の可視化では，グラフィカルな手法を用いることにより，機械学習モデルの学習過程でおこなう計算グラフによる自動微分について説明する．既提案のグラフィカル・コンポーネントを用いて数学記号と説明図の関連づけをおこない，自動微分の仕組みの理解を容易にするための可視化をおこなう．

[2]機械学習モデルの汎化性能を評価するために，学習過程での損失の変化のグラフを用いてオーバーフィッテイングあるいはアンダーフィッティングの可能性を検査することは，機械学習に関する研究・開発の初期からおこなわれている可視化である．オーバーフィッテイングおよびアンダーフィッティングは，モデルの複雑性 (学習性能) と学習に用いたデータの関係により発生することなので，どの機械学習モデルを用いる場合であってもオーバーフィッテイングおよびアンダーフィッティングの検査は必要である．

3 可視化のためのコンポーネントの関連づけ

3.1 関連づけるコンポーネント

反復的線形回帰のパラメータの学習について，計算グラフによる後ろ向き自動微分の可視化をおこなうために，次の構成要素について関連づけをおこなう．

- 自動微分の数式
- 計算ネットワークの計算グラフ
- 自動微分による勾配の変化のグラフ

3.2 Associated Components

HTML 標準規格 [13] に含まれる文法のみを用いて，HTML 要素の動的な関連づけをおこなうための手法として，Associated Components [2], [3], [4] が提案されている．そして，[1] では，数学記号および説明図の形状を SVG グラフィックスを用いて表現した HTML 要素を関連づけるために，カスタム SVG 要素としての Associated Components が提案されている．ボタン要素のための Associated Components のクラス構造を図 3 に示す (詳細は，[2] を参照)．Associated Components の設計では，HTML 標準規格に含まれるカスタム HTML 要素のための機能を用いている．図 3 の HTMLButtonElement クラスは，HTML 標準規格に定義されるクラスであり Web ブラウザに標準実装されている．そのため，HTML プログラムにおいて標準 HTML 要素と Associated Components を混在可能であり，Associated Components 間，および，Associated Components と標準 HTML 要素間での動的な関連づけが可能である．関連づけの定義は，HTML プログラムにおいて Associated Components のカスタム属性にカスタム・イベント・ハンドラを指定することによりおこなう．

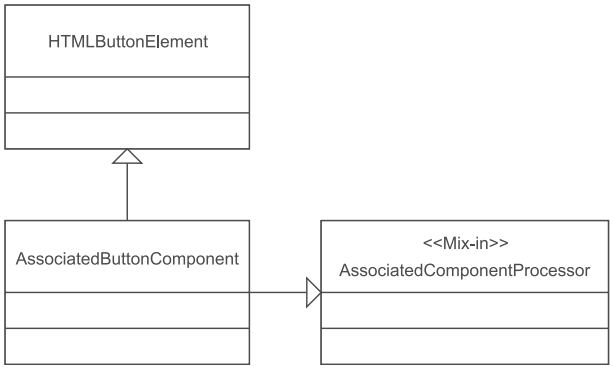

Figure 3: An example of Associated Components class structure.

3.3 機械学習の可視化のための関連づけ

Web ブラウザを用いて機械学習の可視化をおこなうには，次の方式がある．

方式 1 全ての処理を，Web ブラウザのプログラムでおこなう．

方式 2 サーバのサーバ・プログラムと Web ブラウザのクライアント・プログラムにおいて，それぞれが得意な処理を分担しておこなう．

それぞれの方式の特徴は次のとおりである．

方式 1 の特徴 Web ブラウザでのアプリケーション・プログラミングについての知識のみでプログラムを作成できる．プログラムの計算量，メモリ量などが大きい場合には，実行速度が遅くなる．

方式 2 の特徴 サーバおよびクライアントでのアプリケーション・プログラミング，および，通信などの知識が必要である．機械学習のように処理量の大きい計算はサーバでおこない，計算結果のみをクライアントに送信して表示するなどの方式により可視化をおこなうことができる．

本稿で可視化をおこなう反復的に線形回帰のパラメータを求めるプログラムについては計算量が少ないので，Web ブラウザで全ての処理をおこなうという設計であっても特に問題はない．関連研究において，深層学習を含む機械学習のための多くの手法の可視化に提案方式を対応させることを考慮して，**方式 2** により可視化のためのプログラムを設計する．また，機械学習において標準的なフレームワークを使用して可視化をおこなうほうがメリットが大きいと考えた．

次に，数学記号と図の関連づけの方式について述べる．3.1 節でのコンポーネントのうち，数式および計算グラフについては形状を SVG 形式により作成する．その形状をカスタム SVG 要素としての Associated Components に格納することにより，数式と計算グラフの関連づけを可能とする．また，それらと勾配の変化のグラフの関連づけでは，プロキシとしての Associated Components を用いて，Associated Components としての数式および計算グラフを勾配の変化のグラフに間接的に関連づける．

リスト 1: SVG データ定義ファイルの指定

Listing 1: Specifying SVG data definition file.

```
<html>
  <head>
    ...
    <script
      src="./Backpropagation-BackwardPath.js"
    ></script>
    ...
  </head>
  <body> ... </body>
</html>
```

Associated Components では，HTML および CSS の標準規格において定義されている機能については，標準的な記述方法により使用可能であるように設計がおこなわれている．ここでの可視化方式の設計でも，同様に，Associated Components は math augmentation のためのコンポーネントの関連づけのみをおこない，機械学習およびグラフ表示などには既存のフレームワークの機能をそのまま使用可能であるような設計をおこなった．

4 実験結果と検討

4.1 実装

4.1.1 概要

3 節での数式と図のグラフィカルな関連づけのための手法を用いて，反復的に線形回帰のパラメータを求める過程の可視化をおこなった．線形回帰のパラメータを求めるための手法は多く存在するが，自動微分の仕組みの可視化をおこなうために反復的手法を選択した．

実装に使用したプログラミング言語は，HTML, CSS, JavaScript, Python である．線形回帰の計算での計算ネットワークおよび計算グラフの構築，および，自動微分には，機械学習のフレームワークである PyTorch [14] を用いた．

4.1.2 形状データの準備

形状データは，カスタム SVG 要素である clm-associated-svg 要素 (詳細は，[1] を参照) に格納した．clm-associated-svg 要素が使用する形状データを SVG データ定義ファイルに定義し，そのファイル・パスを HTML プログラムの head 部に指定した (リスト 1).

実験で用いた数学記号を，図 4 に示す．記号の形状データ (SVG) を，SVG データ定義ファイルにパス属性

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

$$y = \alpha + b$$

$$\alpha = w\,x$$

$$\frac{\partial y}{\partial \alpha} = \frac{\partial}{\partial \alpha}(\alpha + b) = 1$$

$$\frac{\partial y}{\partial w} = \frac{\partial y}{\partial \alpha}\frac{\partial \alpha}{\partial w} = \frac{\partial y}{\partial \alpha}\frac{\partial}{\partial w}(wx) = 1 \times x$$

Figure 4: Mathematical symbol shapes.

リスト 2: 数学記号の SVG データの定義

Listing 2: Defining SVG data of mathematical symbols.

```
...
[
  {
    id : 'DYDAlpha-OnePath',
    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' },
    ]
  },
  {
    id : 'DYDW-OnePath',
    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' },
    ]
  },
  ...
]
```

(d) の値として格納した (リスト 2, 茶色)．図のように，SVG データ定義ファイルには，clm-associated-svg 要素の ID(青色) を指定することにより標準規格に定義され

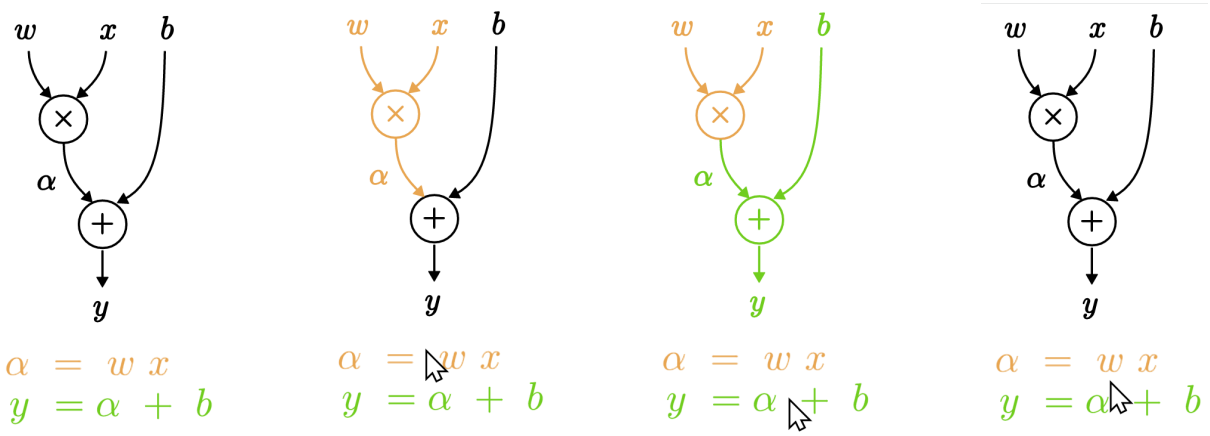

Figure 5: Associating symbols and diagrams of forward path computations: the initial state, click on the upper equation, click on the lower equation, and double click (from left to right).

る SVG 要素の属性 (緑色) を指定できる．

計算グラフについても，clm-associated-svg 要素を用いて形状を表現した．数学記号と同様に，形状を SVG 形式により表現してから，その SVG パスを SVG データ定義ファイルに格納した．

4.1.3 ソフトウエアの構成

可視化のためのソフトウエアは，3.3 節で述べた理由により，サーバとクライアントを使用する構成とした．ソフトウエアの機能を，次のように分担した．

サーバ 計算グラフによる反復的な線形回帰のための計算をおこなう．自動微分により求めた勾配を基にパラメータを更新する．

クライアント 数式，計算グラフ，および，計算過程での重みの勾配の変化のグラフを表示する．そして，それらの対応部分の動的な関連づけをおこなう．関連づけには，clm-associated-svg 要素などの Associated Components を用いる．

計算グラフの重みの勾配などのサーバでの計算結果は，ネットワークを経由してクライアント側に送信される．クライアント側では，受信したデータを用いてグラフをリアルタイムに更新する．なお，ここでの反復的な線形回帰では計算量が少ないので，パラメータを直ぐに計算できる．反復計算であることが分かり易い可視化をおこなうために，サーバでは計算の繰り返しごとに計算結果をクライアントに送信し，その後，一定時間のスリープをおこなっている．

4.1.4 反復的線形回帰の計算

サーバでの線形回帰の計算では，次式を用いる．

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

パラメータを求めるための各処理を，次に示す．

データ生成 $\boldsymbol{x} = \{x_i\}$ を生成する．そして，式 1 により各 x_i について y_i を計算し， $\boldsymbol{y} = \{y_i\}$ とする．y_i の値に正規分布のノイズを加える．

計算グラフの生成 式 1 の計算グラフを生成する．

パラメータの計算 計算グラフの入力を x_i として出力 $\hat{y}_i$ を計算する．そして， $\boldsymbol{y}$ と $\hat{\boldsymbol{y}}$ の平均二乗誤差を計算し，その値を基にしてパラメータ w および b を更新する．以上の計算を，予め決められた回数だけ繰り返す．

4.2 数学記号と計算グラフの関連づけ

4.1.2 節では，反復的線形回帰の説明において用いる数学記号と図の形状を clm-associated-svg 要素を用いて作成した．本節では，その数学記号をクリックすると，計算グラフの対応部分をハイライトするように関連づけをおこなう．

図 5 は，計算グラフを用いた自動微分 [5], [6] による計算のフォワードパスについての結果である．各数式のクリックにより計算グラフの対応部分がハイライトされるのが分かる．また，ダブルクリックにより初期状態に表示がリセットされている．

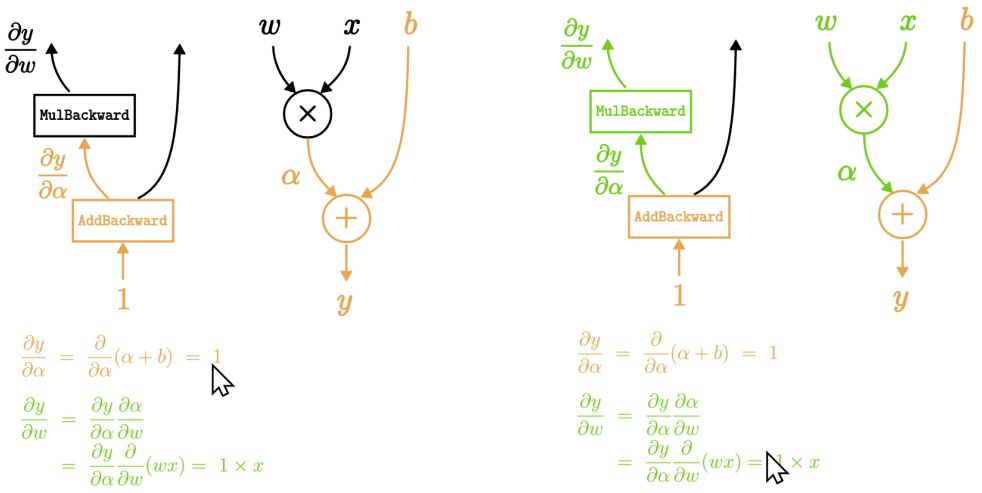

Figure 6: Associating symbols and diagrams of backward path computations: click on the upper equation (left), click on the lower equation (right). Forward path graphs are displayed for easy to understand.

図 6 は，後ろ向き自動微分のバックワードパスについての結果である．各数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るのが分かる．バックワードパスでの計算が分かりやすいように，各グラフの右側にフォワードパスのグラフを表示している．`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の演算名である[3]．`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．

リスト 3 は，図 6 で用いた HTML プログラムでの clm-associated-svg 要素の記述である．カスタム属性 `data-gId` を用いて，id が `DYDWSymbol` の要素のグループ id を `/Diagram/` および `/DYDW/` とした．この要素は，図 6 の $\frac{\partial y}{\partial w}$ である．id が `DYDWArrow` および `MulBackward` の要素についても同様である．id が `DYDAlphaSymbol` の要素のグループ id を `/Diagram/`, `/DYDAlpha/` および `/DYDW/`, id が `AddBackward` の要素のグループ id を `/Diagram/` および `/DYDAlpha/` とした．

id が `DYDW-OnePath` の要素は，図 6 の 2 つ目の方程式である．`onclick` 属性では，この要素のクリックによりグループ id が `/DYDW/` の要素の style 属性の `stroke`, `fill` を設定している．`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 `/Diagram/` の要素の `stroke`, `fill` を初期状態にリセットするように設定した．グループ `/Diagram/` には全要素が所属するように設定しているので，全要素の属性をリセットできる．

[3] この演算名は PyTorch [14] 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．

リスト 3: 図 6 での clm-associated-svg 要素の関連づけ

Listing 3: Associating clm-associated-svg elements in **Fig. 6**.

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

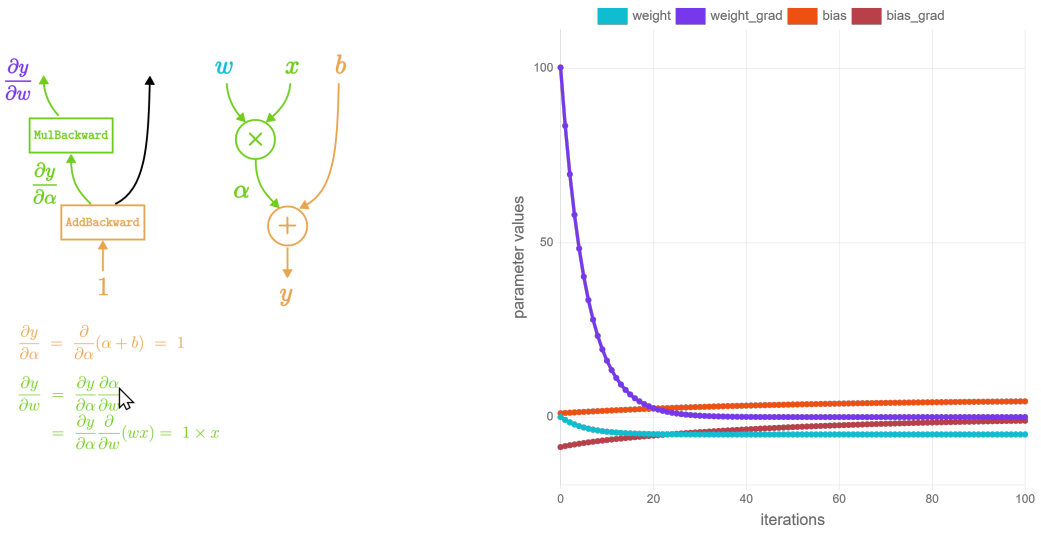

Figure 7: Associating graphical components with chart lines plotted with JavaScript.

id が DYDAlpha-OnePath の要素は，図 6 の 1 つ目の方程式である．onclick 属性では，この要素のクリックによりグループ id が/DYDAlpha/の要素の style 属性の stroke，fill を設定している．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/Diagram/の要素の stroke，fill を初期状態にリセットするように設定した．

id が DYDAlphaSymbol の要素は，グループ/Diagram/, /DYDAlpha/, /DYDW/に所属するのでどちらの方程式をクリックするかにあわせて style 属性が変更されることになる．

以上の要素の関連づけの設定により，図 6 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．

4.3 JavaScript プログラムの動作との関連づけ

サーバとクライアントにより構成されたソフトウエアを用いて，反復的な線形回帰のパラメータの計算過程を可視化した．可視化のために，クライアントの JavaScript プログラムにより折れ線グラフを Web ブラウザに表示した．ここでは，グラフと Associated Components の関連づけをおこなった．

図 7 に，実験結果を示す．手順は，次のとおりである．

Step 1 クライアントに表示されている $\frac{\partial y}{\partial w}$ についての数式をクリックする．これにより，計算グラフの中の $\frac{\partial y}{\partial w}$ が濃い青色，w が薄い青色でハイライトされる．計算グラフの対応する部分の $\frac{\partial y}{\partial w}$ 以外の部分は，緑色でハイライトされる．

Step 2 計算を開始するためのボタンを，クライアントにおいてクリックすると，サーバにおいて反復的に線形回帰のパラメータの計算がおこなわれる．計算の繰り返し毎に計算値はクライアントに送信される．

Step 3 クライアントでは，受信した計算値をグラフにプロットする．重みの勾配の変化を濃い青色，重みの変化を薄い青色でハイライトして表示する．

4.4 検討

4.2 節では，Associated Components を用いて，反復的に線形回帰のパラメータを求めるための計算ネットワークの数式と計算グラフを関連づけた．図 5, 6 から，clm-associated-svg 要素を用いることにより，複雑な形状により表現された偏微分の数式と計算グラフの動作の関連づけが可能であることが分かる．リスト 3 からは，関連づけを HTML プログラムに記述可能であることが分かる．

反復的に線形回帰のパラメータを計算する過程の可視化を，4.3 節でおこなった．線形回帰の計算のために計算グラフを用いた自動微分をサーバでおこない，クライアントではその過程での重みの勾配の変化などをグラフに表示した．図 7 からは，数式，計算グラフ，および，重みの変化などのグラフを表示色により関連づけることにより，それらの対応を分かり易く表示できることが分かる．

Associated Components の特徴は，以上のようなコンポーネントの対応づけを HTML プログラムのみにより記述可能なことである．コンポーネントの関連づけなどの，Web ページ制作でのデザインの試行錯誤を HTML プログラムのみでおこなえる．ページのデザインとシステムのプログラミングを分離できることから，デザイナとプログラマの分業にも有効な手法である．

5 おわりに

本稿では，math augmentation のための既提案手法 [1] により反復的線形回帰の計算過程の可視化をおこなった．カスタマイズした SVG 要素である clm-associated-svg 要素を使用して，数学記号および計算グラフの複雑な形状をグラフィカル・コンポーネントとして柔軟に表現可能であり，それらの関連づけが可能であることを示した．そして，それらのコンポーネントと JavaScript プログラムにより生成したグラフについて，Associated Components の機能による関連づけが可能であることを示した．

今後の課題には，プロキシとしての Associated Components の改良，および，clm-associated-svg 要素で用いる形状データの生成手法などがある．

参考文献

[1] 佐藤信：グラフィカル・コンポーネントの動作による数学記号と説明図の関連づけ, 情報処理学会研究報告, 第 2021-CG-188 巻, pp. 1–8 (2022).

[2] 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, 第 2020-CE-154 巻, pp. 1–8 (2020).

[3] 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, 第 2020-CG-180 巻, pp. 1–8 (2020).

[4] 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, 第 2020-HCI-187 巻, pp. 1–8 (2020).

[5] Merriënboer, B. v., Breuleux, O., et al.: Automatic Differentiation in ML: Where We Are and Where We Should Be Going, in *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, p. 8771?8781 (2018).

[6] Baydin, A. G., Pearlmutter, B. A., et al.: Automatic Differentiation in Machine Learning: A Survey, *J. Mach. Learn. Res.*, Vol. 18, No. 1, p. 5595?5637 (2017).

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

[8] 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).

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

[10] Liu, S., Wang, X., Liu, M. and Zhu, J.: Towards better analysis of machine learning models: A visual analytics perspective, *Visual Informatics*, Vol. 1, No. 1, pp. 48 – 56 (2017).

[11] Liu, M., Shi, J., Li, Z., Li, C., Zhu, J. and Liu, S.: Towards Better Analysis of Deep Convolutional Neural Networks, *IEEE Transactions on Visualization and Computer Graphics*, Vol. 23, No. 1, pp. 91–100 (2017).

[12] 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).

[13] WHATWG, : HTML Living Standard — Last Updated 18 July 2019, `https://html.spec.whatwg.org/(Retrieved: 25 July 2019)`.

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