
ニューラル・ネットワークの数理
Tropical代数入門

ニューラル・ネットワークの数理
Tropical代数入門

はじめに

はじめに

今回のセミナーでは、ニューラル・ネットワークの基本を振り返りながら、ニューラル・ネットワークの基礎理論の候補として最近注目が集まっているTropical代数の紹介と、Tropical代数のニューラル・ネットワークへの応用の紹介を行います。

Tropical代数の紹介は、5月のセミナーでも行う予定です。

そこでは、Tropical代数とカテゴリー論との関係（特に、enriched category論との関係）、量子論との関係（特に、Maslov dequantization の理論）について触れるつもりです。

なぜ、Tropical 代数に注目するのか？
理論的な理由

なぜ、今、Tropical代数が注目を集めているのでしょうか？　そこには、大きく言って2つの理由があると僕は考えています。一つは理論的な理由で、もう一つは実践的な理由です。

第一に、近年の「生成AI」をはじめとして、現代の「人工知能」技術は大きな成功を収めているのですが、ただ、その「成功」がどのような理論的根拠に基づいているのかについての解明は、必ずしも大きく進んでいるわけではありません。

以前に紹介したTai−Danae Bradleyらの、言語とその意味を扱う大規模言語モデルの数学的構造を探ろうとする研究は、そうした状況を打破しようとする貴重な取り組みだと思います。

今また、言語モデルに限らずもっと基本的なレベルで、ニューラル・ネットワークそのものの数学的基礎を探る研究が活発に展開されています。その中で、ニューラル・ネットワークの数学的な基礎理論の候補として浮上しているのが、Tropical代数理論なのです。

なぜ、Tropical 代数に注目するのか？
実践的な理由

Tropical代数に注目が集まる、第二の理由、実践的な理由とは、次のようなものです。

ある変化が起きています。それは、現在の「人工知能」技術が、非常に高価で大規模な、大量の電力を消費するシステムによって支えられているということが、「人工知能」技術の発展自身にとっても足かせになりつつあるという問題意識を、少なくない人が抱きつつあるという変化です。次の「人工知能」技術のイノベーションは、こうした領域で生まれる可能性が高いのです。

Tropical代数は、単純化して言うと、掛け算が足し算に置き換わる奇妙な計算の世界です。ただ、「人工知能」システムで、大きな計算パワーを必要とし、大量の電力を消費しているのが行列とベクトルとの掛け算であることを考えると、Tropical代数が、実際的な技術変化への要請を実現する可能性を持っていることに期待が集まっているのです。

事実、前回のセミナーで紹介した「1−bit LLM」は、理論的にTropical代数に基づいて構築されたものではないのですが、実践的には、16bitの浮動小数点同士の掛け算を、8bitの整数の足し算に置き換えるという実装で、驚異的なパフォーマンスを生み出しています。

今回のセミナーで紹介する論文

今回のセミナーでは、主要に次の論文を紹介しようと思います。

Tropical geometry of deep neural networks,
Liwen Zhang, Gregory Naitzat, and Lek-Heng Lim.
https://arxiv.org/abs/1805.07091

この論文は、2019年のもので、ReLUをactivatorとして持つfeed−forward networkが、Tropical 有理関数と数学的に等価であることを示した画期的な論文です。

AI研究の課題と新しい動向

先にも述べたように、現代の「人工知能」技術は、さまざまな分野で大きな成功を収めているのですが、その「成功」がどのような理論的根拠に基づいているのかについての解明は、必ずしも大きく進んでいるわけではありません。

ただ、こうした状況に変化が起きているように思います。

以前紹介した、Tai-Danae Bradleyらの、大規模言語モデルのカテゴリー論に基づく数学的モデルの提案はその一つです。

今回取り上げる L. Zhang らの仕事も、代数幾何の分野で 21 世紀に入って急速に拡大したTropical代数を用いて「ニューラル・ネットワークの数理」を基礎づけようとする試みです。

近年、AI技術の数学的基礎に対する関心が、広く深く浸透し、拡大しているように、僕は感じています。

それは、実践的な意味を持つか？

一見すると、こうしたAI技術の理論的基礎の研究は、現実に進行するAI技術の変化と実践的には、大きな関連はないように思われるかもしれません。

ただ、そうではないと思います。

それは、遠くないいつかのAI技術の大きな技術的革新を、もっとも確実に準備するものだと僕は考えています。

ニューラル・ネットワークの数理
Tropical代数入門

セミナーは、次のような構成です。

Part 1　DNNの構造を考える

Part 2　Tropical数学入門

Part 3　ニューラル・ネットワークの数理

Part 1　DNNの構造を考える

Part 1 では、基本的なニューラル・ネットワークであるDNNの構造を「関数の合成」として捉え直します。

そこでは、「線形変換」とActivatorによる「非線形変換」の合成を一つのユニットとして、そのユニットの合成が連続的に行われています。

Part 1　DNNの構造を考える

- Deep Neural Netを関数の合成で表わす
- DNNのようなもの
- DNNの「芽」の遍在

Part 2　Tropical代数入門

Part 2 は、Tropical代数の基本の紹介です。

興味深いのは、Tropical多項式の値は、第一にそれを構成する項をすべて線型写像に置き換え、第二に、そうして得られた線型写像の集まりから最小（あるいは最大）のものを選択する「非線形」の過程で計算されることです。

Part 2　Tropical数学入門

- 加算と乗算
- ベクトル・行列計算
- 多項式
- Hypersurface
- 二変数のTropical多項式のHypersurface
- Newton図形

Part 3　ニューラル・ネットワークの数理

Part 3 は、今回のセミナーの中心的なコンテンツになります。

それまでの議論を踏まえて、L. Zhangらの論文をベースに、
DNNの構造とTropical代数の繋がりを解説します。

Zhangらの DNNとTropical有理写像との同値性の証明を追いかけてみたいと思います。

Part 3　ニューラル・ネットワークの数理

- Zhangたちが考えたこと
- mini-plusとmax-plus
- DNNの数学モデルを準備する
- convexな関数の差
- ニューラル・ネットとTropical有理写像
- HypersurfaceとNewton polygonの対応
- Tropical幾何とニューラル・ネットワーク

参考資料

Tropical Geometryについては、次の資料が基本的なものだと思います。

Maclagan, Diane and Sturmfels, Bernd.
Introduction to tropical geometry,
volume 161. American Mathematical Society, 2015

https://www.ams.org/books/gsm/161/gsm161-endmatter.pdf

Diane Maclagan自身による、全5回の講義が、YouTubeで公開されています。
“Introduction to Tropical Algebraic Geometry (1-5)”
https://www.youtube.com/watch?v=unjVp6HQVmc

共著者のBernd Sturmfels もYouTubeで講義を連続公開しています。
“Tropical Geometry – Lecture 1– Plane Curves”
https://youtu.be/TNwCzl02uck

数学の世界だけでなく、情報の世界でも Tropical 幾何に対する関心は広がっています。IEEEのproceeding に掲載された次の論文は、その代表的なものだと思います。

"Tropical Geometry and Machine Learning"

https://ieeexplore.ieee.org/document/9394420


Part 1
DNNの構造を考える
ニューラル・ネットワークの数理 -- Tropical代数入門

Part 1　DNNの構造を考える

- Deep Neural Netを関数の合成で表わす
- DNNのようなもの
- DNNの「芽」の遍在

ニューラル・ネットワークの数理 -- Tropical代数入門

Deep Neural Netを
関数の合成で表わす

このセッションでは、DNN (Deep Neural Network)を関数の合成で表してみようと思います。

ただ、DNNの特徴を捉えるために、少し準備が必要です。
ここで、改めてDNNのグラフ表示を出発点にします。

以前のセッションで、少し抽象度の高いDNNのグラフ表示を見てきました。抽象度が高いというのは、すべてのDNNのグラフは、具体的なノード間の接続を考えずとも、$\phi(Wx + b)$という式を表す基本的なグラフの接続で表現できるということでした。

ここから直接、DNNは関数の合成として表現できることを導くことは簡単にできるのですが、今回は、Liwen Zhangらの原論文のやり方に従うことにしました。

少し回り道ですが、DNNの構造を知るには、いい練習問題かもしれません。

DNNを関数の合成として表す2つのアプローチが、結局は同じものだということは、すぐに分かると思います。

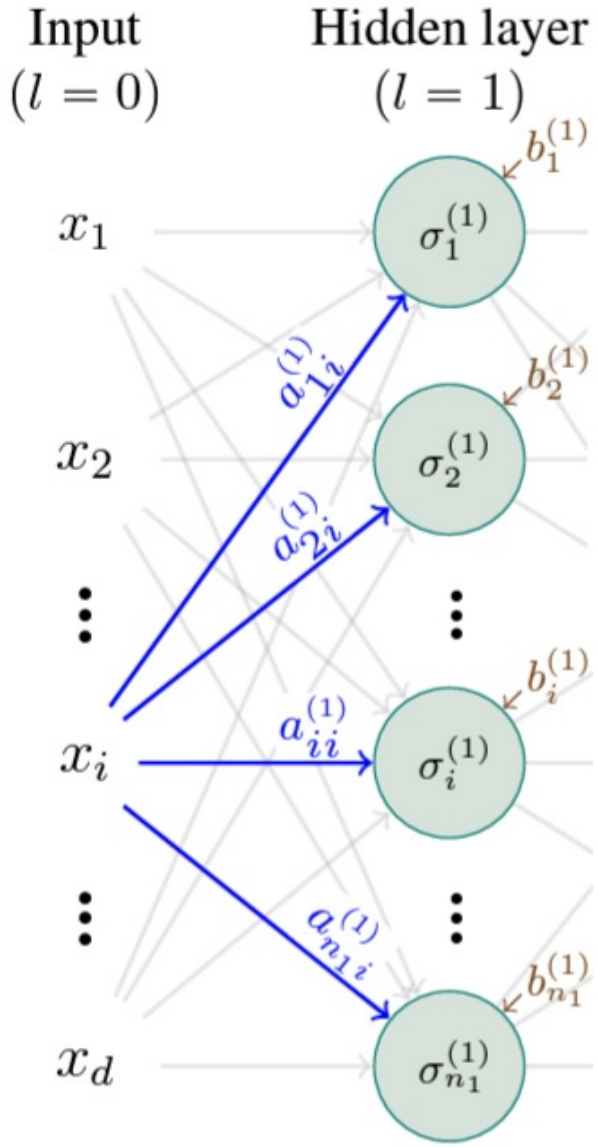
入力層と最初の隠れ層の働きを関数の合成で定義する

まず最初に、入力層と最初の隠れ層のグラフを考え、その働きを関数の合成で定義することを考えます。

次に、中間のl番目とi+1番目の隠れ層のグラフを考え、その働きを関数の合成で表すことを考えます。最初に行った入力層と最初の隠れ層の働きを関数の合成で表したやり方を、すこし広げるといいのです。

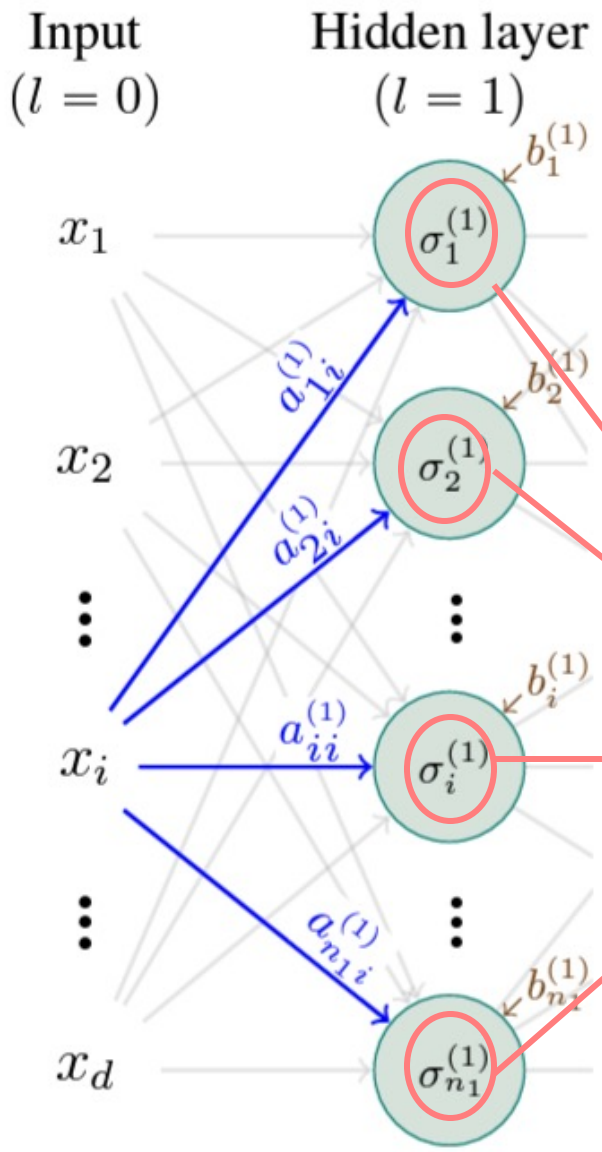
最後に、最後の隠れ層と出力層の関係を考えます。

原論文のグラフは、次のようなものです。



このグラフには、入力層($l = 0$)と、それに隣接する最初の隠れ層($l = 1$)の接続が描かれています。

強調されているブルーの線は、d次元の入力列ベクトル $[x_1, x_2, \ldots, x_d]^T$ の要素の一つであるx_iの情報が、どのように次の($l = 1$)層に伝えられているかを示しています。



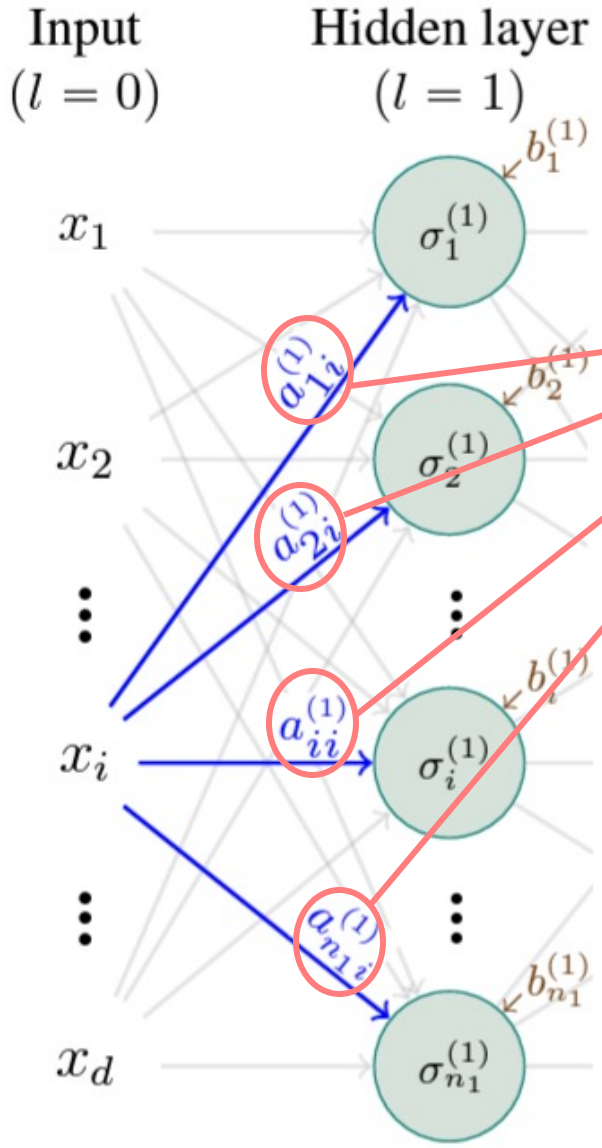
x_iの情報が何個のニューロンに届けられているかは、すなわち、$(l=1)$層が何個のニューロンからなるかは、4本の矢印以外は図では省略されています。

ただ、$(l=1)$層を構成するニューロンを表すサークルの内側の記号の並び

$$\sigma_1^{(1)}, \sigma_2^{(1)}, \ldots, \sigma_i^{(1)}, \ldots, \sigma_{n_1}^{(1)}$$

を見ると、$(l=1)$層にはn_1個のニューロンがあることがわかります。

$\sigma_i^{(1)}$は、実は、$(l=1)$層のActivatorを表しているのですが、それについては後で説明します。



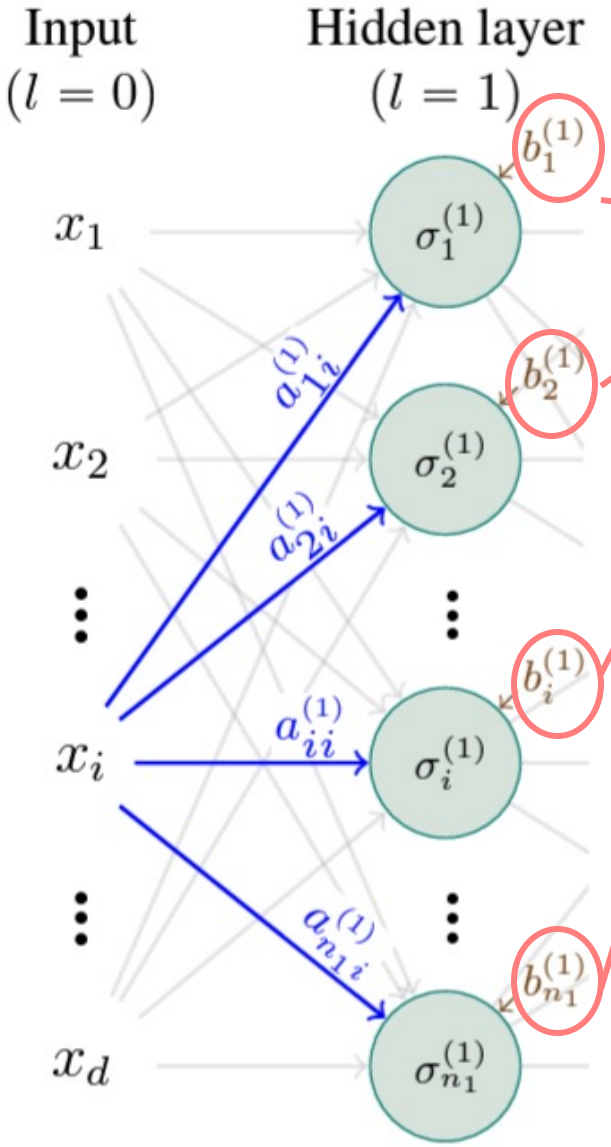
強調されているブルーの線上の次の記号は、何を表しているのでしょう？

$$a_{1i}^{(1)}, a_{2i}^{(1)}, \ldots, a_{ii}^{(1)}, \ldots, a_{n_1 i}^{(1)}$$

これは、x_iと$(l = 1)$層のn_1個のニューロンを結ぶシナプスの重みを表しています。

$1 \leq i \leq d$ であるすべてのiと$1 \leq j \leq n_1$であるすべてのjについて、$a_{ij}^{(1)}$なる重みを集めて、次のように「重み行列」$A^{(1)}$を作ることができます。

$$A^{(1)} = \begin{bmatrix} a_{11}^{(1)} & \cdots & a_{1i}^{(1)} & \cdots & a_{1d}^{(1)} \\ a_{21}^{(1)} & \cdots & a_{2i}^{(1)} & \cdots & a_{2d}^{(1)} \\ \vdots & \vdots & \vdots & & \vdots \\ a_{n_1 1}^{(1)} & \cdots & a_{n_1 i}^{(1)} & \cdots & a_{n_1 d}^{(1)} \end{bmatrix}$$



最後に、これはなんでしょう？

$b_1^{(1)}, b_2^{(1)}, \ldots, b_i^{(1)}, \ldots, b_{n_1}^{(1)}$

容易に想像できるように、これは、$(l = 1)$層の「バイアス」です。

バイアスのベクトルの次元は、その層のニューロンの個数と一致します。

$(l = 1)$層のニューロンの個数はn_1ですので、バイアスのベクトルの次元はn_1です。

入力層と最初の隠れ層との関係を ベクトルの次元で整理する

実数のn次元のベクトルは、$\mathbb{R}^n$ で表現されます。

入力層に与えられたデータをベクトルxとすると、$x \in \mathbb{R}^d$

隠れ層では、Activator 適用以前には、線形変換の基本式

$$y = Ax + b$$

にしたがって、次のような計算が行われています。

$$\begin{bmatrix} a_{11}^{(1)} & \cdots & a_{1i}^{(1)} & \cdots & a_{1d}^{(1)} \\ a_{21}^{(1)} & \cdots & a_{2i}^{(1)} & \cdots & a_{2d}^{(1)} \\ \vdots & \vdots & \vdots & & \vdots \\ a_{n_1 1}^{(1)} & \cdots & a_{n_1 i}^{(1)} & \cdots & a_{n_1 d}^{(1)} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_d \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_{n_1} \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_{n_1} \end{bmatrix}$$

左辺のn_1行d列の行列Aとd行のベクトルxの積が、n_1次元の行ベクトルになることを確認してください。b, $y \in \mathbb{R}^{n_1}$です。

Activate以前の関数 ρ

Activatorの適用前の、$(l = 1)$層の線形変換の働きを、入力を$\nu^{(0)}$とする関数$\rho^{(0)}$として次のように表すことにします。先のネーミングとつながるように、$\nu^{(0)}(x) \coloneqq x$とします。

$$x = \nu^{(0)}(x) \in \mathbb{R}^d, A^{(1)} \in \mathbb{Z}^{n_1 \times d}, \mathrm{b}^{(1)} \in \mathbb{R}^{n_1}で$$
$$\rho^{(1)}\left(\nu^{(0)}\right) \coloneqq A^{(1)}\nu^{(0)} + \mathrm{b}^{(1)}$$

この時、関数 $\rho^{(1)}$は、

$$\rho^{(1)} \colon \mathbb{R}^d \to \mathbb{R}^{n_1}$$

と表すことができます。

Activator σ

Activator σは、ベクトルyに対しても、そのベクトルの要素であるスカラーy_iに対しても、要素ごとに次のように定義されます。

$$\sigma(y) = \sigma\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_{n_1} \end{bmatrix} = \begin{bmatrix} \sigma(y_1) \\ \sigma(y_2) \\ \vdots \\ \sigma(y_{n_1}) \end{bmatrix}$$

ベクトルを引数に持つ関数$\sigma^{(1)}$は、次のように定義できます。

$$\sigma^{(1)}: \mathbb{R}^{n_1} \rightarrow \mathbb{R}^{n_1}$$

入力層と最初の隠れ層の働きを関数の合成で定義する

入力層($l=0$)への入力を、$\nu^{(0)}$

Activatorの適用前の($l=1$)層の線形変換を、$\rho^{(1)}$

($l=1$)層のActivator を、$\sigma^{(1)}$とし、

Activator 適用後の($l=1$)層の最終出力を、$\nu^{(1)}$とすると、

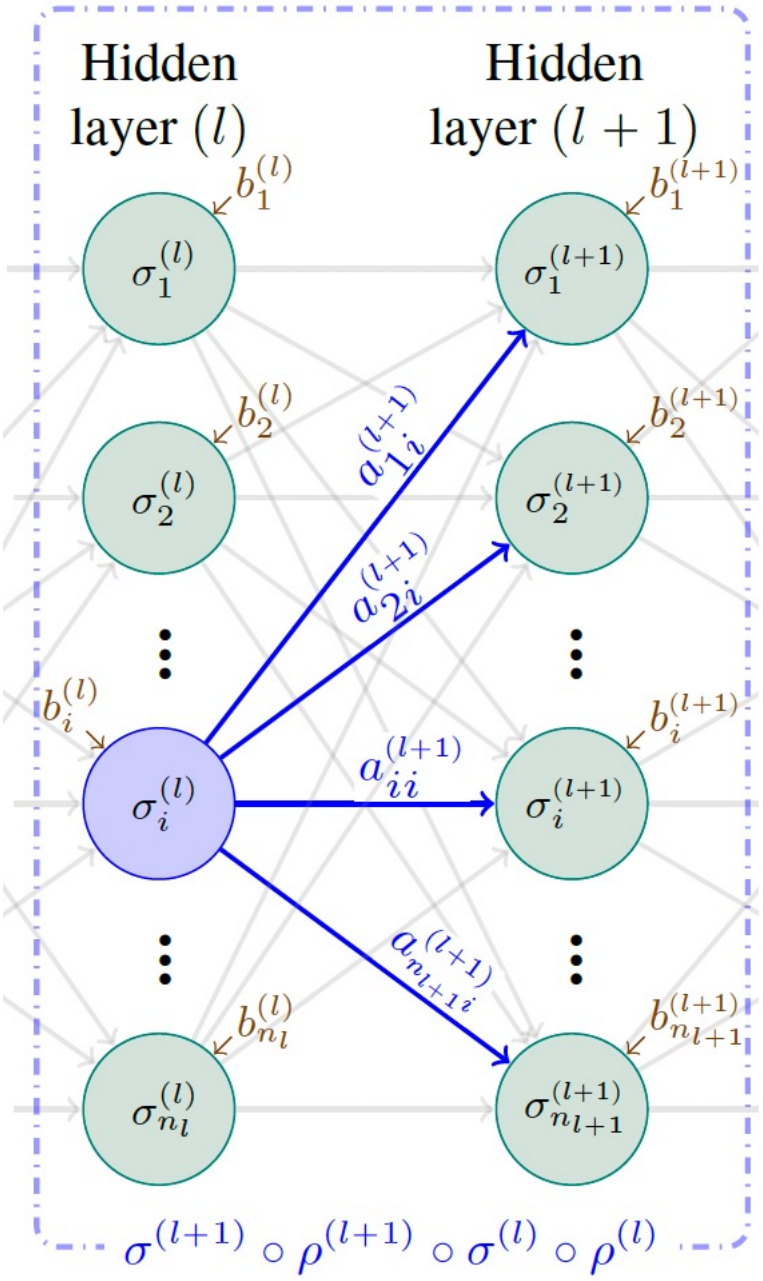
次の式が成り立つことがわかります。

$$\nu^{(1)} = \sigma^{(1)} \circ \rho^{(1)} \left(\nu^{(0)}\right)$$

入力層と最初の隠れ層の働きは、関数の合成 $\sigma^{(1)} \circ \rho^{(1)}$で定義されています。

l番目と$l+1$番目の隠れ層のグラフ

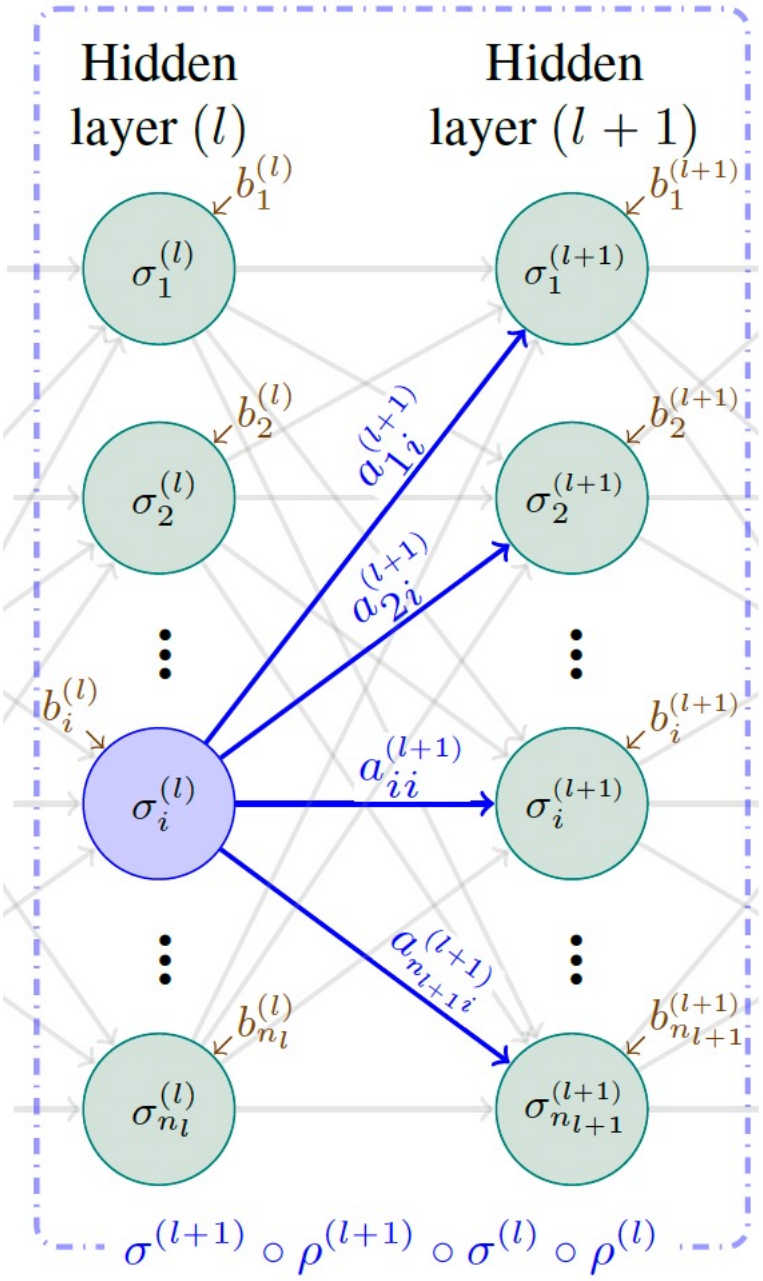
中間のl番目と$l+1$番目の隠れ層のグラフは、次のようになります。



層lには、n_l個のニューロンがあり、層$l+1$には、n_{l+1}個のニューロンがあることをグラフから確認してください。

グラフでブルーの矢印で表されているのは、l層のi番目のニューロンから、$l+1$層のすべてのニューロンに接続があることを表しています。（4つ以外は省略されていますが）

このブルーの接続では、どんな値が$l+1$層に渡されているのでしょうか？



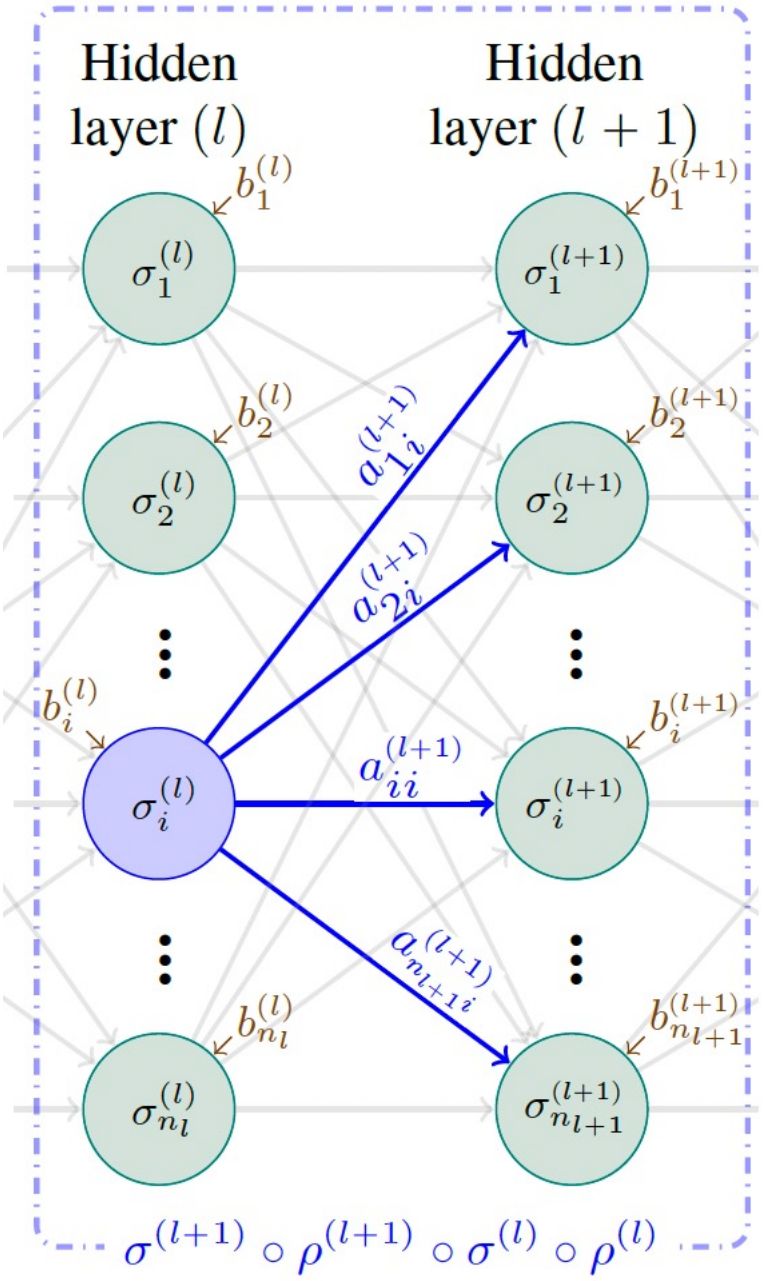
その値は、ブルーの線に添えられているブルーの

$a_{1i}^{(l+1)}, a_{2i}^{(l+1)}, \ldots, a_{ii}^{(l+1)}, \ldots, a_{n_1 i}^{(l+1)}$

値ではありません。

このブルーの値は、$l+1$層の重み行列$A^{(l+1)}$のi列目を構成するものです。

σ_i^lと記されたサークルのニューロンから$l+1$層のニューロンに届けられる値は、すべて等しいもので、$\sigma^{(l)} \circ \rho^{(l)}$で計算される$l$層の出力$\nu^{(l)}$の$i$番目の要素です。



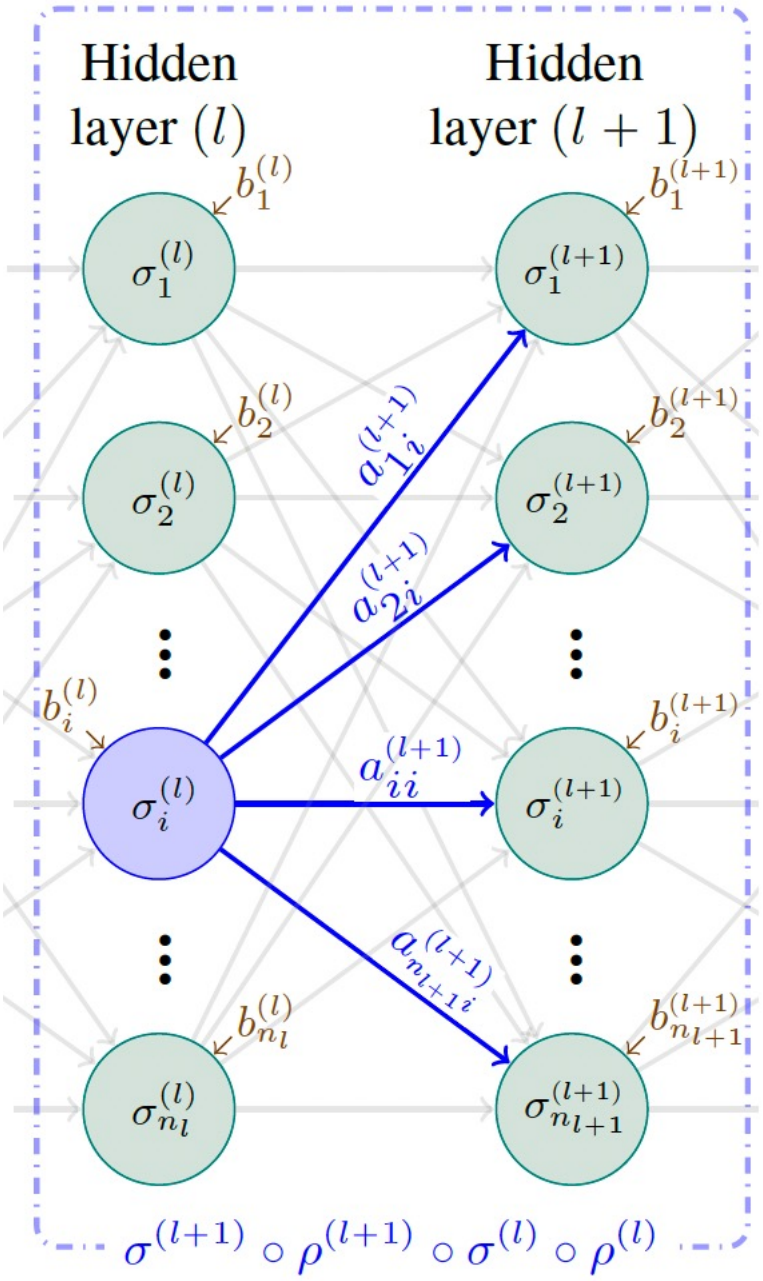
それでは、$l+1$層の出力$\nu^{(l+1)}$は、どうなるのでしょう？

それは、先に見た$l+1$層の重み行列$A^{(l+1)}$とバイアス$b^{(l+1)}$を使って

$$\rho^{(l+1)}\left(\nu^{(l)}\right) = A^{(l+1)}\nu^{(l)} + b^{(l+1)}$$

を計算し、その結果にActivator $\sigma^{(l+1)}$を適用したものです。

$$\nu^{(l+1)} = \sigma^{(l+1)} \circ \rho^{(l+1)}\left(\nu^{(l)}\right)$$



一つ注意したいのは、

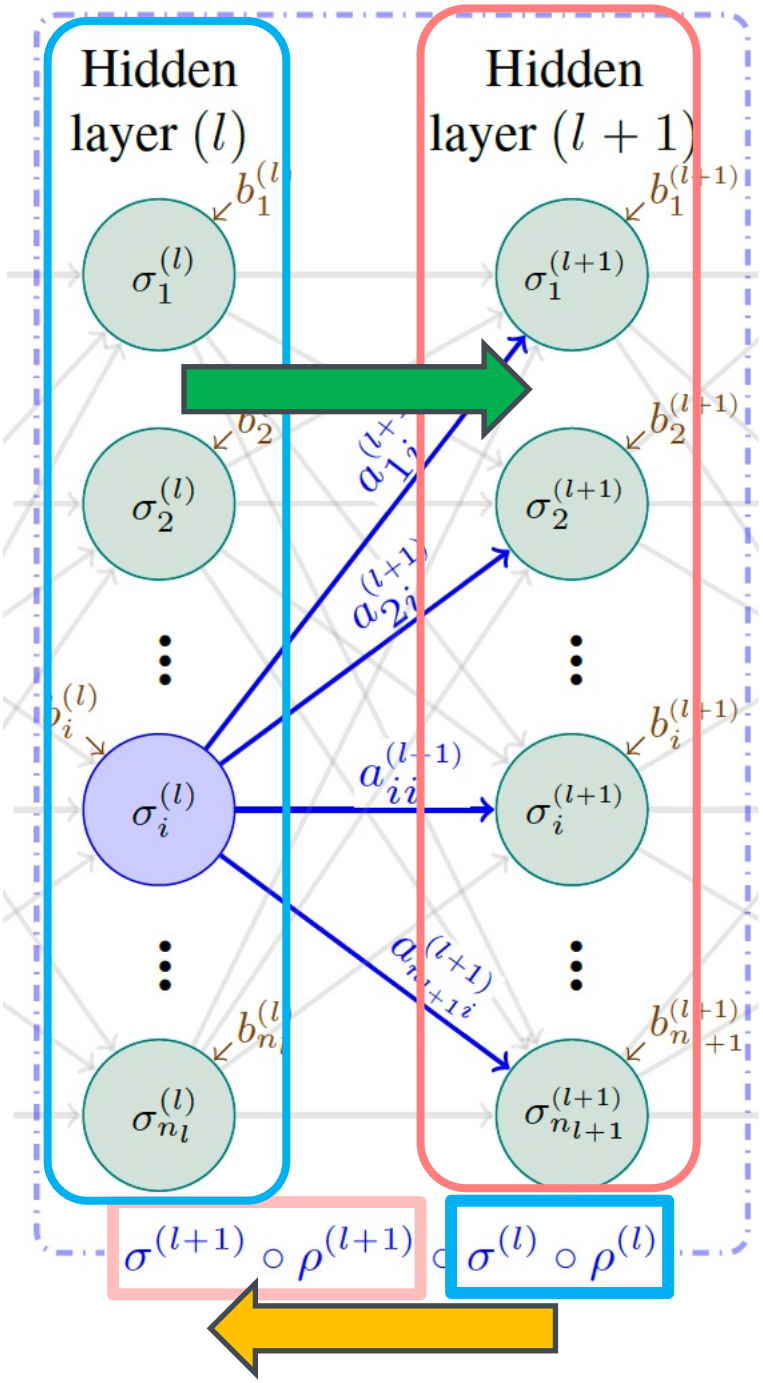
l層の出力は、$\sigma^{(l)} \circ \rho^{(l)}$で計算され、

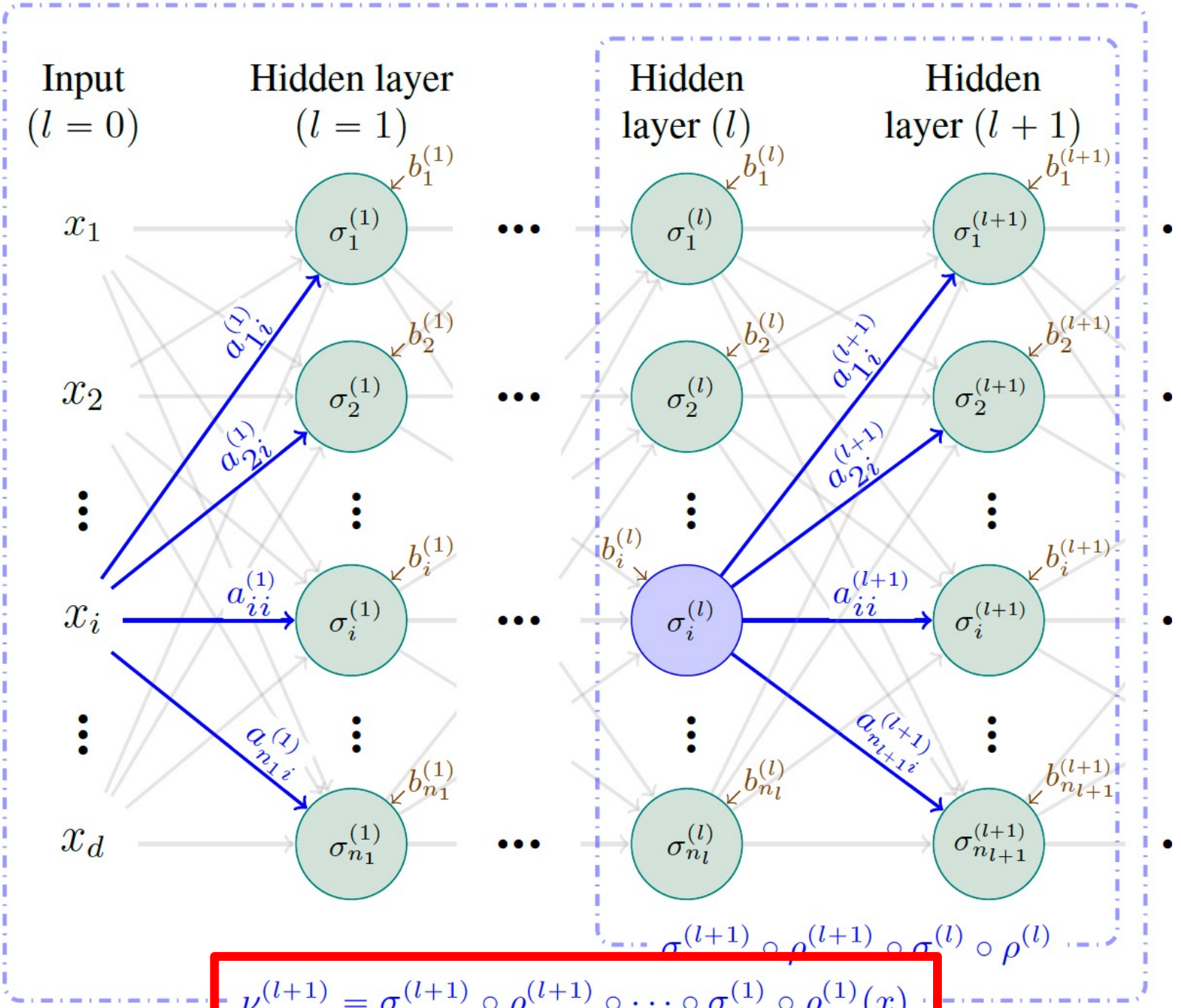
$l+1$層の出力は、$\sigma^{(l+1)} \circ \rho^{(l+1)}$で計算されるということなのですが、

グラフ上での情報の流れと、

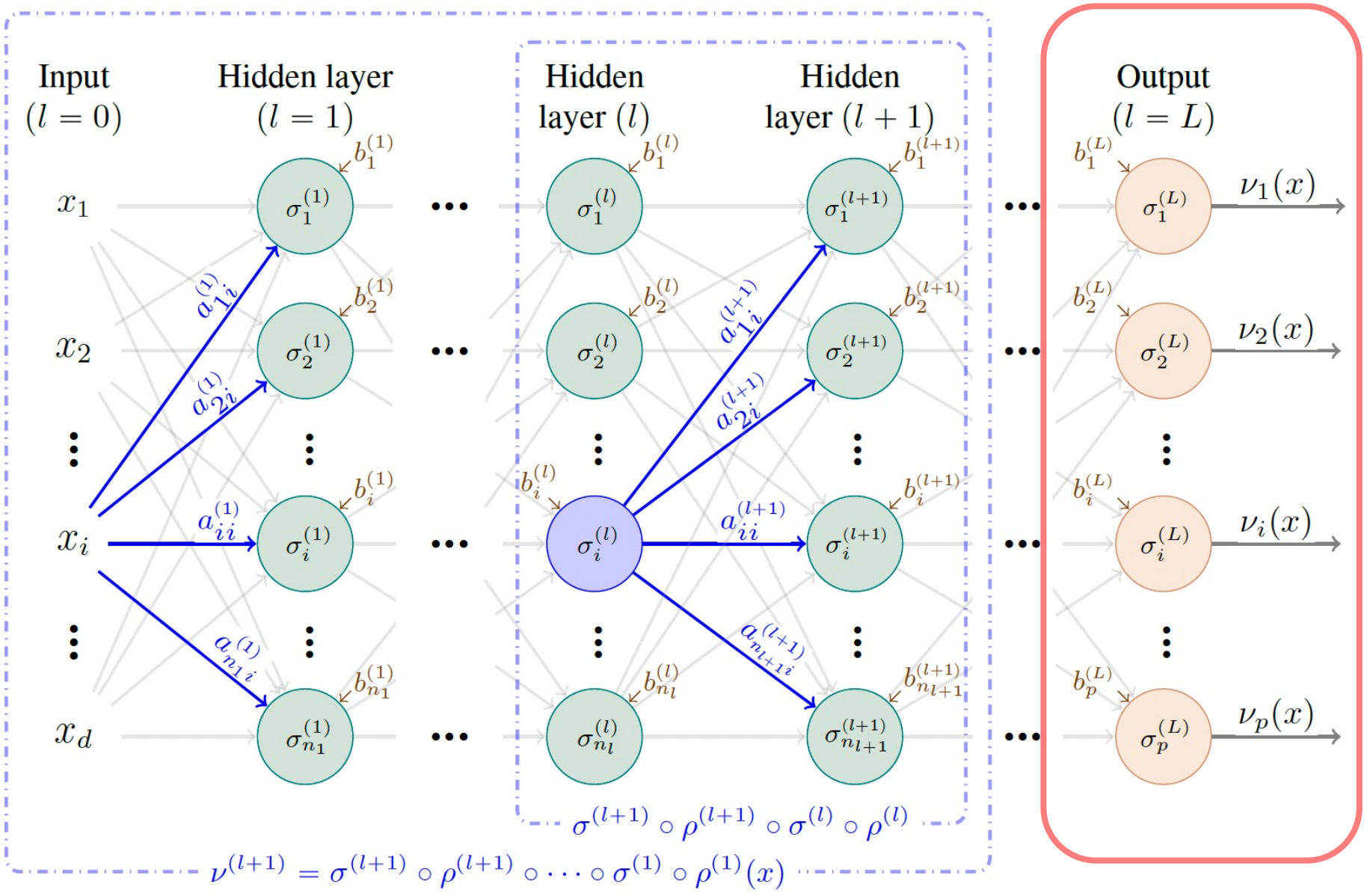
関数の合成の式の並びが、反対になっていることです。

ここまでのまとめを次の図に示します。赤い枠で囲った式が、ここまでのDNNの層の関数の合成での表示です。




Input
$(l = 0)$
Hidden layer
$(l = 1)$
Hidden
layer (l)
Hidden
layer $(l + 1)$
x_1
x_2
x_i
x_d
$\sigma_1^{(1)}$
$\sigma_2^{(1)}$
$\sigma_i^{(1)}$
$\sigma_{n_1}^{(1)}$
$\sigma_1^{(l)}$
$\sigma_2^{(l)}$
$\sigma_i^{(l)}$
$\sigma_{n_l}^{(l)}$
$\sigma_1^{(l+1)}$
$\sigma_2^{(l+1)}$
$\sigma_i^{(l+1)}$
$\sigma_{n_{l+1}}^{(l+1)}$
$b_1^{(1)}$
$b_2^{(1)}$
$b_i^{(1)}$
$b_{n_1}^{(1)}$
$b_1^{(l)}$
$b_2^{(l)}$
$b_i^{(l)}$
$b_{n_l}^{(l)}$
$b_1^{(l+1)}$
$b_2^{(l+1)}$
$b_i^{(l+1)}$
$b_{n_{l+1}}^{(l+1)}$
$a_{1i}^{(1)}$
$a_{2i}^{(1)}$
$a_{ii}^{(1)}$
$a_{n_1 i}^{(1)}$
$a_{1i}^{(l+1)}$
$a_{2i}^{(l+1)}$
$a_{ii}^{(l+1)}$
$a_{n_{l+1} i}^{(l+1)}$
$\sigma^{(l+1)} \circ \rho^{(l+1)} \circ \sigma^{(l)} \circ \rho^{(l)}$
$\nu^{(l+1)} = \sigma^{(l+1)} \circ \rho^{(l+1)} \circ \cdots \circ \sigma^{(1)} \circ \rho^{(1)}(x)$

出力層までのグラフ


Input
$(l=0)$
Hidden layer
$(l=1)$
Hidden
layer (l)
Hidden
layer $(l+1)$
Output
$(l=L)$
x_1
x_2
x_i
x_d
$\sigma_1^{(1)}$
$\sigma_2^{(1)}$
$\sigma_i^{(1)}$
$\sigma_{n_1}^{(1)}$
$\sigma_1^{(l)}$
$\sigma_2^{(l)}$
$\sigma_i^{(l)}$
$\sigma_{n_l}^{(l)}$
$\sigma_1^{(l+1)}$
$\sigma_2^{(l+1)}$
$\sigma_i^{(l+1)}$
$\sigma_{n_{l+1}}^{(l+1)}$
$\sigma_1^{(L)}$
$\sigma_2^{(L)}$
$\sigma_i^{(L)}$
$\sigma_p^{(L)}$
$b_1^{(1)}$
$b_2^{(1)}$
$b_i^{(1)}$
$b_{n_1}^{(1)}$
$b_1^{(l)}$
$b_2^{(l)}$
$b_i^{(l)}$
$b_{n_l}^{(l)}$
$b_1^{(l+1)}$
$b_2^{(l+1)}$
$b_i^{(l+1)}$
$b_{n_{l+1}}^{(l+1)}$
$b_1^{(L)}$
$b_2^{(L)}$
$b_i^{(L)}$
$b_p^{(L)}$
$a_{1i}^{(1)}$
$a_{2i}^{(1)}$
$a_{ii}^{(1)}$
$a_{n_1 i}^{(1)}$
$a_{1i}^{(l+1)}$
$a_{2i}^{(l+1)}$
$a_{ii}^{(l+1)}$
$a_{n_{l+1}i}^{(l+1)}$
$\nu_1(x)$
$\nu_2(x)$
$\nu_i(x)$
$\nu_p(x)$
$\sigma^{(l+1)} \circ \rho^{(l+1)} \circ \sigma^{(l)} \circ \rho^{(l)}$
$\nu^{(l+1)} = \sigma^{(l+1)} \circ \rho^{(l+1)} \circ \cdots \circ \sigma^{(1)} \circ \rho^{(1)}(x)$

ニューラル・ネットワークの数理 -- Tropical代数入門

いまどき、DNNなの？

今回のセミナーは、ニューラル・ネットワークの基礎の数学理論としてTropical代数に注目が集まっていることを紹介しようとしています。

ただ、その理論展開のフィールドは、突破口となったZhangらの論文がそうであったことによるのですが、主要に、DNN (Deep Neural Network) に限られています。

「いまどき、DNNなの？」

そう思った人も多いと思います。

たしかに。

ニューラル・ネットワークのアーキテクチャーは、時代とともに大きく変化しています。現代の眼からみれば、DNNアーキテクチャーは「過去の遺物」のように見えるはずです。

それに、DNNの数学的基礎が解明されたとしても、それが、現代のニューラル・ネットワークの数学的基礎を与えるとは限りません。

人間向けの「分類器」？

今回と次回のセッションで、様々なニューラル・ネットワーク・アーキテクチャーのいろんなところに、DNNに起源を持つ構造が生き残っているというという話をしたいと思います。

確かに、さまざまなニューラル・ネットワーク・アーキテクチャーの最後の層に、大抵はSoftMaxをActivator としたDNN層が置かれるのは普通です。それは、新しいアーキテクチャーが新しく獲得した能力の中では二次的な役割しか果たしていないように見えます。

DNNの残存は、人間向けの「分類器」、あるいは、他のシステムに情報をtransfer するための単なる「正規化装置」として解釈できると思います。

DNNは、どこへ行ったのか

ただ、DNNというアーキテクチャーが、ニューラル・ネットワーク・アーキテクチャーの進化の中で、二次的な「外部向けのインターフェース」に退化したのだとは、僕は考えていません。

DNNアーキテクチャーは形を変えて、「DNN的なもの」としてニューラル・ネットワーク・システムの全体に浸透・拡大したのだと僕は思っています。

「DNN的なもの」

ここで、「DNN的なもの」「DNNのようなもの」とは、どんなものなのかを述べてみようと思います。

DNNは、前回のセッションで見たように、次のようなタイプの合成の繰り返しの形をしています。

$$\cdots \boldsymbol{\sigma} \circ \boldsymbol{\rho} \circ \boldsymbol{\sigma} \circ \boldsymbol{\rho} \circ \boldsymbol{\sigma} \circ \boldsymbol{\rho} \cdots$$

ここで、

ρ は、$Ax + \mathrm{b}$ の形の線形写像で、

σ は、Activator で非線形写像です。

DNNの「芽」

DNNに「起源を持つDNNのようなもの」とか、「DNN的なもの」といった表現は曖昧なので、これらを「DNNの芽」と呼んで、次のように定義したいと思います。

ニューラル・ネットワークの部分システムが、線形写像 ρ と非線形写像 σ の合成として $\boldsymbol{\sigma \circ \rho}$ の形で表される時、それを「DNNの芽」と呼ぶ。

次のセッションでは、さまざまなニューラル・ネットワークのアーキテクチャーに、アーキテクチャーの違いを超えて、こうした「DNNの芽」がたくさん含まれていることを見ていきたいと思います。

ニューラル・ネットワークの数理 -- Tropical代数入門

「DNNの芽」を探す

このセッションでは、さまざまなニューラル・ネットワークのアーキテクチャーに、アーキテクチャーの違いを超えて、先に見た「DNNの芽」がたくさん含まれていることを見ていきたいと思います。

「DNNの芽」と呼んでいるのは、次のものです。

ニューラル・ネットワークの部分システムが、線形写像 ρ と非線形写像 σ の合成として $\boldsymbol{\sigma \circ \rho}$ の形で表される時、それを「DNNの芽」と呼ぶ。

様々なニューラル・ネットワークのアーキテクチャー

まず、簡単にAIの歴史を振り返って、どのようなアーキテクチャーが生まれてきたかを見てみたいと思います。

最初に、DNN = Multi Layer Perceptron の出発点となった、RosenblattのPerceptron について見ていきたいと思います。
Perceptronが「DNNの芽」を持つのは当然です。

以下のPerceptronのパートは、2017年7月のマルレク「人工知能の歴史を振り返る」から転用しています。
https://www.marulabo.net/docs/20170731-marulec02/

ここでは、次のようなアーキテクチャーを取り上げます。

- Perceptron
- DNN -- Multi Layer Perceptron
- CNN -- Convolutional Neural Network
- RNN -- Recurrent Neural Network
- LSTM
- Transformer

結論から言うと、これらの様々なアーキテクチャーの全てに、
「DNNの芽」が含まれていることが確かめられます。

「DNNの芽」は、ニューラル・ネットワークに遍在するのです。

DNNの数学的基礎の研究が、ニューラル・ネットワークの数学的基礎の研究への第一歩になる

このことは、Tropical代数を用いたDNNの数学的基礎の研究が、ニューラル・ネットワーク全般の数学的基礎の研究への第一歩になることを示していると、僕は考えています。

Perceptron

Rosenblattのパーセプトロンは、現在のニューラル・ネットワークの最初の起原にあたる。

パーセプトロンの限界を指摘したのは、Minskyである。

1958年 Perceptron

1943年 Formal Neuron

A LOGICAL CALCULUS OF THE IDEAS IMMANENT IN NERVOUS ACTIVITY

WARREN S. MCCULLOCH AND WALTER PITTS

FROM THE UNIVERSITY OF ILLINOIS, COLLEGE OF MEDICINE,
DEPARTMENT OF PSYCHIATRY AT THE ILLINOIS NEUROPSYCHIATRIC INSTITUTE,
AND THE UNIVERSITY OF CHICAGO

Because of the "all-or-none" character of nervous activity, neural events and the relations among them can be treated by means of propositional logic. It is found that the behavior of every net can be described in these terms, with the addition of more complicated logical means for nets containing circles; and that for any logical expression satisfying certain conditions, one can find a net behaving in the fashion it describes. It is shown that many particular choices among possible neurophysiological assumptions are equivalent, in the sense that for every net behaving under one assumption, there exists another net which behaves under the other and gives the same results, although perhaps not in the same time. Various applications of the calculus are discussed.

- w：重みづけ（実数）
- x：入力信号（0 または 1）
- h：しきい値（実数）
- H：ヘヴィサイドの階段関数（出力は 0 または 1）

$$H\left(\sum_{i=1}^{N} w_i x_i - h\right)$$

AND

$$H(x_1 + x_2 - 1.5)$$

OR

$$H(x_1 + x_2 - 0.5)$$

NOT

$$H(-x_1 + 0.5)$$

XOR

$$H(x_1 + x_2 - 2H(x_1 + x_2 - 1.5) - 0.5)$$

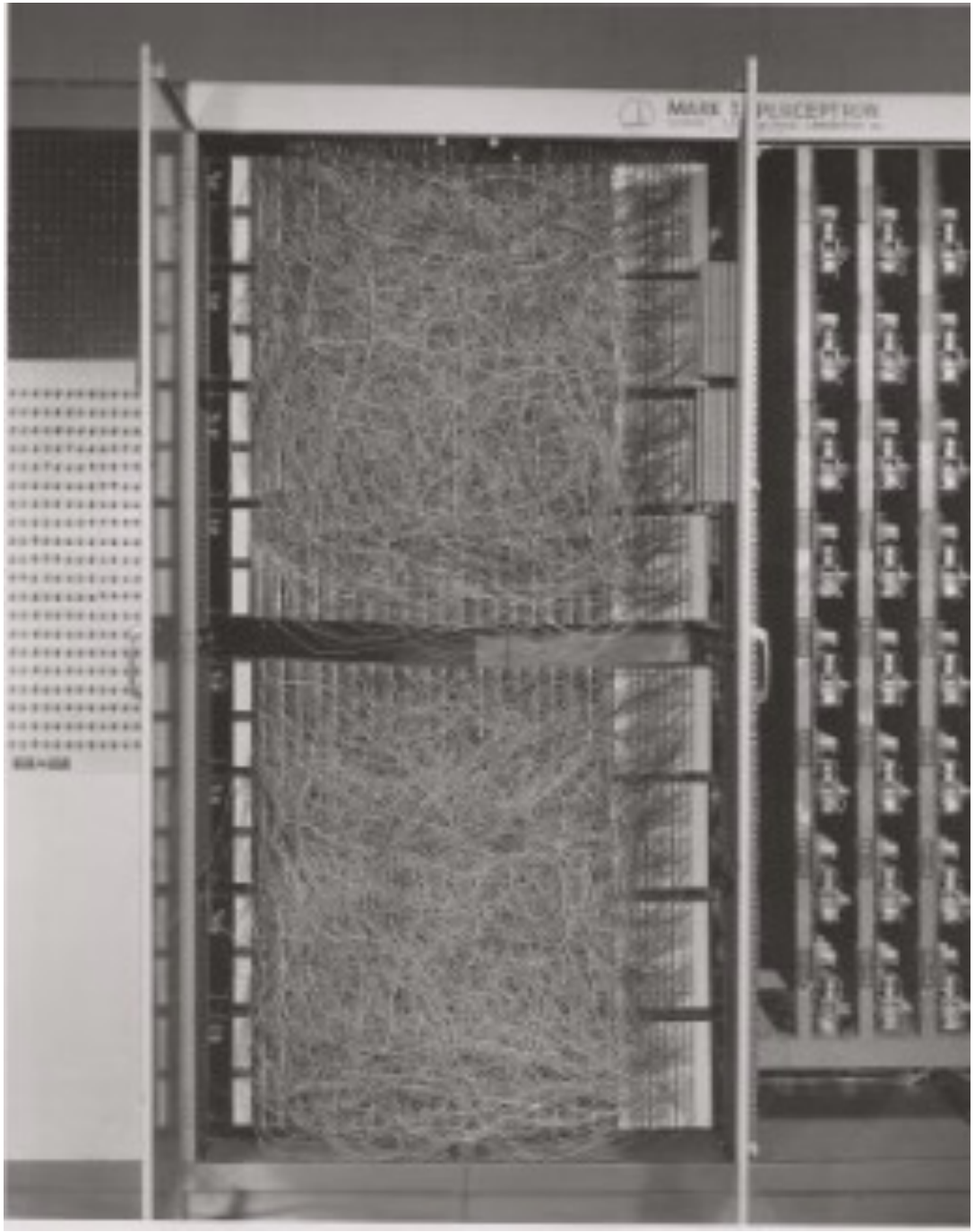
https://goo.gl/4GCkQn

“The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain” Rosenblatt, Frank (1958)

「ここに報告された理論は、認知システムの組織に対する定量的で統計的アプローチが、実現可能で実り多いことを明確に示している。パーセプトロンのようなシステムの研究によって、機械や人間を含む情報を扱うシステムの全てに共通する基本的な組織の法則性が最終的に理解されることが期待される。」

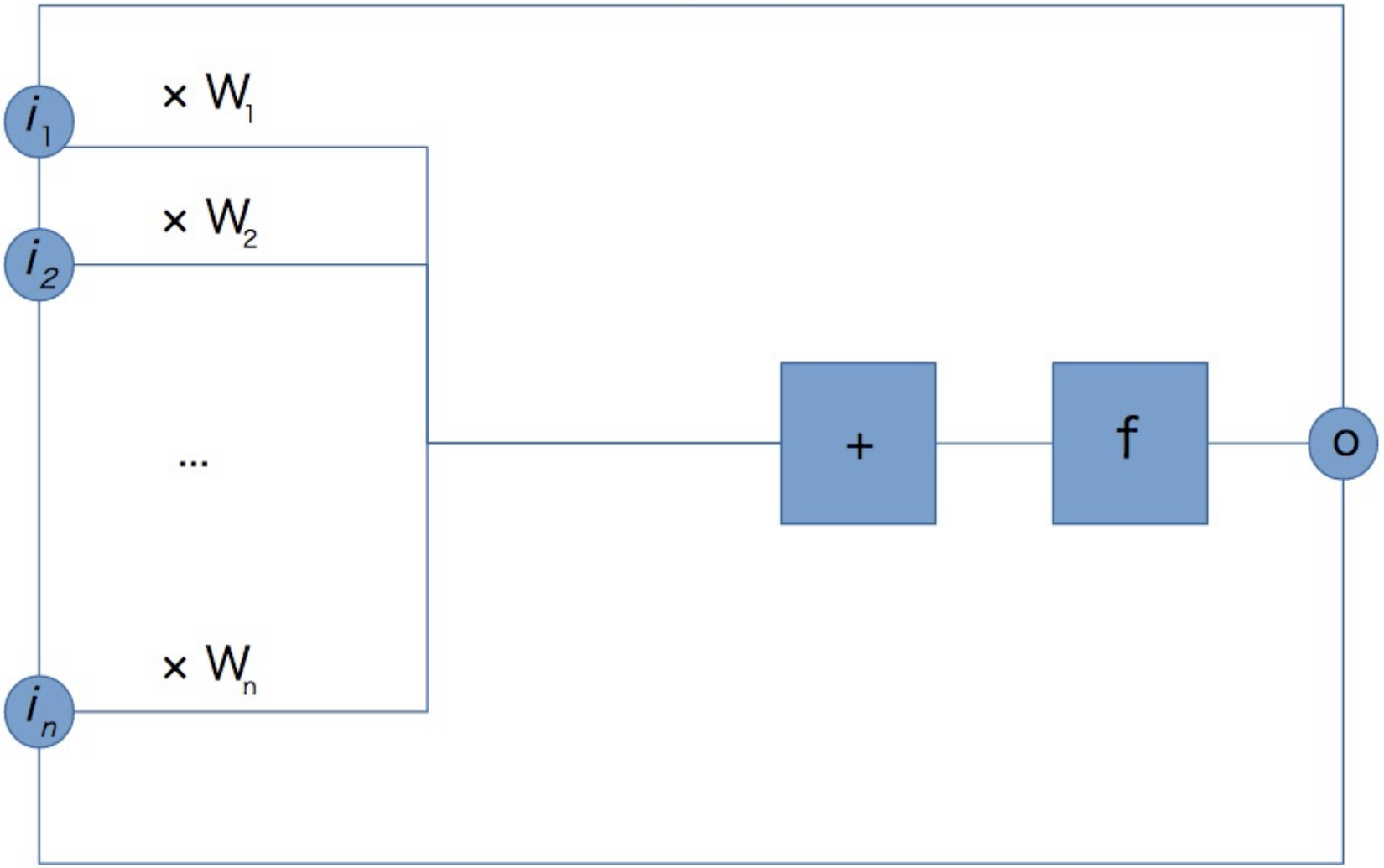
「視覚と脳の機能をモデル化したものであり、パターン認識を行う。シンプルなネットワークでありながら学習能力を持つ。1960年代に爆発的なニューラルネットブームを巻き起こしたが、1969年に人工知能学者マービン・ミンスキーらによって線形分離可能なものしか学習できないことが指摘されたことによって下火となった。」

https://goo.gl/ylS7sj



The Mark I Perceptron machine was the first implementation of the perceptron algorithm. The machine was connected to a camera that used 20×20 cadmium sulfide photocells to produce a 400-pixel image. The main visible feature is a patchboard that allowed experimentation with different combinations of input features. To the right of that are arrays of potentiometers that implemented the adaptive weights.[2]:213

Perceptron



$$o = f\left(\sum_{k=1}^{n} i_k \cdot W_k\right)$$

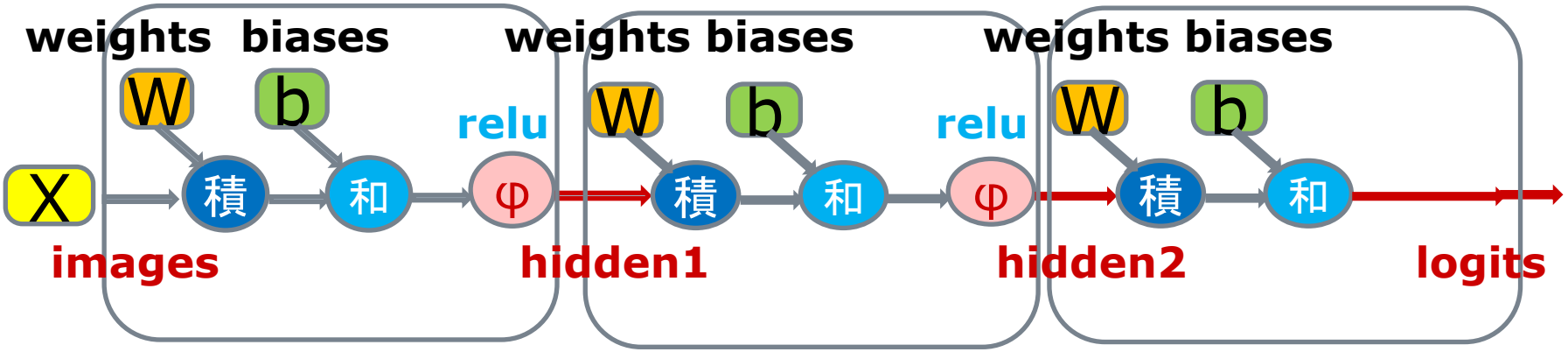
現在のMulti-Layer Perceptronの次のような構成と比較せよ。

入力層の
入力:
784
784次元

Hidden1層の
ニューロン数:128
重み W:
784x128次元
バイアス b:
128次元

Hidden 2層の
ニューロン数:32
重み W:
128x32次元
バイアス b:
32次元

Linear層の
ニューロン数:10
重み W:
32x10次元
バイアス b:
10次元



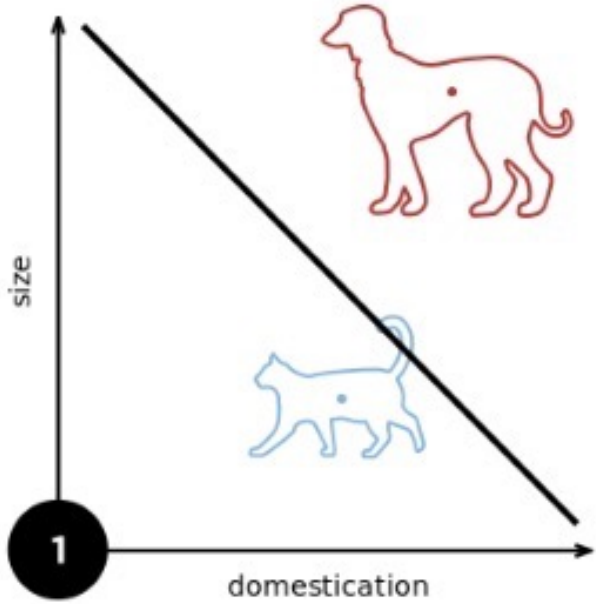
入力層　Hidden 1層　Hidden 2層　Linear層

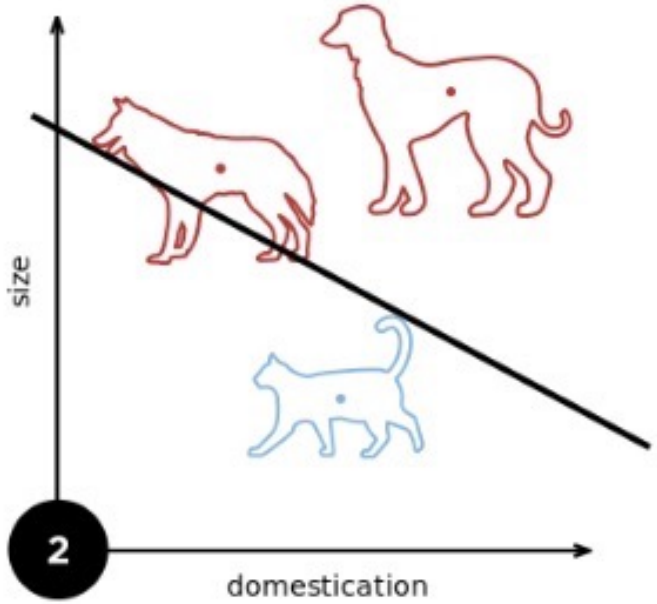
Minsky & Papert “Perceptron” 1969年

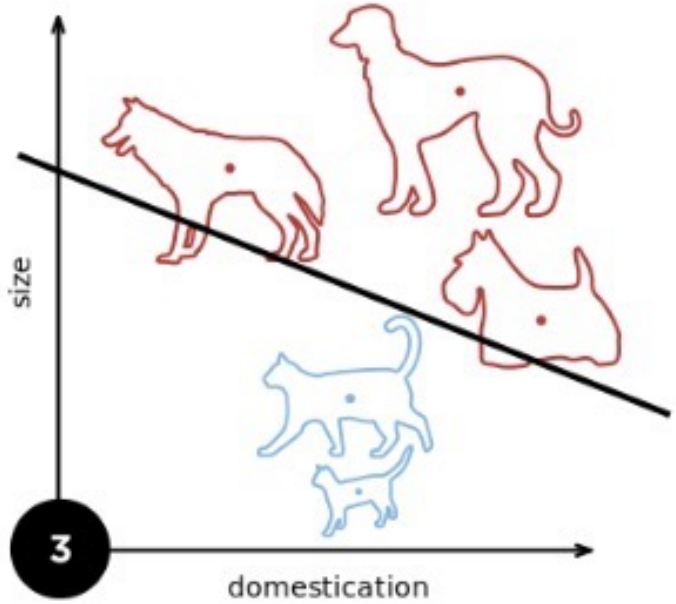
Single layer perceptrons are only capable of learning linearly separable patterns;

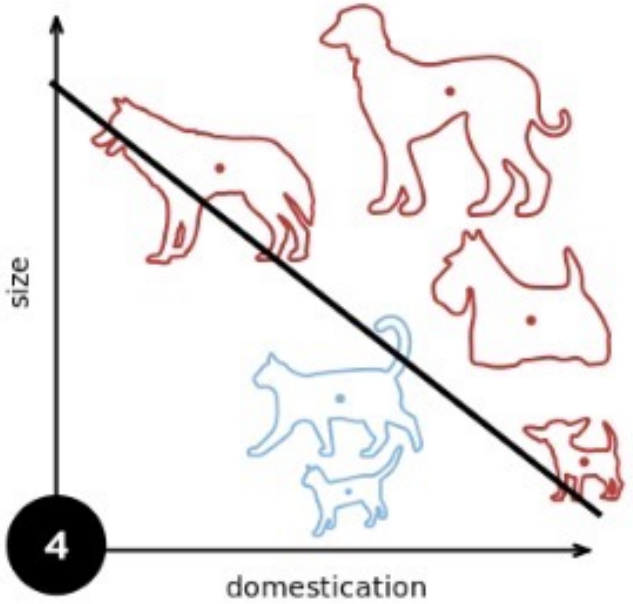
in 1969 a famous book entitled Perceptrons by Marvin Minsky and Seymour Papert showed that it was impossible for these classes of network to learn an XOR function. It is often believed that they also conjectured (incorrectly) that a similar result would hold for a multi-layer perceptron network. However, this is not true, as both Minsky and Papert already knew that **multi-layer perceptrons** were capable of producing an XOR function.

https://goo.gl/py6sNA


size
1
domestication


size
2
domestication

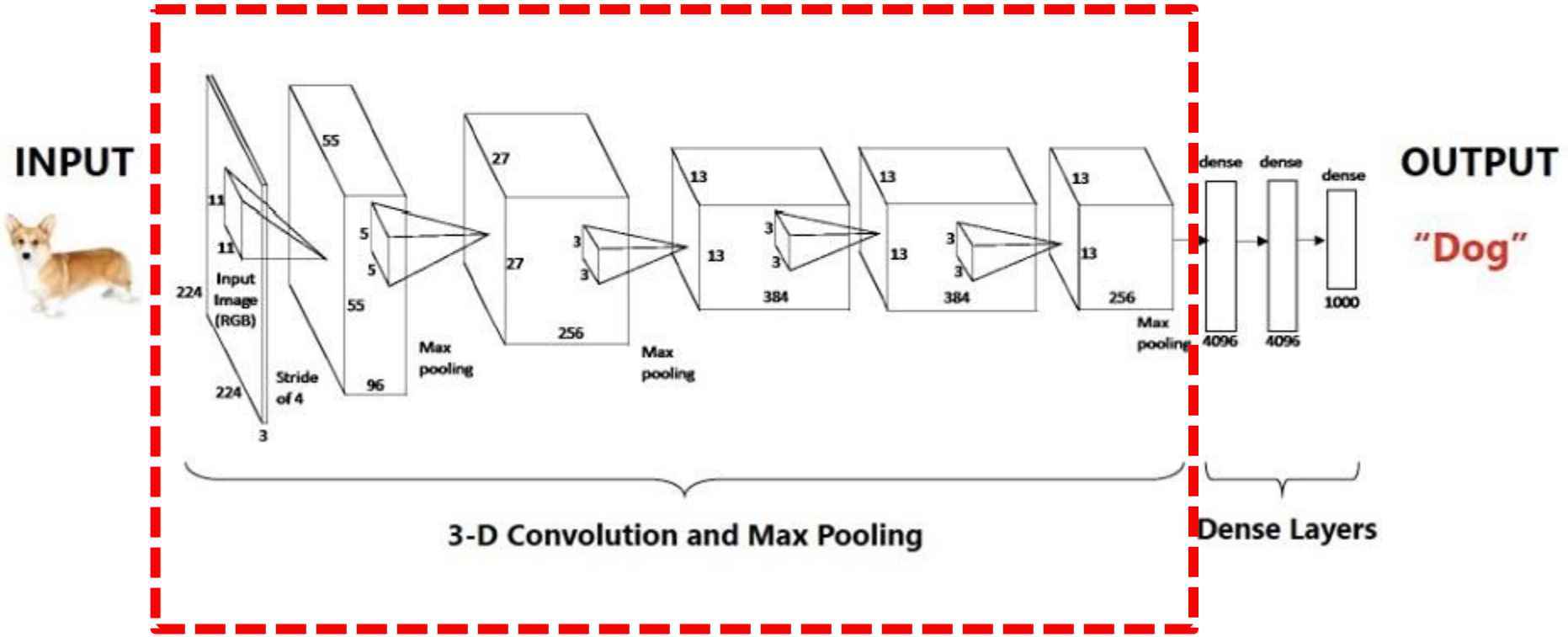

size
3
domestication


size
4
domestication

CNN
Convolutional Neural Network

マルレク「Convolutional Neural Network入門」

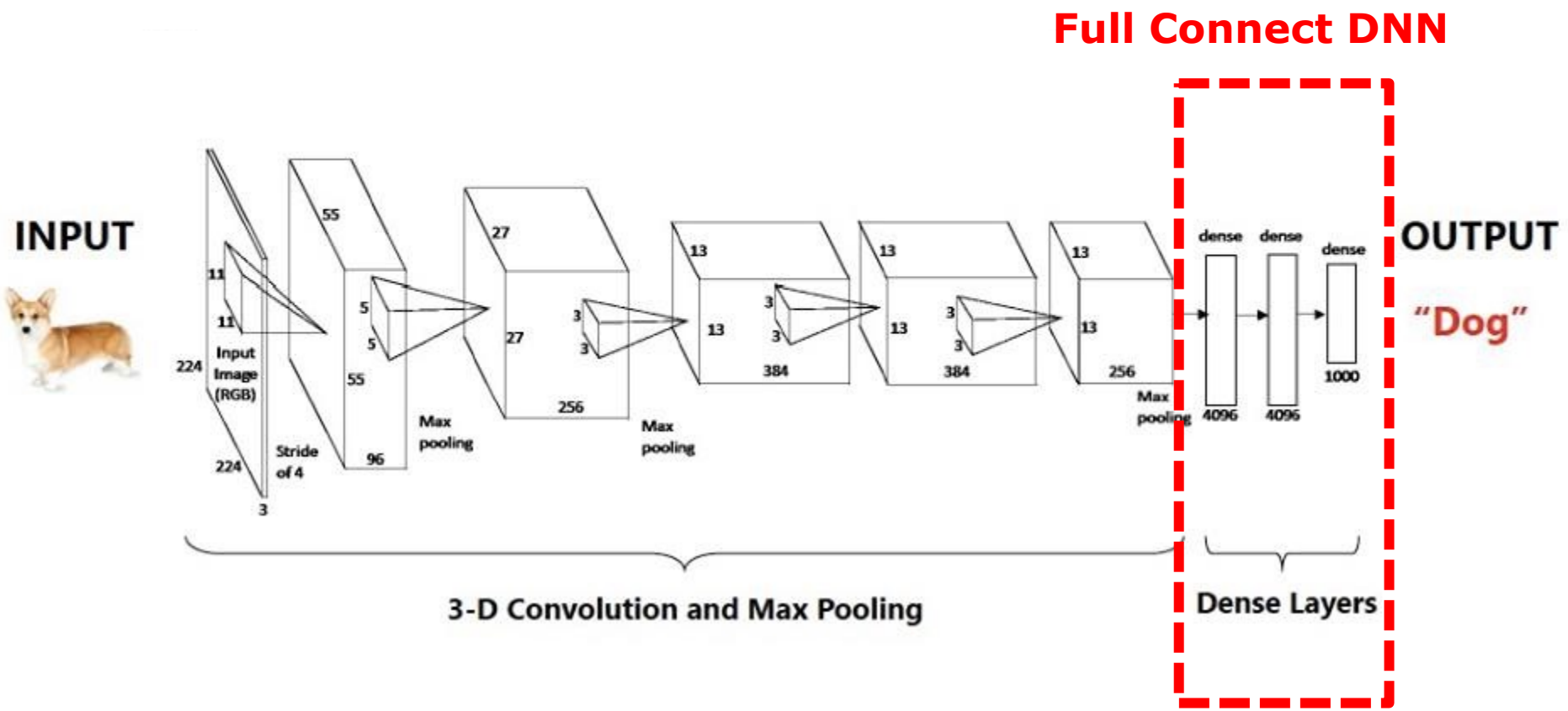
https://www.marulabo.net/docs/20160513-marulec01/



この講演が扱う範囲

マルレク「Convolutional Neural Network入門」

https://www.marulabo.net/docs/20160513-marulec01/



Convolutional Neural Networks

http://cs231n.stanford.edu/

Convolutional Neural Networksは、前章で見た通常のニューラル・ネットワークと、とてもよく似ている。それは、学習可能な重みとバイアスを持つニューロンから構成されている。それぞれのニューロンは、入力を受け取り、内積を計算して、ある場合には、それに非線形の関数が続く。

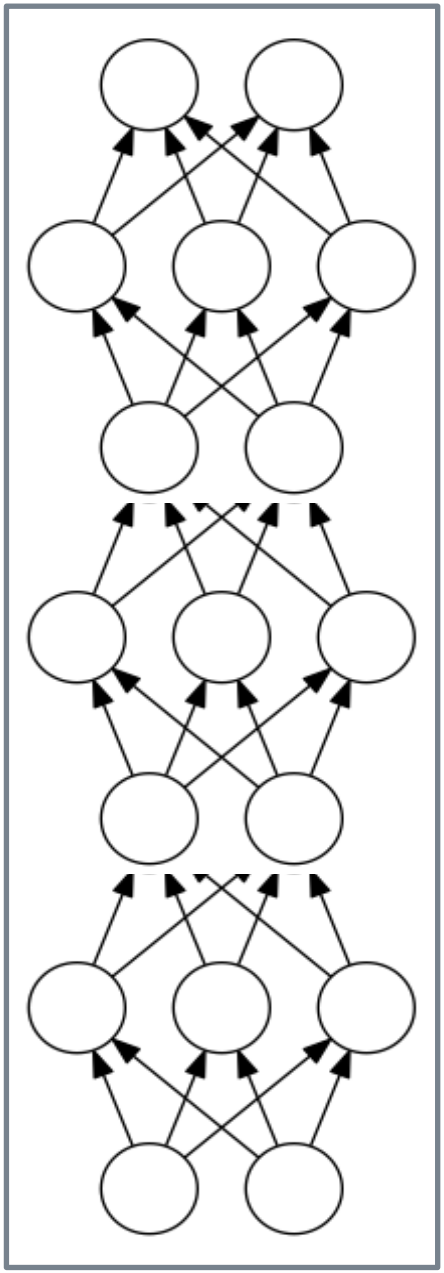
ネットワーク全体は、ここでも、一方の端の、生の画像のピクセルから、他方の端のクラスのスコアまで、一つの微分可能なスコア関数を表現している。

さらにCNNにも、最後の層（Full Connect）には、（SVM/Softmaxのような）損失関数もある。

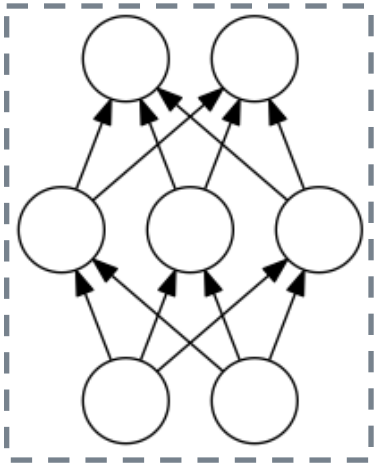
通常のニューラル・ネットワークのために開発してきた、手法やノウハウは、全て、CNNにも適用できる。

RNN
Recurrent Neural Network

「RNN と LSTMの基礎 -- Sequence to Sequence --」
https://www.marulabo.net/docs/20170222-marulec05/



縦方向に
並べる

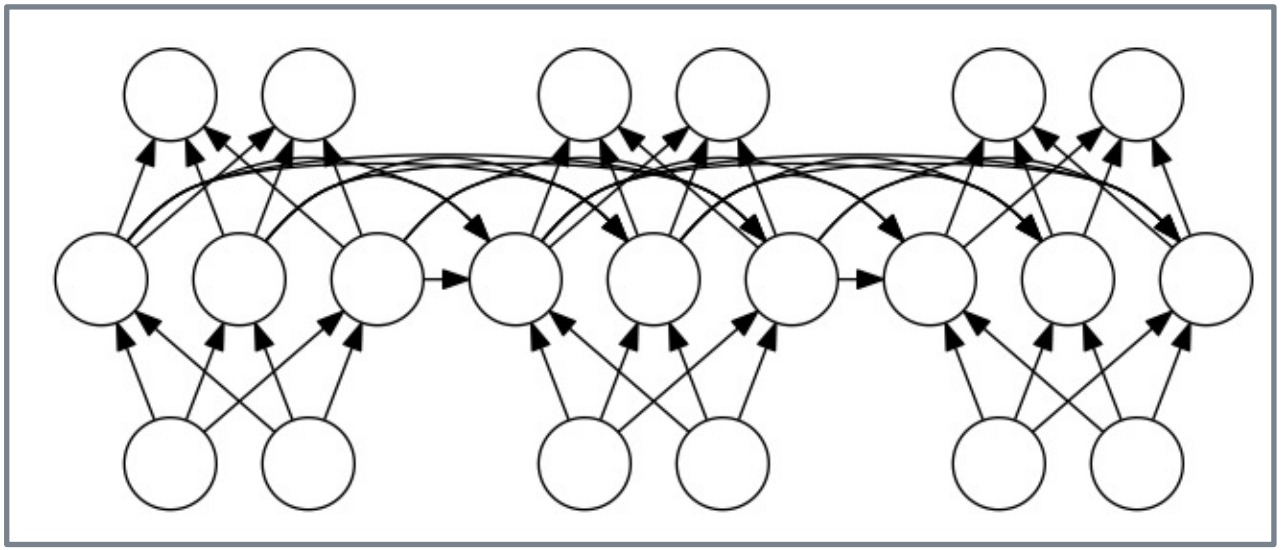


単純なネットワーク

Full Connect
DNN

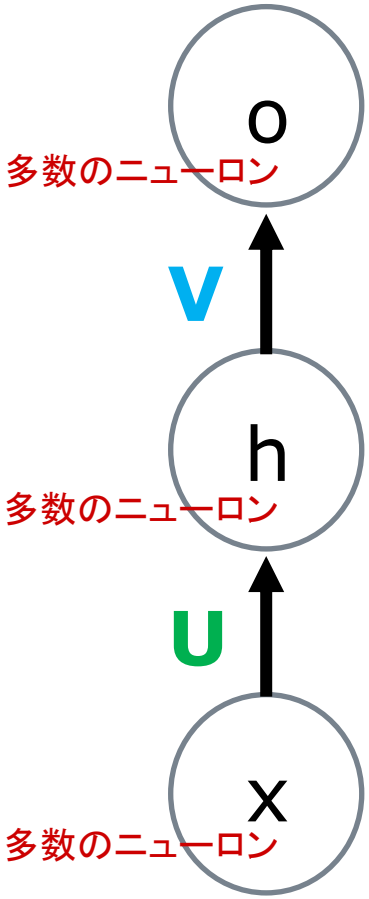
横方向に
並べる

RNN



単純な2層のDNNでの
$\varphi\,(\,WX + b\,)$ の復習

φ：活性化関数
W：重み
b：バイアス
X：入力



$$o = \varphi_o(\,Vh + b_o\,)$$

o ：出力層のベクトル
φ_o：出力層の活性化関数
V ：出力層の重み
b_o：出力層のバイアス
h ：隠れ層のベクトル

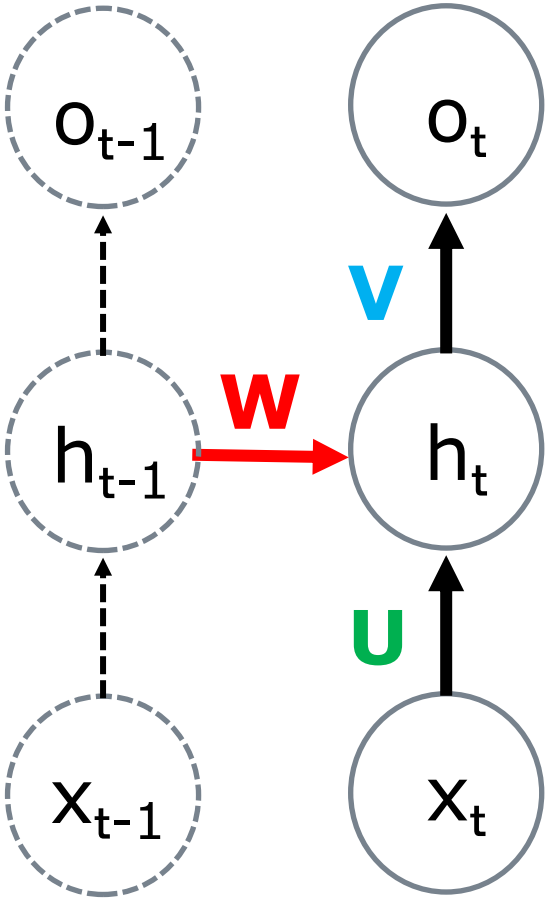
$$h = \varphi_h(\,Ux + b_h\,)$$

h ：隠れ層のベクトル
φ_h：隠れ層の活性化関数
U ：隠れ層の重み
b_h：隠れ層のバイアス
x ：入力層のベクトル

RNNのグラフ表記を式で表す

- 出力層の式は同じだが、隠れ層の式に、ちょっとした変化がある。それは、RNNの隠れ層が、入力層からだけでなく、隣の隠れ層からの入力も受けるので当然のことである。
- 次の図で、赤い字で書かれた部分が、それである。
- RNNでは、隣り合う隠れ層は、フルコネクトでつながっている。隠れ層同士のつながりの重みを***W***とすれば、隣の隠れ層 ***h(t-1)***から受け取るベクトルは、***W***と***h(t-1)***をかけたものになる。それは、隠れ層が、重み***U*** でフルコネクトでつながっている入力層 ***x(t)***から受け取るベクトルが、***U***と***x(t)*** をかけたものになるのと同じことである。
- こうして、RNNの隠れ層が外から受け取るベクトルは、入力層からくるベクトルに、隠れ層からくるベクトルを足したものだと考えればいいことになる。

RNN(展開形)での
$\varphi (WX + b)$ の応用



$$o_t = \varphi_o(Vh_t + b_o)$$

$$h_t = \varphi_h(Ux_t + Wh_{t-1} + b_h)$$

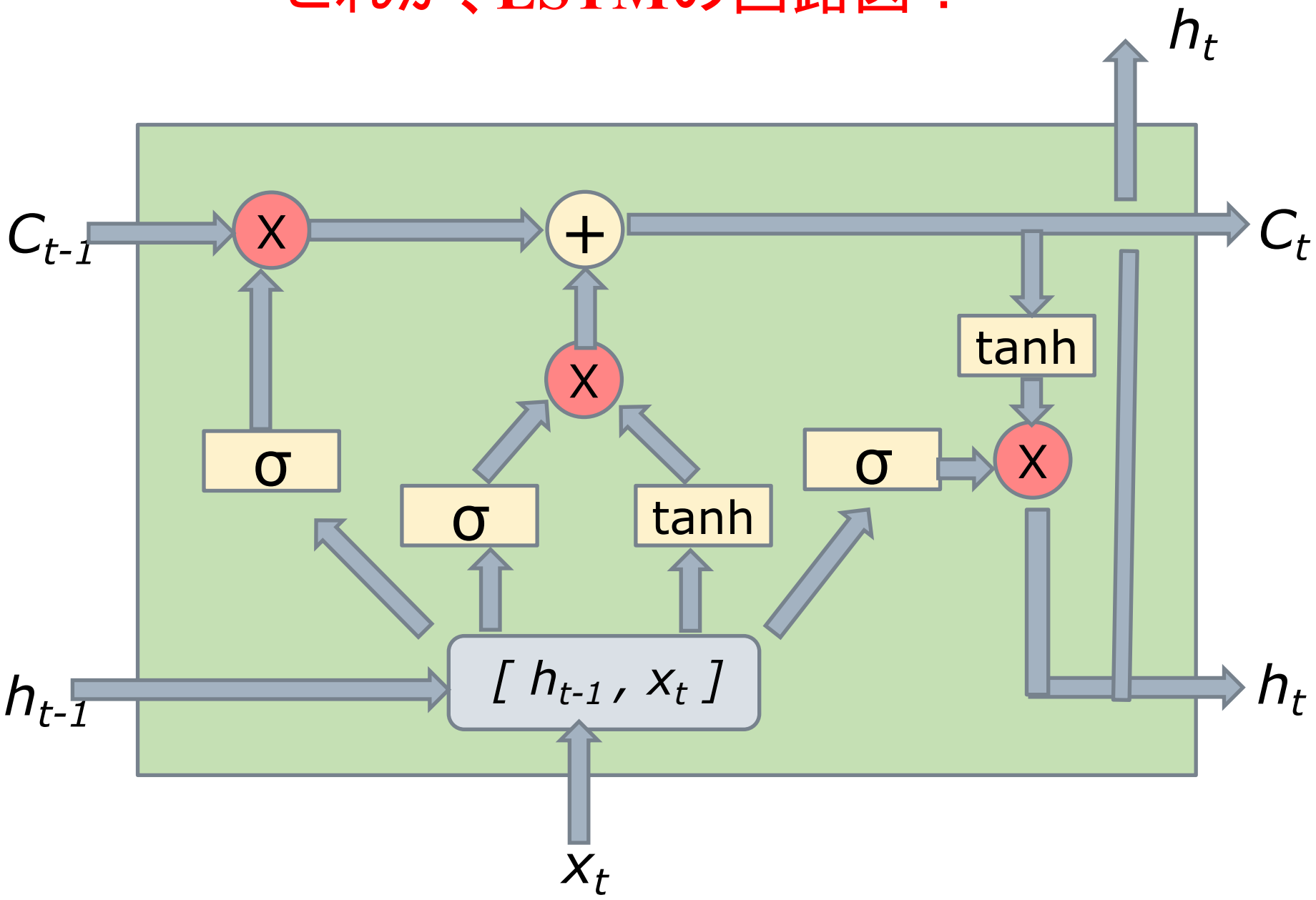
新たに追加された項。
隣の隠れ層の寄与分。
もちろん、重みxベクトル
の形をしている。

LSTM
RNNの復活

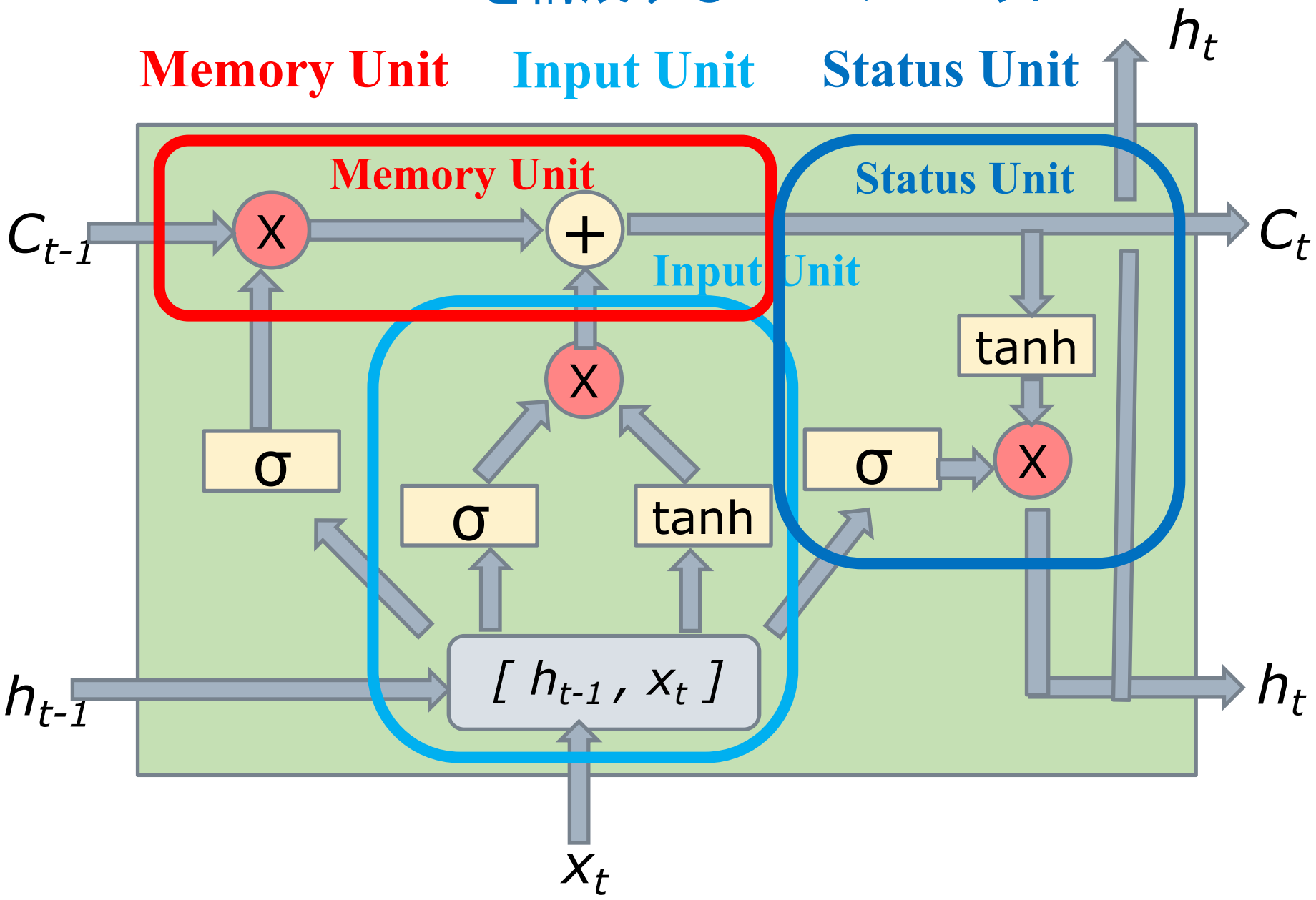
RNNは、その“Sequence to Sequence”の能力を生かして、現在は、Deep Learningの主役の一人に返り咲いている。では、先に見たRNNの抱えていた困難は、どのように克服されたのであろうか？

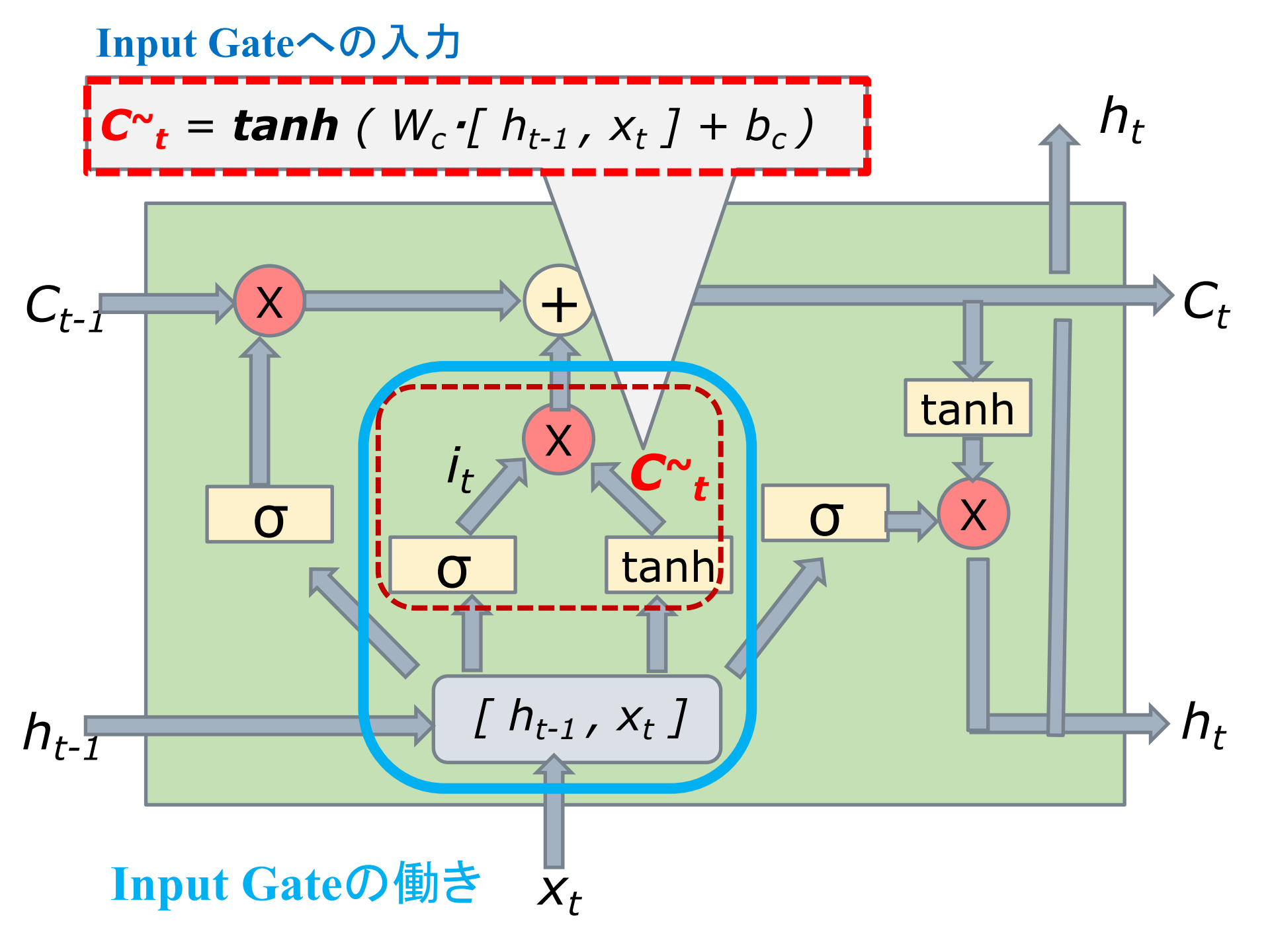
「RNN と LSTMの基礎 -- Sequence to Sequence --」
https://www.marulabo.net/docs/20170222-marulec05/

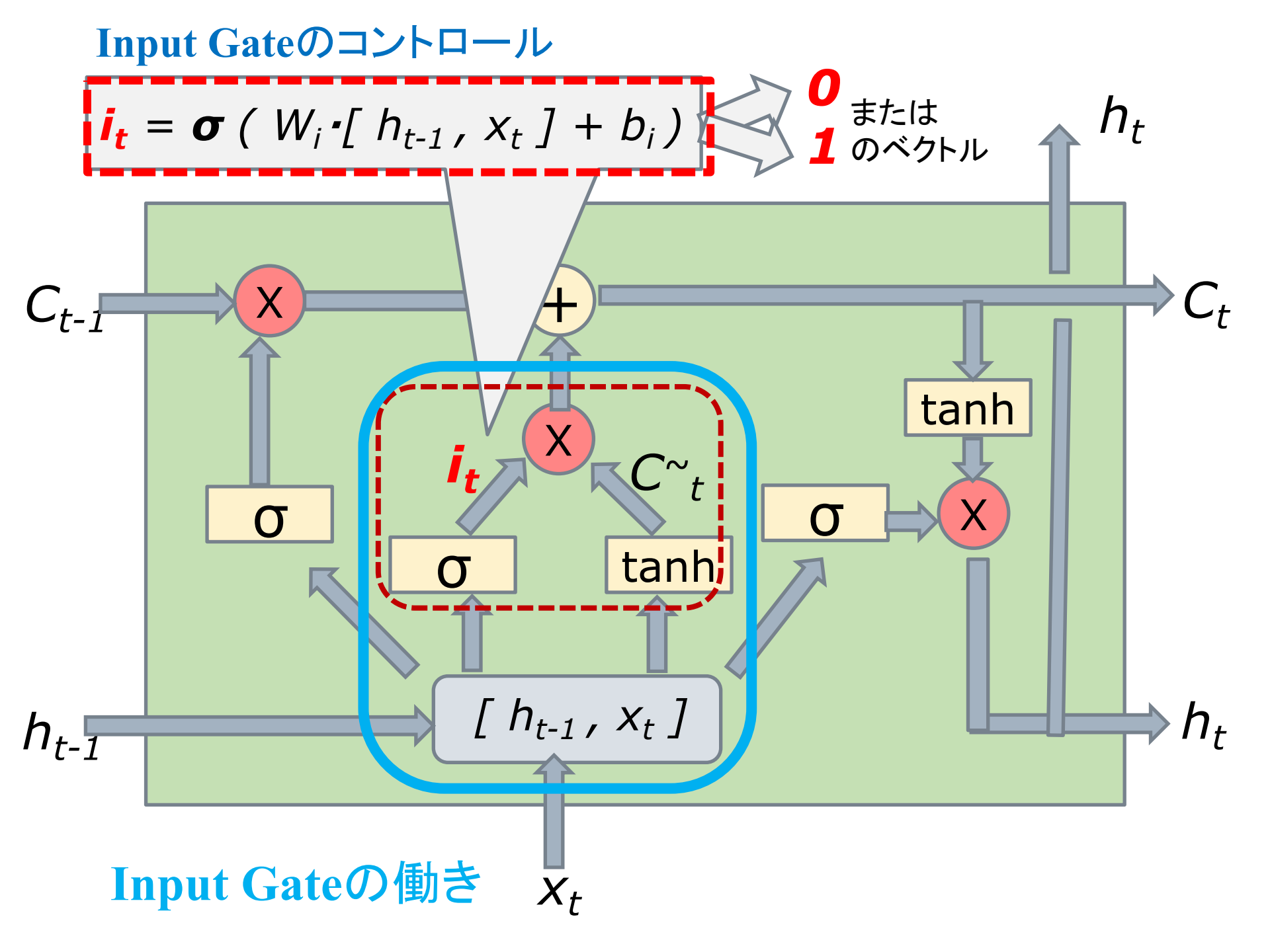
これが、LSTMの回路図！

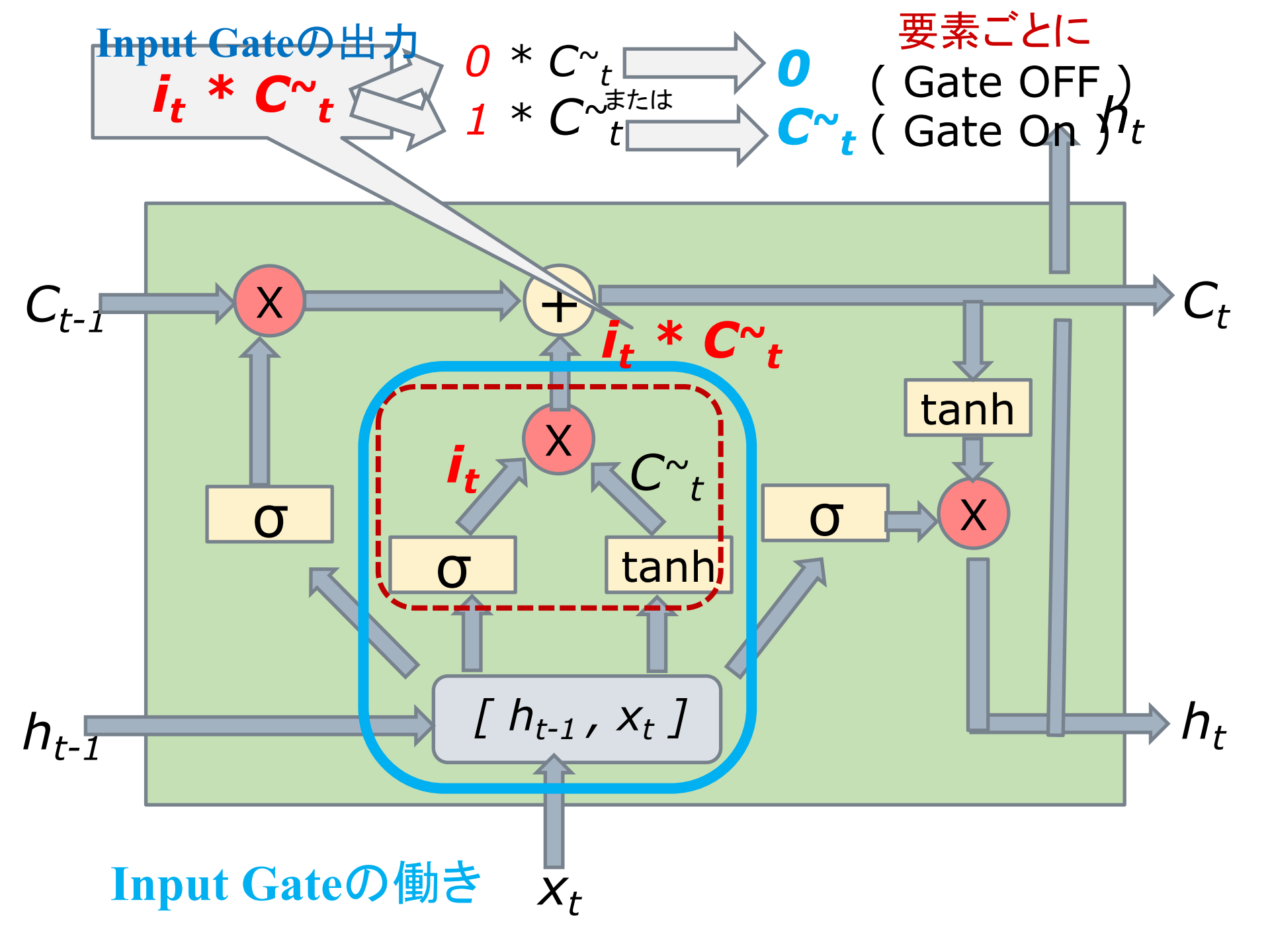


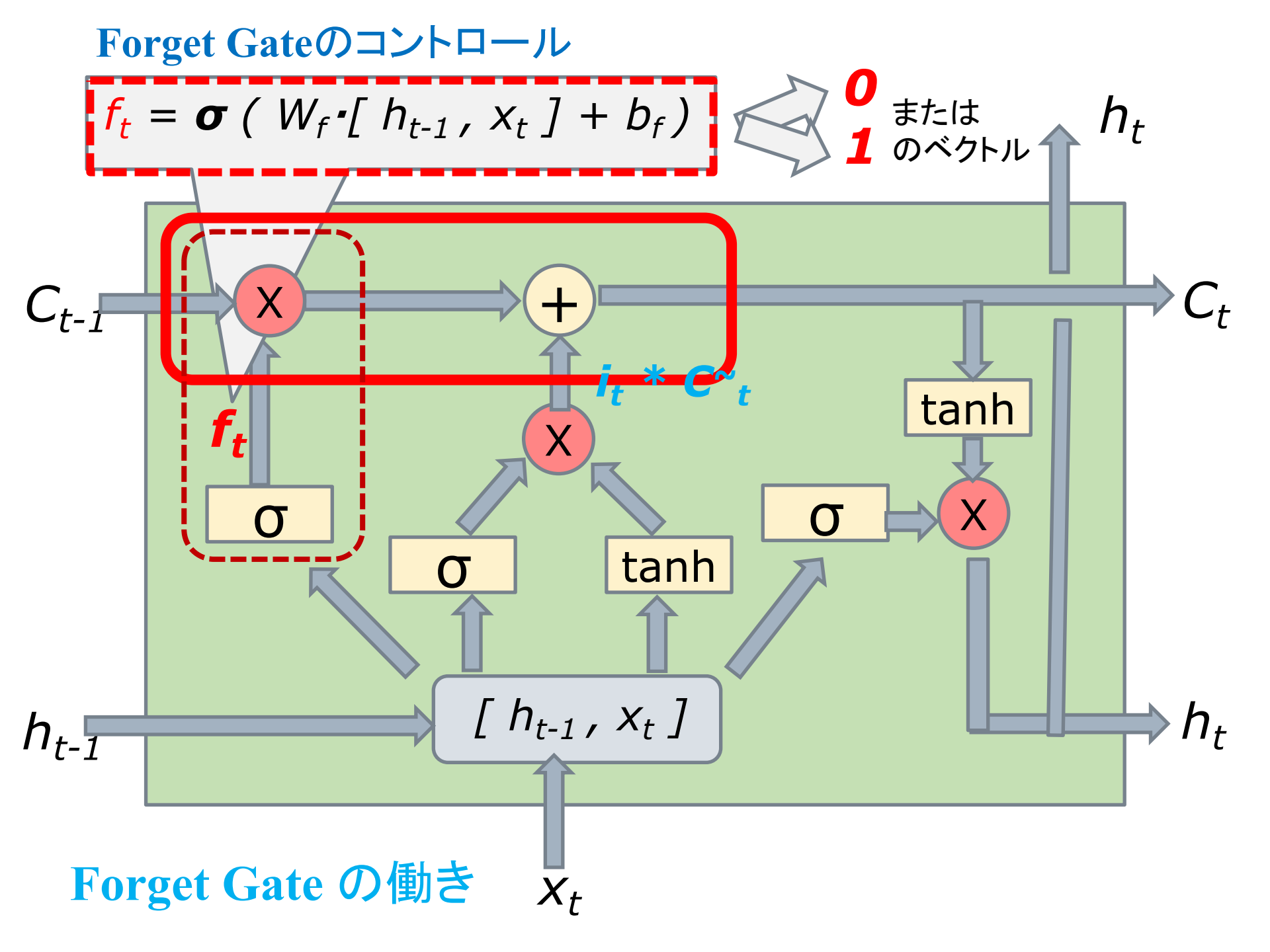
LSTMを構成する三つのユニット




Input Gateへの入力
$\tilde{C}_t = \tanh(W_c \cdot [h_{t-1}, x_t] + b_c)$
h_t
C_{t-1}
X
+
C_t
tanh
X
i_t
$\tilde{C}_t$
σ
σ
tanh
σ
X
$[h_{t-1}, x_t]$
h_{t-1}
h_t
x_t
Input Gateの働き


Input Gateのコントロール
$i_t = \sigma (W_i \cdot [h_{t-1} , x_t] + b_i)$
0
1
または
のベクトル
h_t
C_{t-1}
C_t
X
+
tanh
i_t
X
$C\tilde{}_t$
σ
σ
tanh
σ
X
$[h_{t-1} , x_t]$
h_{t-1}
h_t
x_t
Input Gateの働き


Input Gateの出力
$i_t * C\tilde{}_t$
$0 * C\tilde{}_t$
$1 * C\tilde{}_t$
または
0
$C\tilde{}_t$
要素ごとに
(Gate OFF)
(Gate On)
h_t
C_{t-1}
X
+
$i_t * C\tilde{}_t$
C_t
tanh
X
i_t
$C\tilde{}_t$
σ
σ
tanh
σ
X
h_{t-1}
[h_{t-1} , x_t]
h_t
x_t
Input Gateの働き

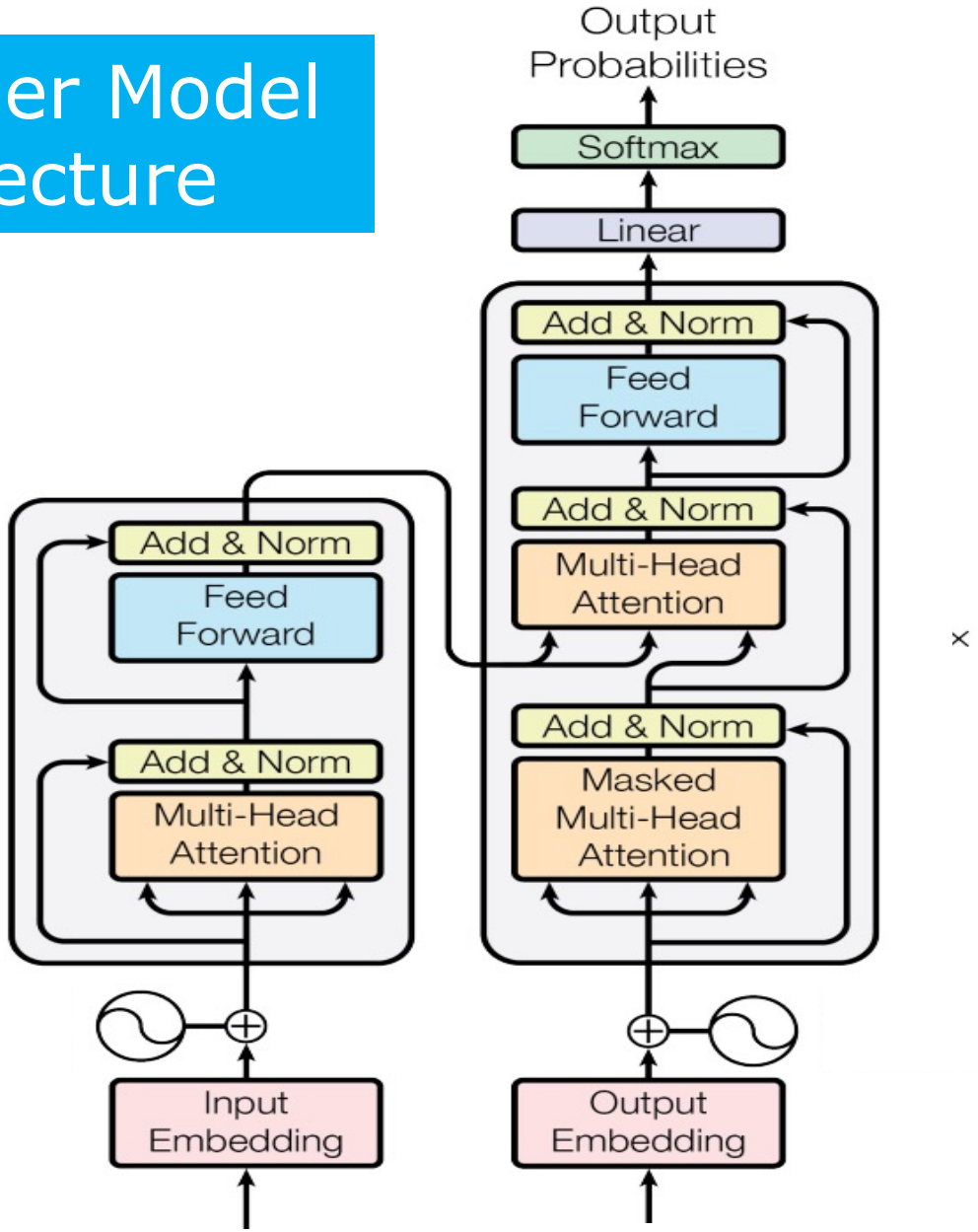

Forget Gateのコントロール
$f_t = \sigma (W_f \cdot [h_{t-1} , x_t] + b_f)$
0
1
またはのベクトル
h_t
C_{t-1}
C_t
X
+
$i_t * \tilde{C}_t$
f_t
X
tanh
σ
σ
tanh
σ
X
$[h_{t-1} , x_t]$
h_{t-1}
h_t
x_t
Forget Gate の働き

LSTM の働きを式で表す

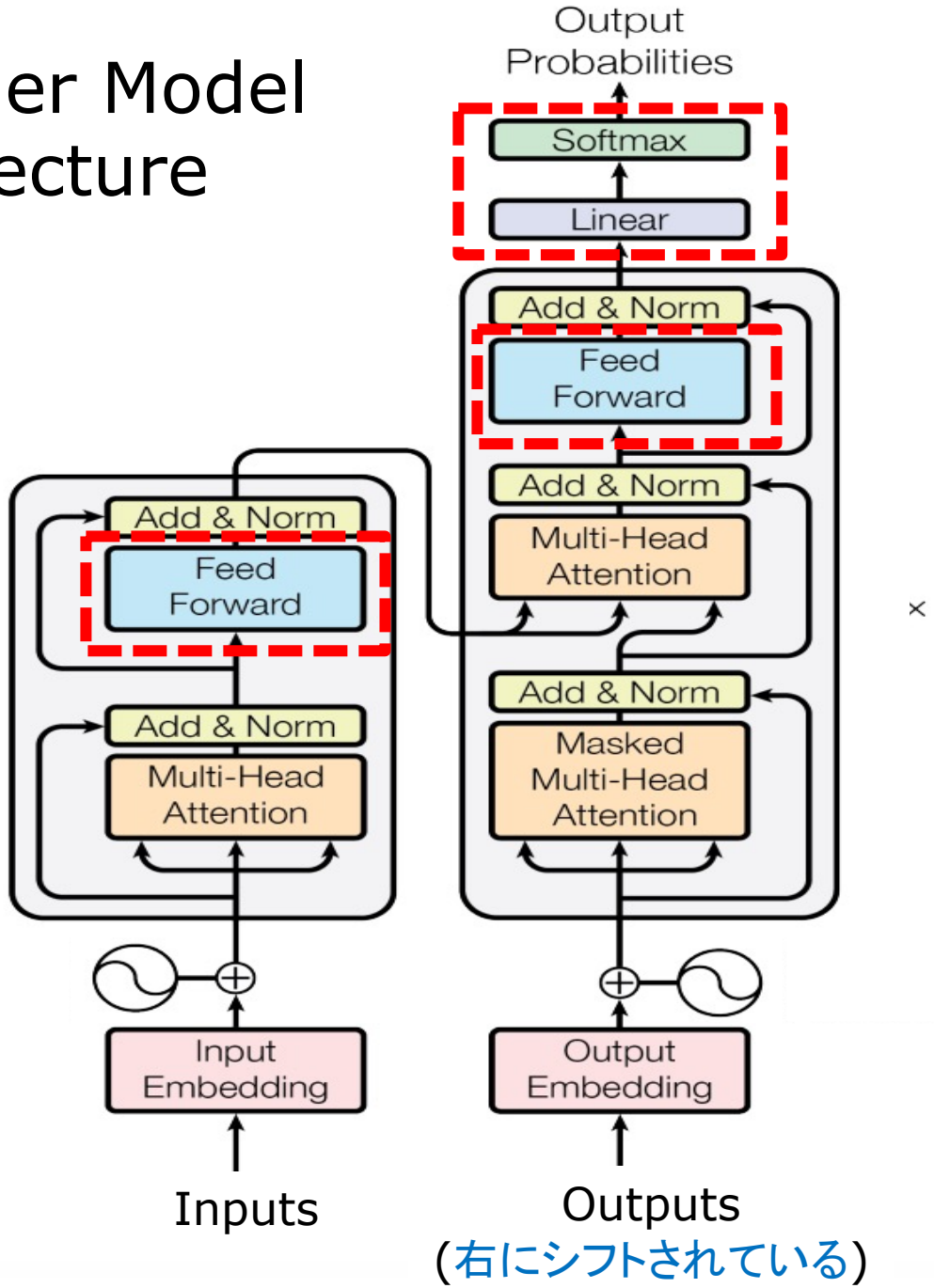
- **Input Gateのコントロール:**
  $$i_t = \sigma (W_i \cdot [h_{t-1} , x_t] + b_i)$$
- **Forget Gateのコントロール:**
  $$f_t = \sigma (W_f \cdot [h_{t-1} , x_t] + b_f)$$
- **Output Gateのコントロール:**
  $$o_t = \sigma (W_o \cdot [h_{t-1} , x_t] + b_o)$$

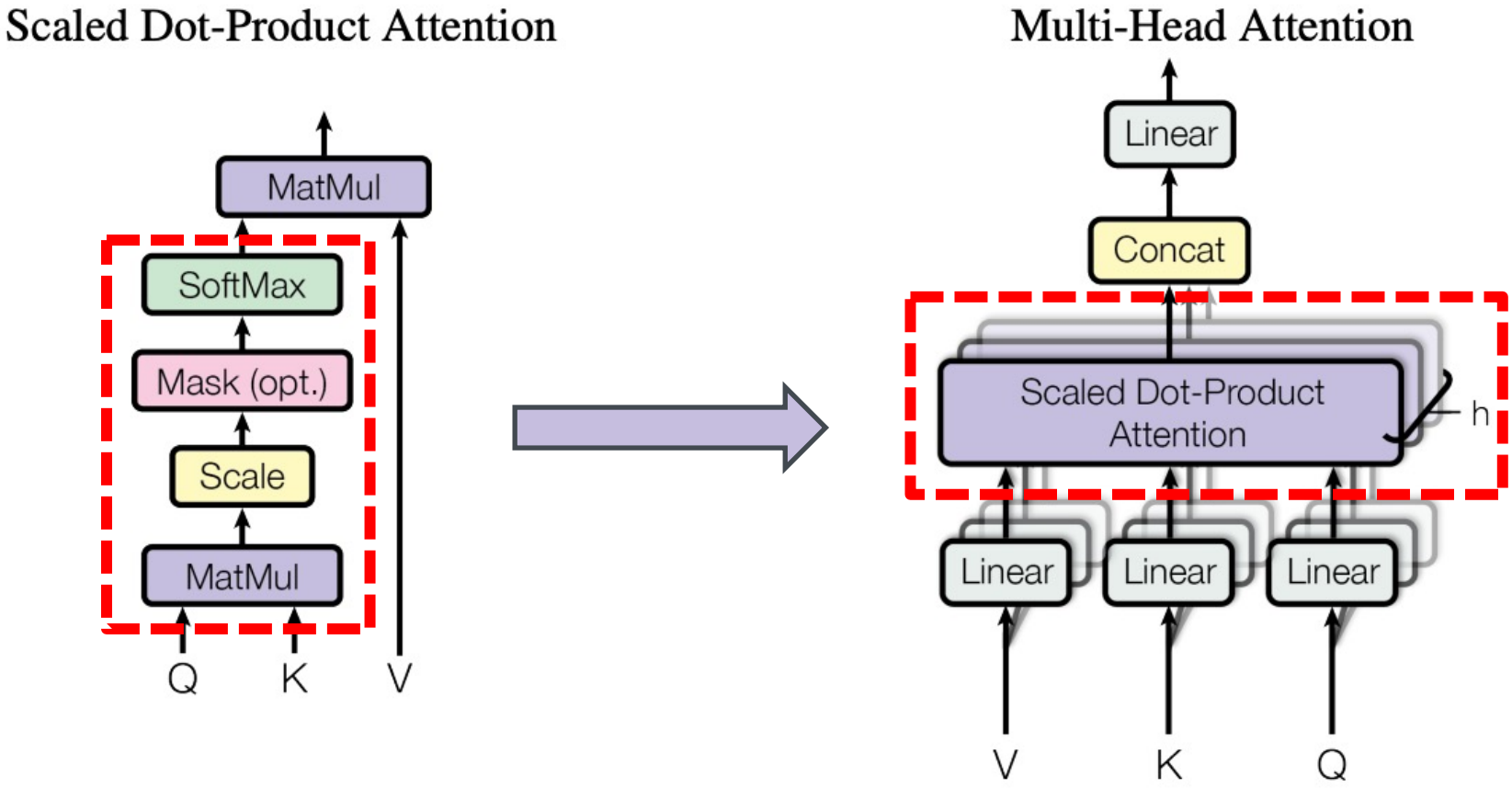
Transformer

Transformer Model Architecture



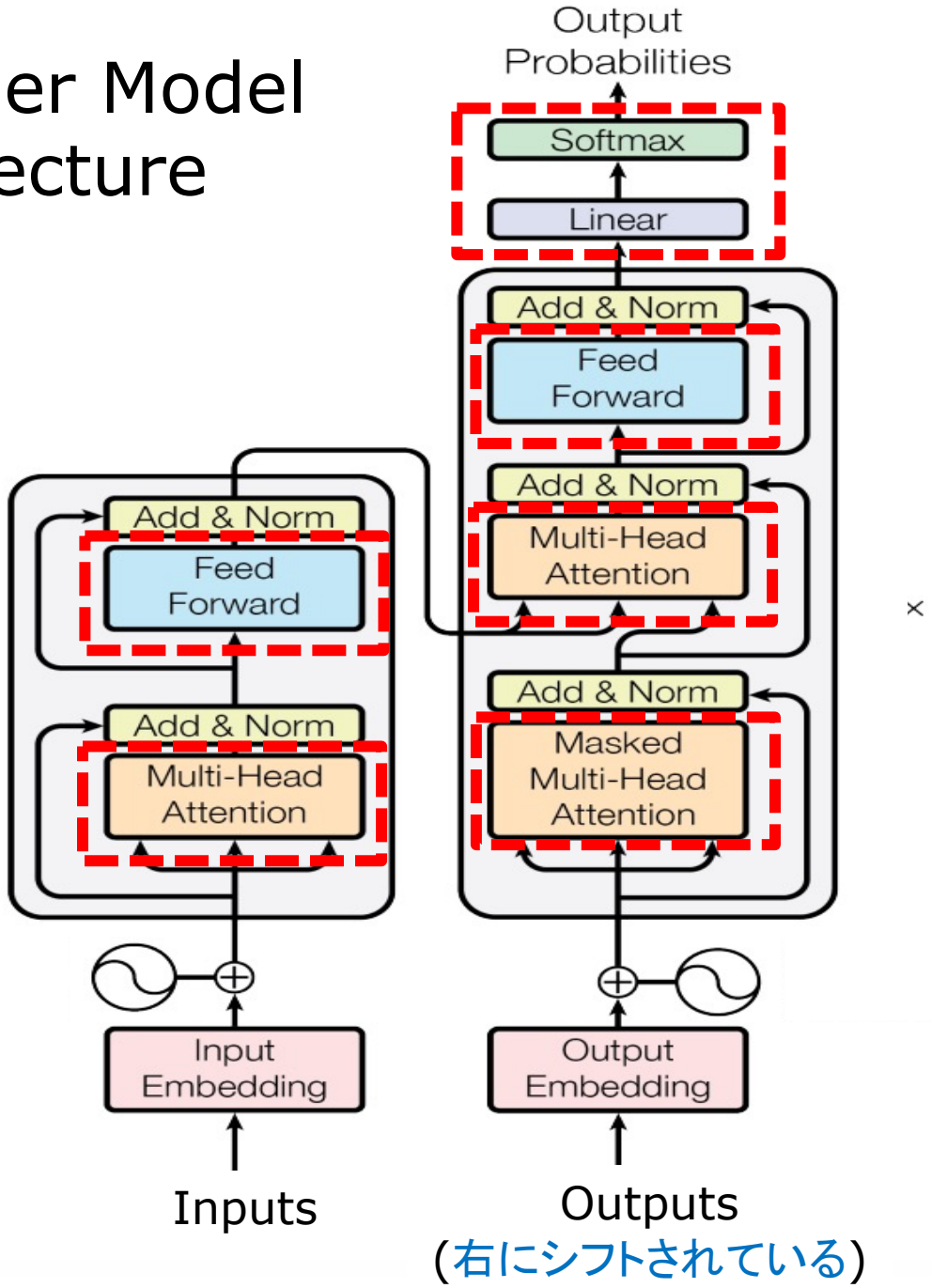
Transformer Model Architecture



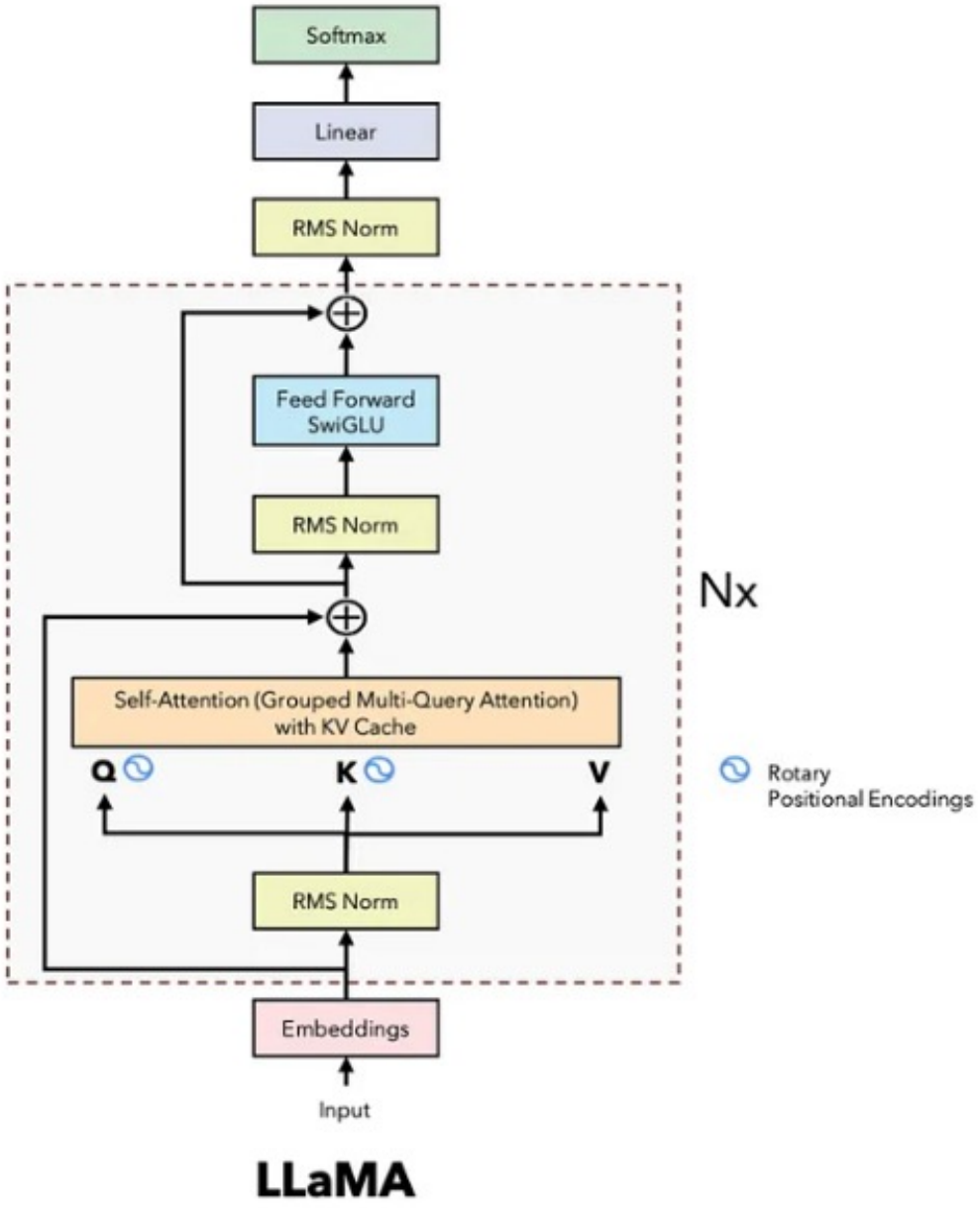


Attention Is All You Need
https://arxiv.org/pdf/1706.03762.pdf

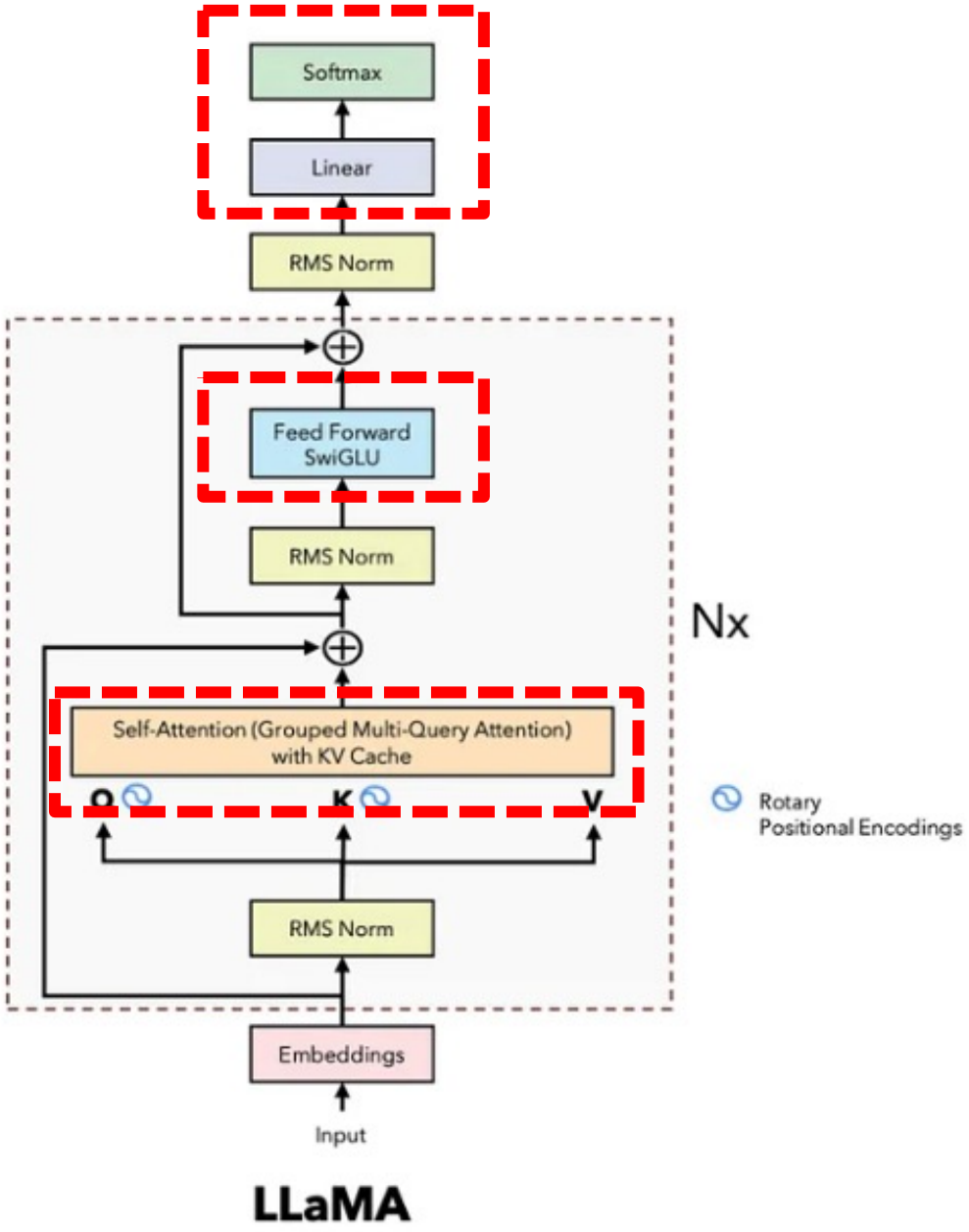
Transformer Model Architecture



LLaMa2 Transformer Architecture



LLaMa2 Transformer Architecture




Part 2
Tropical数学入門
ニューラル・ネットワークの数理 -- Tropical代数入門

Part 2　Tropical数学入門

- 加算と乗算
- ベクトル・行列計算
- 多項式
- Hypersurface
- 二変数のTropical多項式のHypersurface
- Newton図形


Tropical数学入門 (1)
加算と乗算
ニューラル・ネットワークの数理 -- Tropical数学入門

The adjective "tropical" was coined by French mathematicians, notably Jean-Eric Pin, to honor their Brazilian colleague Imre Simon, who pioneered the use of min-plus algebra in optimization theory.

There is no deeper meaning in the adjective "tropical". It simply stands for the French view of Brazil.

--- Diane Maclagan

1992年

Introduction to Linear Categories and Applications

F. William Lawvere

the scalar quantities are all nonnegative real numbers, including ∞ ; but the addition and multiplication are defined by

$a + b \underset{def}{=}$ the minimum of a,b

$a \cdot b \underset{def}{=}$ the usual sum of a,b

Tropicalな数学と普通の数学

これから、Tropicalな数学の話をしようと思います。Tropicalな数学の世界は、普通の数学の世界とは、ちょっと変わった世界です。

ただ、両方の数学の世界は、とても似ているところがあります。

- まず、両方の世界には、「数」が存在します。
- また、両方の世界には、「数」同士の「足し算」と「掛け算」といった計算が定義されています。
- 計算が定義されているということは、ある数どうしの計算の結果がある数に等しいという「等式」が定義されているということです。等式の扱いや、論理的な推論の仕方は、両方の世界で共通です。

両方の数学の世界の最大の共通性は
両方の世界に、同じ数が存在すること

こうした二つの世界の類似で、一番重要なことは、具体的な「数」については、両方の世界には、同じ数が存在するということです。

例えば、我々の普通の世界には、2024 という整数が存在するのですが、Tropicalな世界にも、整数 2024 が存在します。

整数だけではありません。我々の数の世界には、実数も複素数も存在するのですが、これらはすべてTropicalの数の世界にも存在します。二つの世界の数は、一致していて、同じものです。

両方の数学の世界の最大の違いは
足し算と掛け算の定義が異なること

両方の数学の世界の最大の違いは、足し算と掛け算の定義が異なることです。

普通の計算とTropicalな計算を区別するために、Tropicalな掛け算と足し算を⊙と⊕という記号で表すことにします。

掛け算と足し算の定義は、次のようになります。

$$x \odot y = x + y$$

$$x \oplus y = \min(x, y)$$

あるいは

$$x \oplus y = \max(x, y)$$

Tropical数学の世界は、二種類ある

Tropical数学の足し算⊕の定義が二つ出てきたのですが、実はTropical数学の世界は、足し算が min (最小値)で定義される世界と、足し算が max (最大値)で定義される世界の二種類あるのです。両方の世界で、掛け算の定義（plus で定義される）は同じです。

前者をTropical数学 min plus、後者をTropical 数学 max plus と呼びます。Tropical数学という場合、二つのうちどちらの定義を取ったのかを明確にする必要があります。

今回のセッションでは、min plus で、サンプルを説明したいと思います。

二つの世界は対応している
対応は、普通の世界で解釈される

Tropical数学の世界と、普通の数学の世界は、同じ「数」を共有することで、対応しています。

同じ数字が二つの世界で対応するというのは、わかりやすいのですが、両者の計算の仕方が異なるので、両者の対応関係をイメージするには、最初は、少し戸惑うかもしれません。

ただ、計算での対応をみるのにはポイントがあります。それは、両者の計算での対応関係は、我々がよく知っている普通の世界での計算にもどして考えることが基本になるということです。

もう少し、具体的に（まわりくどく）説明しましょう。

先に、 $x \odot y = x + y$ で Tropicalな掛け算を定義したのですが、少し飛躍があります。なぜなら、左辺の $x \odot y$ は、Tropicalな世界で行われる計算で、右辺の $x + y$ は、普通の世界の普通の足し算です。両者は、計算が行われる世界が違うのです。

左辺と右辺が等しいという、この掛け算の定義は、次のことを意味しています。

- $x \odot y$ を、Tropicalな世界で計算するということは、Tropicalな世界に数 x, y が存在するということだから、普通の世界にも数x, y は存在して、それは同じものである。
- 普通の世界に移って $x + y$ を実行してその答えを、$x + y = z$ とする。
- zと同じzが、Tropical な世界にも存在する。
- Tropicalの世界に戻って、それを、$x \odot y = z$ とする。

計算練習
（記号が違うので、慣れると簡単である）

ちょっと計算を練習してみましょう。Toropicalな世界での計算はピンクで、普通の世界での計算はライトブルーで表しています。

$$4 \odot 5 = 4 + 5 = 9$$

$$3 \oplus 100 = \min(3, 100) = 3$$

$$7 \odot 5 \oplus 7 \odot 6$$
$$= (7 + 5) \oplus (7 + 6) = 12 \oplus 13 = \min(12,13) = 12$$

$$7 \odot (5 \oplus 6)$$
$$= 7 \odot \min(5, 6) = 7 \odot 5 = 7 + 5 = 12$$

Tropical数学での加算と乗算の可換性

加算と乗算の可換性とは、普通の世界では、$x + y = y + x, xy = yx$ であることを言います。

$x \oplus \mathrm{y} = \mathrm{y} \oplus \mathrm{x}$ の証明。

$$x \oplus \mathrm{y} = \min(\mathrm{x}, \mathrm{y}) = \min(\mathrm{y}, \mathrm{x}) = \mathrm{y} \oplus \mathrm{x}$$

$x \odot y = y \odot x$ の証明

$$x \odot y = x + y = y + x = y \odot x$$

Tropical数学での加算と乗算の分配律

加算と乗算の分配律とは、普通の世界では$x(y+z)=xy+xz$であることを言います。

$x \odot (\mathrm{y} \oplus \mathrm{z}) = x \odot y \oplus \mathrm{x} \odot y$ の証明。

$$
\begin{aligned}
x \odot (\mathrm{y} \oplus \mathrm{z}) &= \mathrm{x} + \min(\mathrm{y}, \mathrm{z}) \\
&= \min(\mathrm{x}+\mathrm{y}, \mathrm{x}+\mathrm{z}) \\
&= x \odot y \oplus \mathrm{x} \odot y
\end{aligned}
$$

Tropical数学での加算の零元

普通の世界では、任意のxについて $x + 0 = x$ ですので、加算の零元は $\mathbf{0}$です。

Tropical数学で、任意のxについて、$x \oplus \mathrm{z} = \mathrm{x}$ すなわち、
$x \oplus \mathrm{z} = \min(\mathrm{x}, \mathrm{z}) = \mathrm{x}$ を満たすzは存在するでしょうか？

$$z = +\infty$$

とすると、どんなxも$+\infty$ よりは小さいので、Tropical数学での加算の零元は、$+\infty$ とおけばいいことがわかります。

Tropical数学では、「数」に$+\infty$を加えて、それを加算についての零元とします。

Tropical数学での乗算の単位元

普通の世界では、任意のxについて $x\times 1 = x$ ですので、乗算の単位元は 1です。

Tropical数学で、任意のxについて、$x \odot \mathrm{z} = \mathrm{x}$ すなわち、
$x \odot \mathrm{z} = \mathrm{x} + \mathrm{z} = \mathrm{x}$ を満たすzは存在するでしょうか？

少し意外ですが、$z = 0$ がその条件を満たすことがわかります。

Tropical数学での乗算の単位元は、0になります。

$$x \odot 0 = 0 \odot x = x$$

が成り立ちます。

Tropical数学では、引き算は定義されない！

Tropical数学の、最も奇妙な性質の一つは、そこでは「引き算」が定義されないことです。

普通の数学の世界では、$x + 4 = 13$ から、$x = 13 - 4 = 7$ が、引き算で簡単に求めることができるのですが、Tropical数学の

$$x \oplus 4 = 13$$

は解を持ちません。

なぜなら、$x \oplus 4 = \min(x, 4) = 13$ となるようなxは存在しないからです。


Tropical数学入門 (2)
ベクトル・行列計算
ニューラル・ネットワークの数理 -- Tropical数学入門

Tropicalなベクトル・行列計算

このセッションでは、Tropicalなベクトル・行列の計算を見ていきます。

ただ、前回のセッションで、すこし積み残したことがあるので、最初にそれを補足しておきます。

前回のセッションの補足

前回のセッションで、Tropical数学では、

- ＋∞をが加算についての零元になり、
  $a \oplus \infty = \infty \oplus a = a$ が成り立つ。
- 0が乗算についての単位元になり、
  $a \otimes 0 = 0 \otimes a = a$ が成り立つ

ことを見てきました。

この ∞ と 0 について次の式が成り立ちます。

$$a \otimes \infty = a + \infty = \infty$$

$a \oplus 0 = \min(a, 0)$ ですので、

この値は、$a \geq 0$の時は0に、$a < 0$の時はa になります。

Tropicalな割り算

Tropical数学では、加算 $\oplus$ 、乗算 $\otimes$ にくわえて、除算が定義されます。記号は $\oslash$ を使います。

定義は、次のように与えられます。

$$a \oslash \mathrm{b} = \mathrm{a} - \mathrm{b}$$

この定義が妥当なことは、次のように考えるとわかります。

$b \otimes x = a$ を満たすxを求めよという問題を考えます。

この時、$x = a \oslash \mathrm{b}$ が答えになることは、次で確かめられます。

$$b \otimes x = b \otimes (a \oslash \mathrm{b}) = \mathrm{b} + (a \oslash \mathrm{b})$$
$$= b + (a - b) = a$$

Tropicalなベクトル・行列計算

Tropicalな数について、基本的な加算・乗算が定義されると、
Tropicalな数から構成される、Tropicalなベクトルや行列への
Tropicalな演算も、自然に定義されるようになります。

二つのベクトルの積

まず、二つのベクトルの内積について見てみましょう。

$$(u_1, u_2, u_3) \otimes (v_1, v_2. v_3)^T$$

$$= u_1 \otimes v_1 \oplus u_2 \otimes v_2 \oplus u_3 \otimes v_3$$
$$= \min(u_1 \otimes v_1, u_2 \otimes v_2, u_3 \otimes v_3)$$

$$= \min(u_1 + v_1, u_2 + v_2, u_3 + v_3)$$

もう一つの、行列を形成する二つのベクトルの積は、こうなります。

$$(u_1, u_2, u_3)^T \otimes (v_1, v_2. v_3)$$

$$= \begin{pmatrix} u_1 \otimes v_1 & u_1 \otimes v_2 & u_1 \otimes v_3 \\ u_2 \otimes v_1 & u_2 \otimes v_2 & u_2 \otimes v_3 \\ u_3 \otimes v_1 & u_3 \otimes v_2 & u_3 \otimes v_3 \end{pmatrix}$$

$$= \begin{pmatrix} u_1 + v_1 & u_1 + v_2 & u_1 + v_3 \\ u_2 + v_1 & u_2 + v_2 & u_2 + v_3 \\ u_3 + v_1 & u_3 + v_2 & u_3 + v_3 \end{pmatrix}$$

いくつかの計算例

ベクトルのスカラー倍

$$
\begin{aligned}
&2 \otimes (3, -7, 6) \\
&= (2 \otimes 3, 2 \otimes -7, 2 \otimes 6) \\
&= (2 + 3, 2 + (-7), 2 + 6) \\
&= (5, -5, 8)
\end{aligned}
$$

ベクトルの内積

$$
\begin{aligned}
&(\infty, 0, 1) \otimes (0, 1, \infty)^T \\
&= (\infty \otimes 0 \oplus 0 \otimes 1 \oplus 1 \otimes \infty) \\
&= (\infty \oplus 1 \oplus \infty) \\
&= \min(\infty . 1, \infty) \\
&= 1
\end{aligned}
$$

行列の和

$$\begin{pmatrix} 3 & 3 \\ 0 & 7 \end{pmatrix} \oplus \begin{pmatrix} 4 & 1 \\ 5 & 2 \end{pmatrix}$$

$$= \begin{pmatrix} 3 \oplus 4 & 3 \oplus 1 \\ 0 \oplus 5 & 7 \oplus 2 \end{pmatrix}$$

$$= \begin{pmatrix} \min(3,4) & \min(3,1) \\ \min(0,5) & \min(7,2) \end{pmatrix}$$

$$= \begin{pmatrix} 3 & 1 \\ 0 & 2 \end{pmatrix}$$

行列の積

$$\begin{pmatrix} 3 & 3 \\ 0 & 7 \end{pmatrix} \otimes \begin{pmatrix} 4 & 1 \\ 5 & 2 \end{pmatrix}$$

$$= \begin{pmatrix} 3 \otimes 4 \oplus 3 \otimes 5 & 3 \otimes 1 \oplus 3 \otimes 2 \\ 0 \otimes 4 \oplus 7 \otimes 5 & 0 \otimes 1 \oplus 7 \otimes 2 \end{pmatrix}$$

$$= \begin{pmatrix} 7 \oplus 8 & 4 \oplus 5 \\ 4 \oplus 12 & 1 \oplus 9 \end{pmatrix}$$

$$= \begin{pmatrix} \min(7,8) & \min(4,5) \\ \min(4,12) & \min(1,9) \end{pmatrix}$$

$$= \begin{pmatrix} 7 & 4 \\ 4 & 1 \end{pmatrix}$$


Tropical数学入門 (3)
Tropical多項式

Tropical積のノテーションを変更しました
$\otimes \rightarrow \odot$

すみません。

この間、Tropical積の記号に$\otimes$を使っていたのですが、前回のセッションでベクトルの話をしていて、この$\otimes$だとテンソル積の記号と同じになるなと思いました。

よくみてみると、ほとんどの文献、$\otimes$ ではなく $\odot$を使っています。
今回のセッションから、Tropical積の記号、$\odot$に変えます。

過去2回の「Tropical数学入門」の(1)と(2)のpdf資料については、記号を修正しました。確認ください。

Tropical多項式
算術から代数へ

これまでは具体的な数のTropicalな計算（算術）をしてきたのですが、このセッションから、変数を使ってTropicalな式を計算します。「Tropical代数」が始まるのは、これからです。

まずは、Tropicalな多項式を考えます。今回のセッションでは、最後に、Tropical多項式のグラフがどういうものになるか紹介しようと思います。

これが、次回のセッションで取り上げる「Tropical幾何」につながっていきます。

冪乗 $x^{\odot a}$

整数 $a \in \mathbb{N}$ と実数$x \in \mathbb{R}$について、「xのa乗」のTropical版を考えて、それを $x^{\odot a}$と表すことにしましょう。

「xのa乗」とは、xをa回掛けることですので、そのTropical版$x^{\odot a}$は、xのa回の$Tropical$積として、次のように表すことができます。

$$x^{\odot a} = \overbrace{x \odot x \odot \cdots \odot x}^{a\text{個}}$$

この式の右辺は、次のように変形できます。

$$= x + x + \cdots + x$$

$$= a \cdot x$$

最後の式の・は、普通の実数との積です。

∞の冪乗 $\infty^{\odot a}$

Tropical代数　min-plus では、実数$\mathbb{R}$ の領域が ∞を含むように$\mathbb{R} \cup \{\infty\}$に拡大されています。

∞の冪乗 $\infty^{\odot a}$ を次のように定義します。

$a > 0$ の時　　$\infty^{\odot a} = \infty$

$a = 0$ の時　　$\infty^{\odot a} = 0$

Tropical単項式

以下、Tropicalな冪の $x^{\odot a}$ を、特に紛れがない限り、x^a と表すことにします。

d変数のTropical 単項式を次の形の式とします。

$$c \odot x_1^{a_1} \odot x_2^{a_2} \odot \cdots \odot x_d^{a_d}$$

ここで、$x_1, x_2, \cdots, x_d$は、実数値を取る d個の変数です。
cも実数値を取るものとします。それぞれの変数 $x_1, x_2, \cdots, x_d$が、$a_1, a_2, \cdots, a_d$乗されています。

Tropical単項式の例

四つの変数x_1, x_2, x_3, x_4からなる次のようなTropical積が与えられたとしましょう。

$$x_2 \odot x_1 \odot x_3 \odot x_1 \odot \mathrm{x}_4 \odot x_2 \odot x_3 \odot x_2$$

Tropical積は可換で$x_i \odot x_j = x_j \odot x_i$ですので、この性質を繰り返し使って、同じ変数をまとめて冪乗の形にすると、上の式は次のように変形できます。

$$x_1 \odot x_1 \odot x_2 \odot x_2 \odot \mathrm{x}_2 \odot x_3 \odot x_3 \odot x_4$$
$$= x_1^2 \odot x_2^3 \odot x_3^2 \odot x_4$$

この式の値を、普通の世界で計算すると、次のようになります。

$$= 2x_1 + 3x_2 + 2x_3 + x_4$$

四つの変数の冪を含んだ式の値は、四つの変数の冪を含まない線形の式になります。

Tropical多項式

n変数のTropical多項式 $p(x_1, \ldots, x_n)$は、有限個のTropical単項式の (Tropical和 $\oplus$ のもとで)の線形結合です。

次のような形をしています。

$$p(x_1, \ldots, x_n) = a \odot x_1^{i_1} x_2^{i_2} \cdots x_n^{\mathrm{in}} \oplus b \odot x_1^{j_1} x_2^{j_2} \cdots x_n^{\mathrm{jn}} \oplus \cdots$$

ここで、係数 $a, b, \cdots$ は実数で、冪乗の係数 $i_1, j_1, \cdots$は整数です。

Tropical多項式の性質

Tropical多項式$p(x_1, \ldots, x_n)$は、次のような性質を持ちます。

- $Tropical$多項式 $p(x_1, \ldots, x_n)$は、$\mathbb{R}^n \to \mathbb{R}$の関数である。

この関数がどのような値を取るか考えてみましょう。

$$p(x_1, \ldots, x_n) = a \odot x_1^{i_1} x_2^{i_2} \cdots x_n^{\mathrm{in}} \oplus b \odot x_1^{j_1} x_2^{j_2} \cdots x_n^{\mathrm{jn}} \oplus \cdots$$

ですので、

$$= \min(a + i_1 x_1 + i_2 x_2 + \cdots + i_n x_n,$$
$$b + j_1 x_1 + j_2 x_2 + \cdots + j_n \mathrm{x_n}, \cdots\cdots)$$

です。

- $Tropical$多項式 $p(x_1, \ldots, x_n)$は、複数の線形関数の集まりで定義され、その最小値を取る。

1変数3次のTropical多項式の例

最後の説明がわかりにくかったかもしれません。

ここでは、次のような1変数3次のTropical多項式$p(x)$を考えてみましょう。

$$p(x) = a \odot x^3 \oplus b \odot x^3 \oplus c \odot x + d$$

この$p(x)$は、$\mathbb{R} \to \mathbb{R}$の関数で、その値は次になります。

$$= \min(a + 3x, b + 2x, c + x, d)$$

この値をyとすると、yは次の四つの線形関数の集まりで定義され、その最小値を取ることになります。

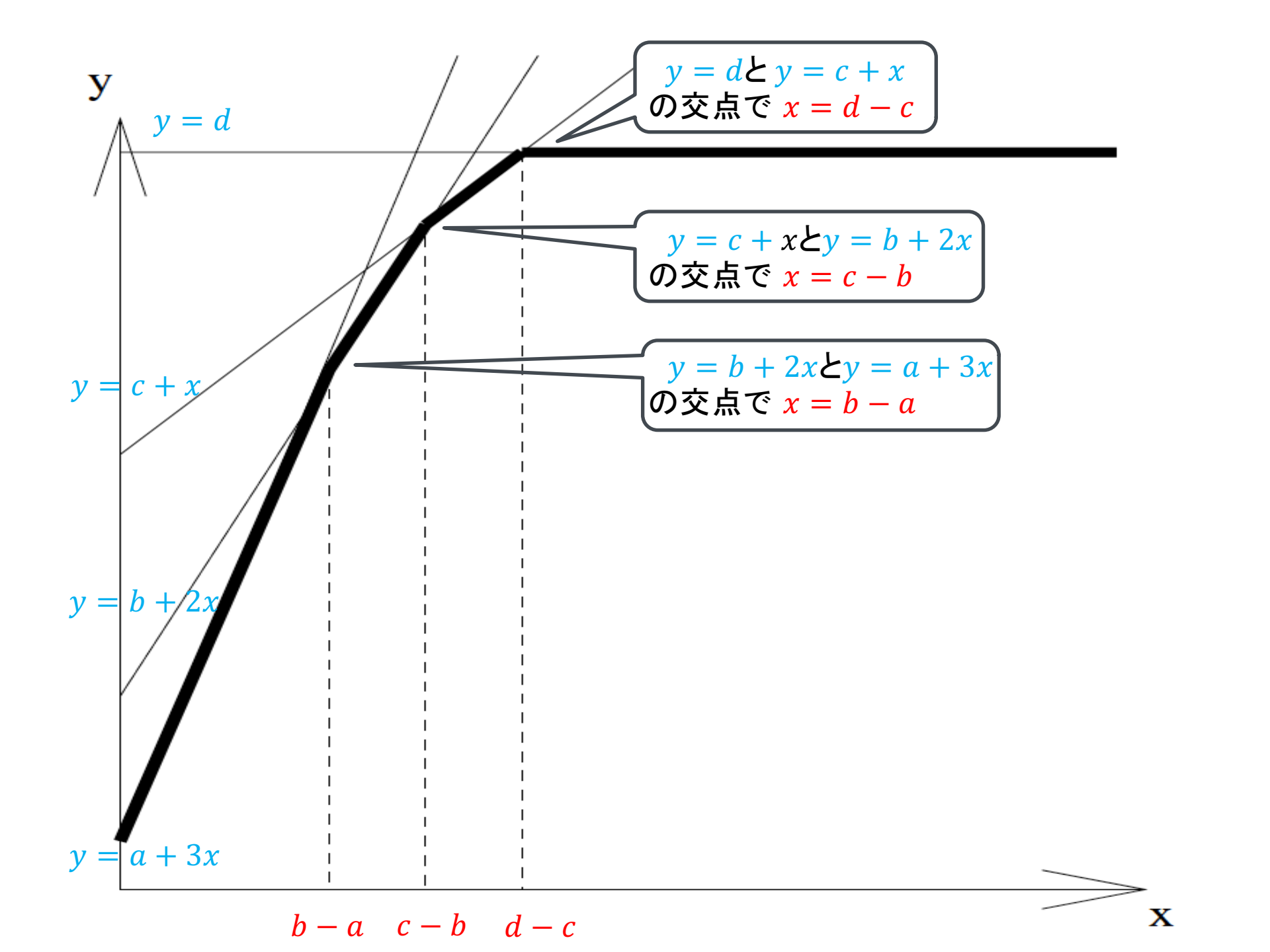
1. $y = a + 3x$
2. $y = b + 2x$
3. $y = c + x$
4. $y = d$

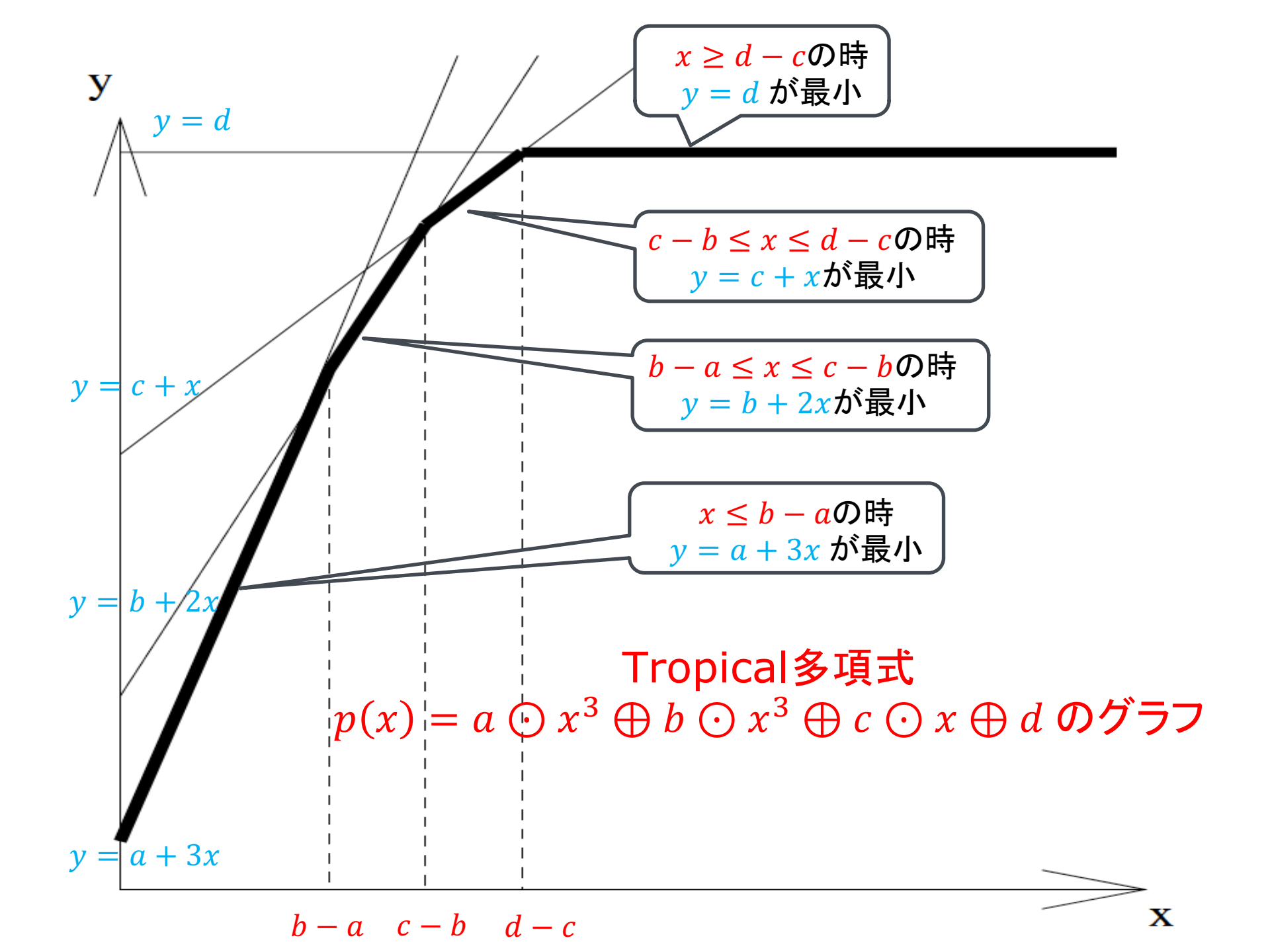
1変数3次のTropical多項式
$p(x) = a \odot x^3 \oplus b \odot x^3 \oplus c \odot x \oplus d$ のグラフ

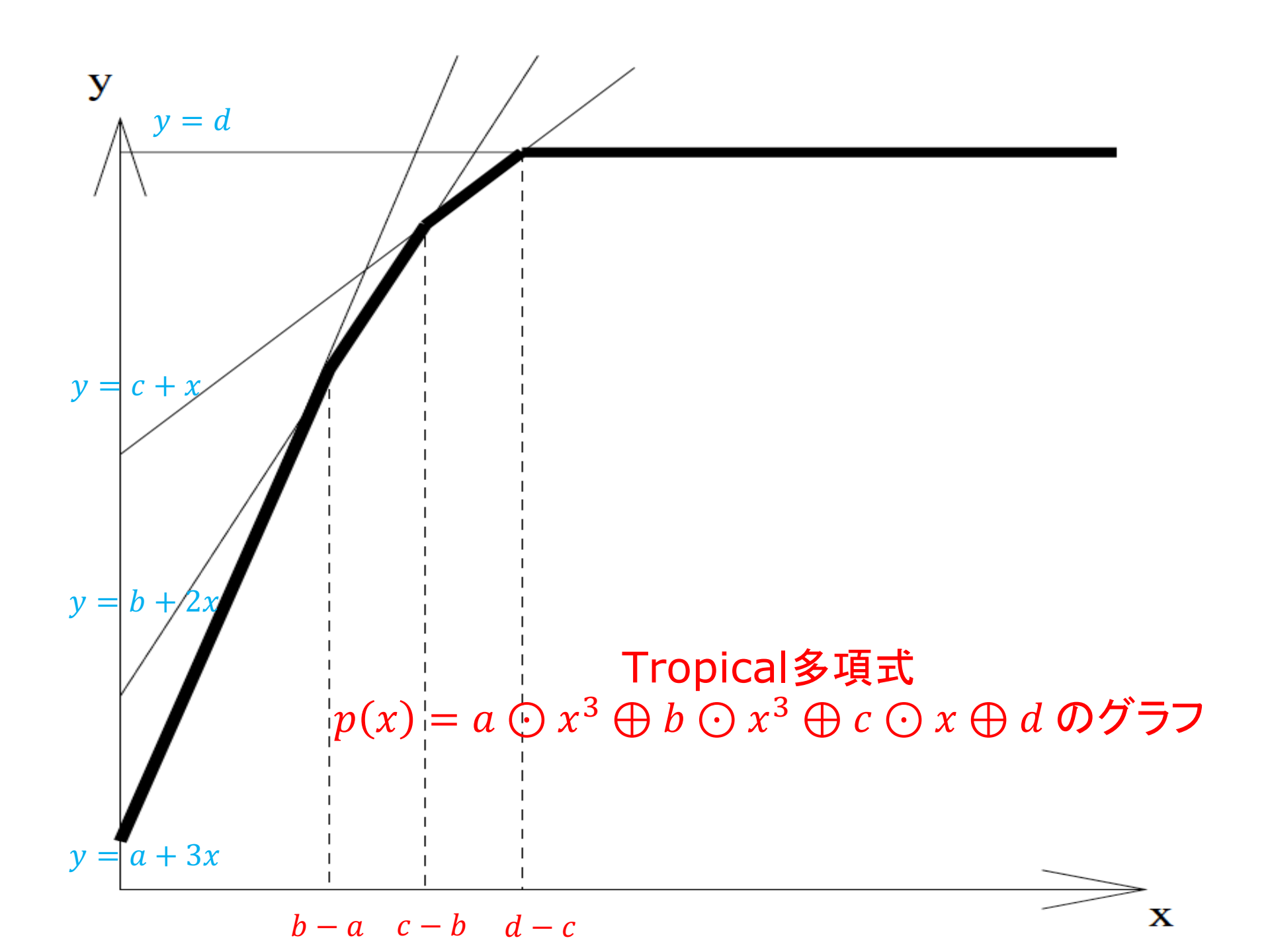
ただ、この四つの関数のうちどれがが最小の値を取るかは、この式のままではわかりません。xの値が変わるにつれて、最小の値を取る関数は変わります。

この四つの関数のグラフをかいて、大小関係を調べてみましょう。

1. $y = a + 3x$
2. $y = b + 2x$
3. $y = c + x$
4. $y = d$


y
x
$y = d$
$y = c + x$
$y = b + 2x$
$y = a + 3x$
$y = d$と$y = c + x$
の交点で $x = d - c$
$y = c + x$と$y = b + 2x$
の交点で $x = c - b$
$y = b + 2x$と$y = a + 3x$
の交点で $x = b - a$
$b - a$
$c - b$
$d - c$


y
$y = d$
$y = c + x$
$y = b + 2x$
$y = a + 3x$
$x \geq d - c$の時
$y = d$ が最小
$c - b \leq x \leq d - c$の時
$y = c + x$が最小
$b - a \leq x \leq c - b$の時
$y = b + 2x$が最小
$x \leq b - a$の時
$y = a + 3x$ が最小
Tropical多項式
$p(x) = a \odot x^3 \oplus b \odot x^3 \oplus c \odot x \oplus d$ のグラフ
$b - a$
$c - b$
$d - c$
x



Tropical多項式

$p(x) = a \odot x^3 \oplus b \odot x^3 \oplus c \odot x \oplus d$ のグラフ

区分的線形関数

このグラフのように、区分によって別々の線形関数を、連続に繋げてできる上に凸な関数を、区分的線形関数(piecewise-linear concave functions)といいます。

Tropical 多項式が表す関数は、区分的線形関数になります。


Tropical数学入門 (4)
Hypersurface
ニューラル・ネットワークの数理 -- Tropical数学入門

Hypersurface ?

このセッションでは、Tropical geometryの対象である、Tropicalな幾何学的オブジェクトの話をします。

前回、Tropicalな多項式の「グラフ」を見てきたのですが、そうした「グラフ」が、直接Tropical幾何の主要な関心の対象になるわけではないのです。

Tropical幾何の関心は、その多項式が表す「グラフ」の「表面」に向けられています。それをHypersurfaceと言います。

Hypersurface=「拡張された表面」

すこし、「表面」について考えてみましょう。

「表面」というと、二次元の平面を主要に考えると思いますが、次のように「表面」の概念は拡張できます。

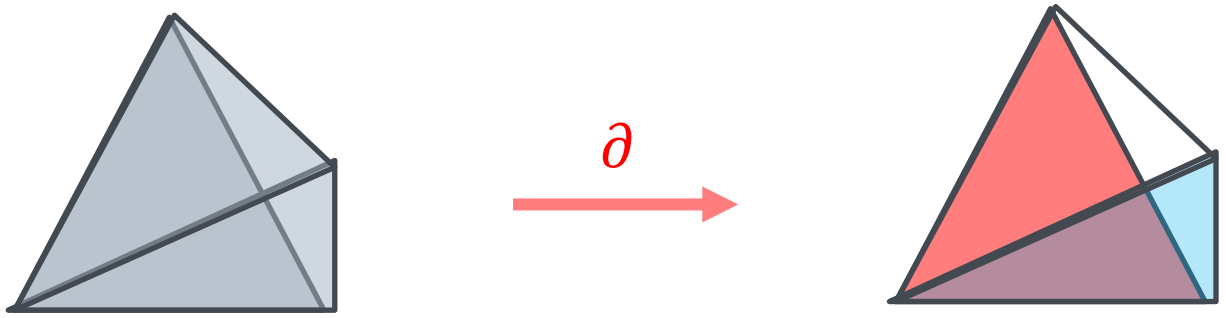
ある図形Aが与えられた時、その表面を∂Aで表すことにします。

立体から平面に

次のように、中身が詰まった三次元の三角錐Aがあったとします。

この立体の表面∂Aを考えてみましょう。

三角錐の表面は、赤・ブルー・白の三つの三角形からできています。
この三角形は、厚みを持たない二次元の平面です。



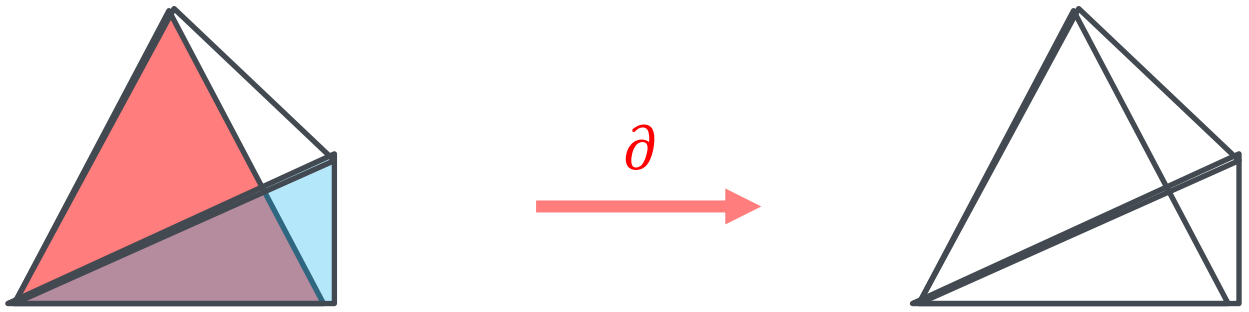
平面から直線へ

この三つの三角形に対して、その「表面」を考えてみましょう。

二次元の三角形の「表面」は、三角形を構成する「線分」と考えます。

こうして、6本の直線からなるワイアーフレームのような図形が、その「表面」として残ります。

それを構成しているのは、一次元の線分です。



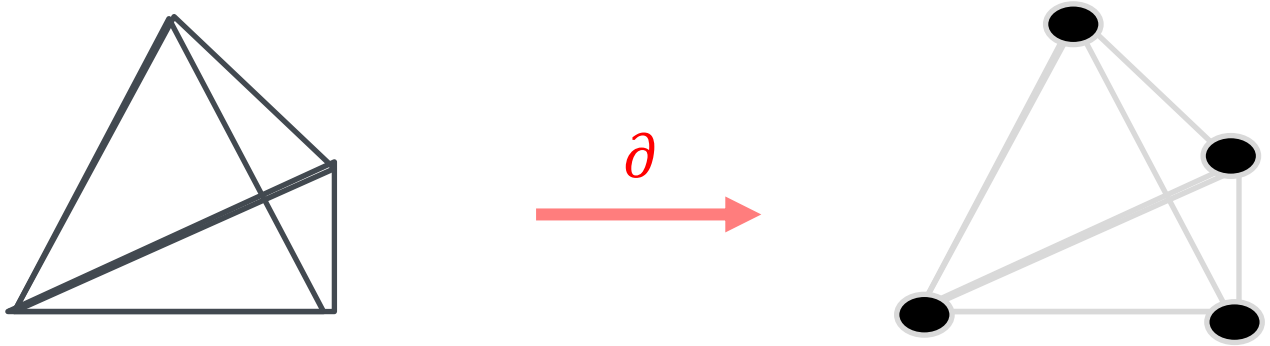
直線から点へ

この直線からなるワイアーフレームに対して、その「表面」を考えてみましょう。

一次元の線分の「表面」は、線分の端の点と考えます。

こうして、ワイアーフレームから、その「表面」として、四つの点が残ります。

それを構成しているのは、四つの0次元の点です。



なぜ、Hypersurfaceに注目するのか？

なぜ、Tropical幾何学では、Tropical多項式のグラフのHypersurfaceに注目するのでしょうか？

それは、Tropical多項式のHypersurfaceに、とても役にたつ情報が凝縮されているからです。

前回のセッションの例（一変数三次のTropical多項式）を改めて振り返ってみましょう。

1変数3次のTropical多項式の例

次のような1変数3次のTropical多項式$p(x)$を考えてみましょう。

$$p(x) = a \odot x^3 \oplus b \odot x^2 \oplus c \odot x \oplus d$$

この$p(x)$は、$\mathbb{R} \to \mathbb{R}$の関数で、その値は次になります。

$$= \min(a + 3x, b + 2x, c + x, d)$$

この値をyとすると、yは次の四つの線形関数の集まりで定義され、その最小値を取ることになります。

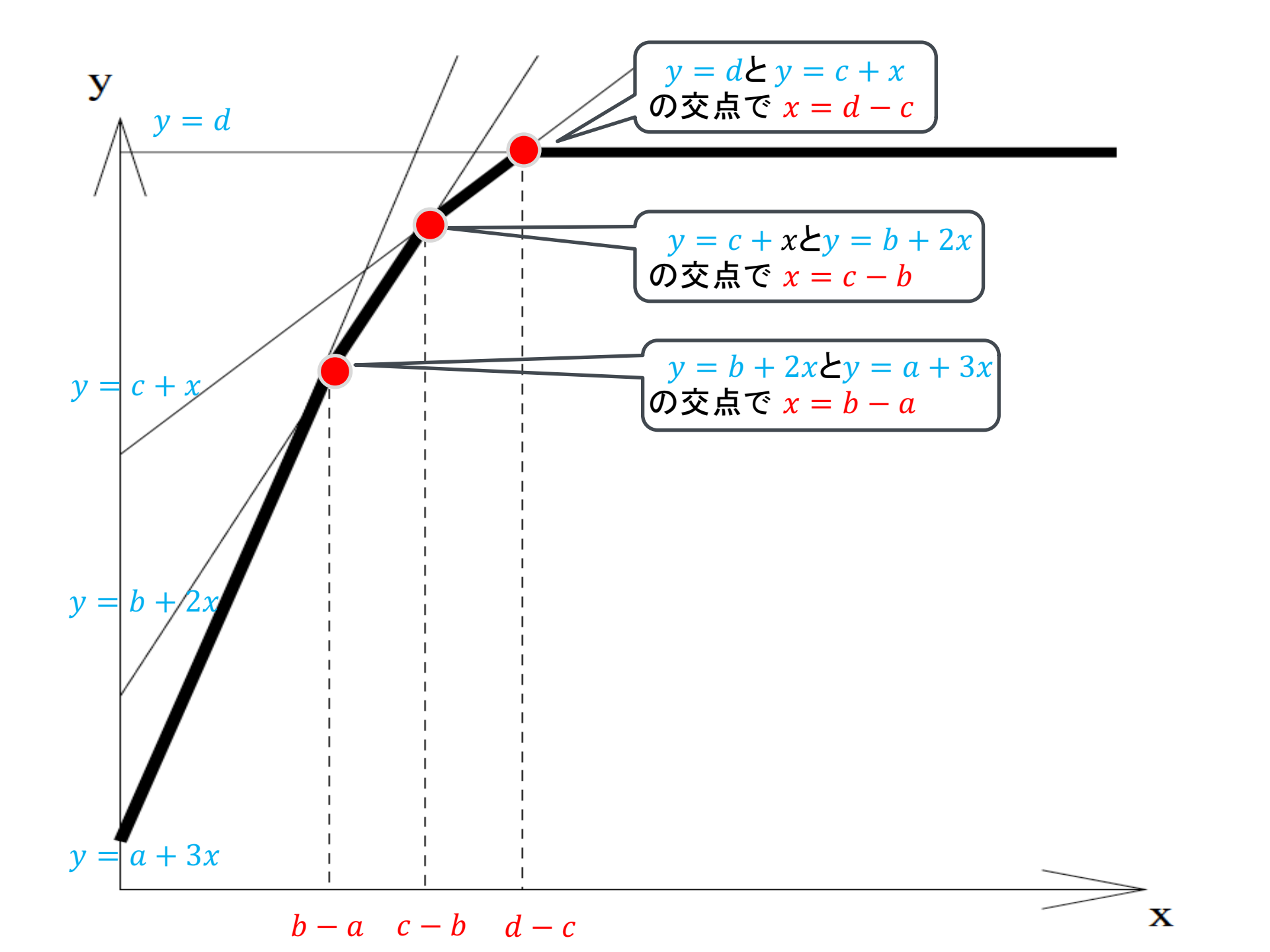
1. $y = a + 3x \quad \Leftarrow \quad a \odot x^3$
2. $y = b + 2x \quad \Leftarrow \quad b \odot x^2$
3. $y = c + x \quad \Leftarrow \quad c \odot x$
4. $y = d \quad \Leftarrow \quad d$

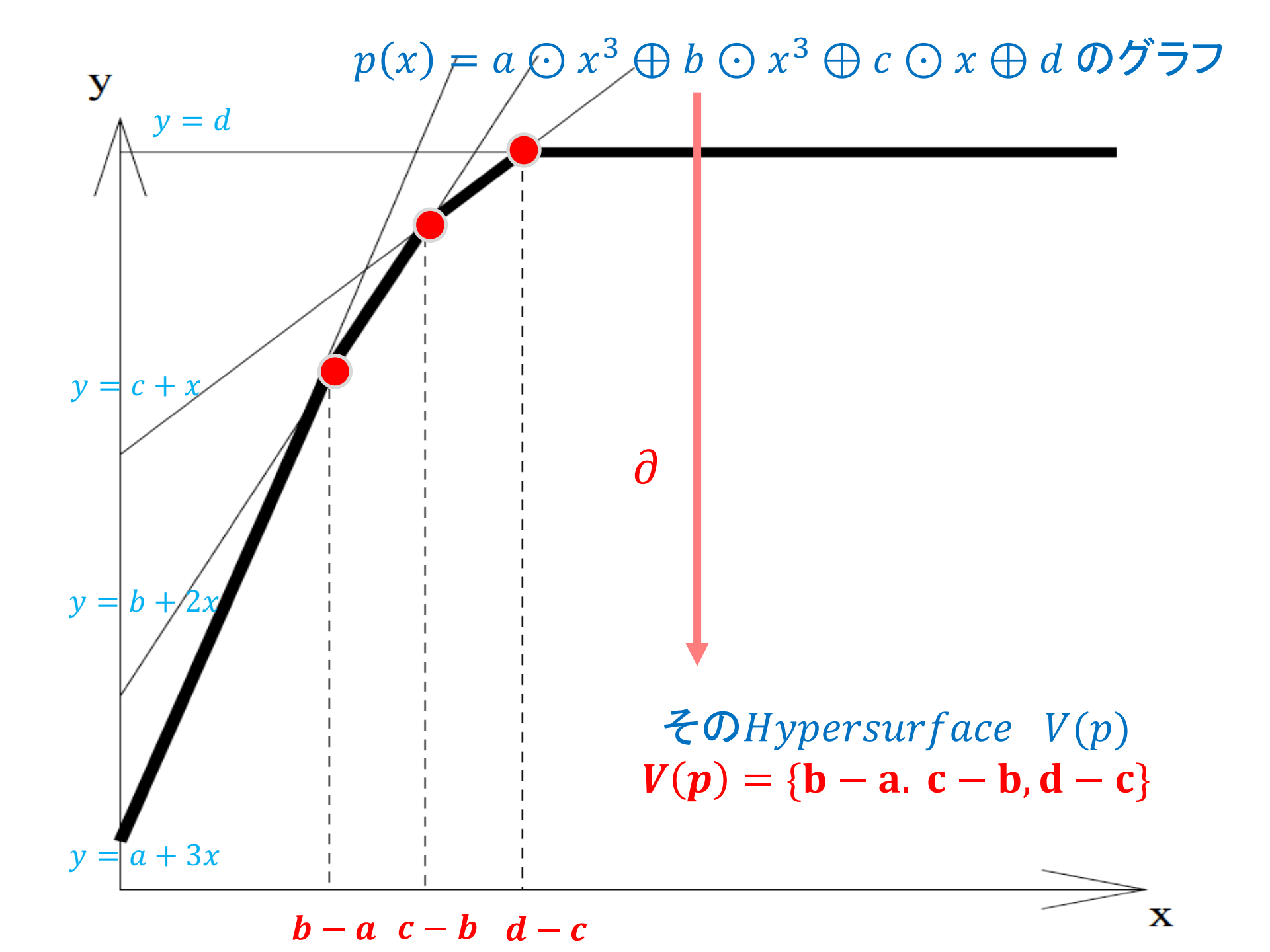
1変数3次のTropical多項式
$p(x) = a \odot x^3 \oplus b \odot x^3 \oplus c \odot x \oplus d$ のグラフ

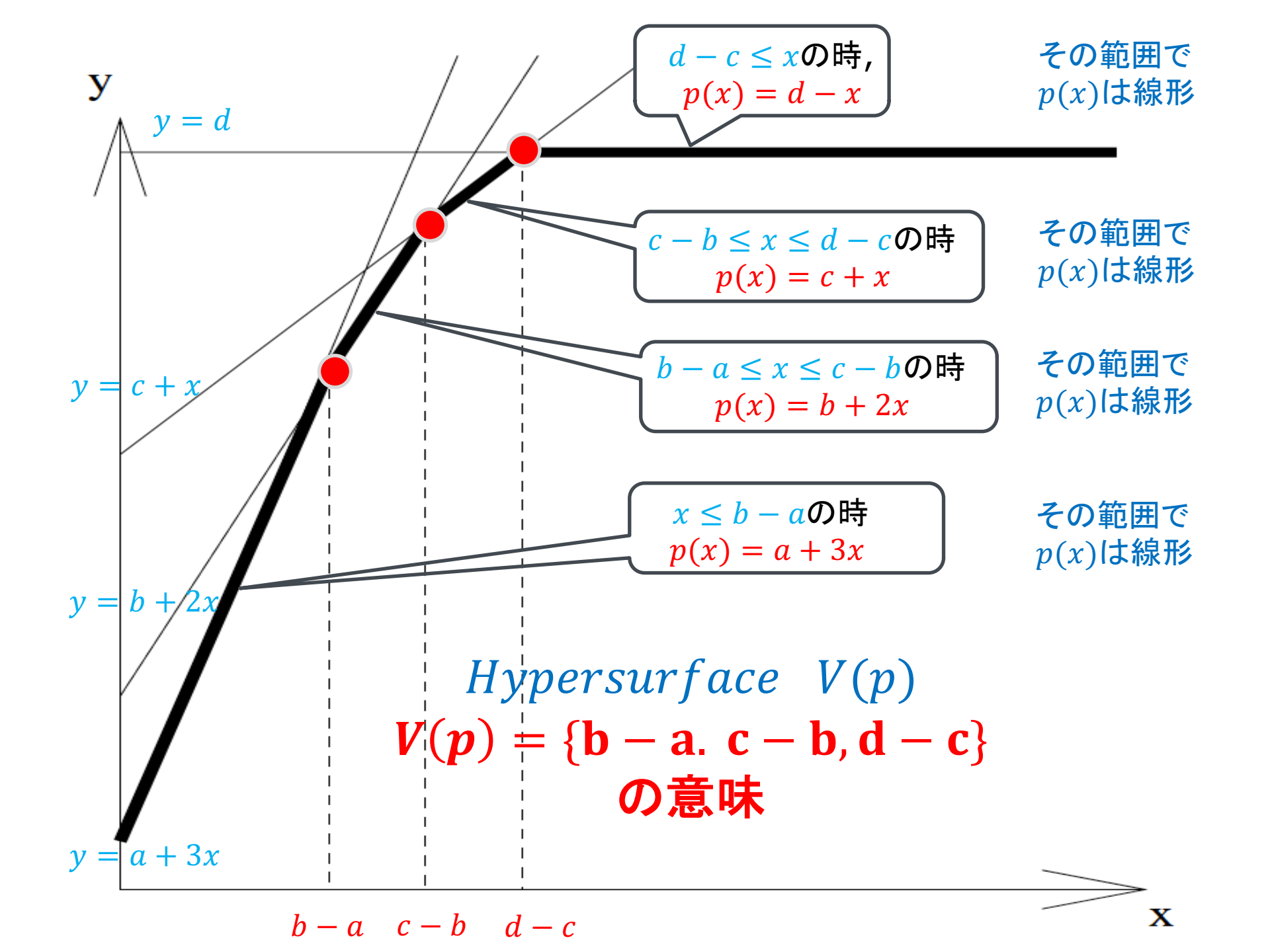
ただ、この四つの関数のうちどれがが最小の値を取るかは、この式のままではわかりません。xの値が変わるにつれて、最小の値を取る関数は変わります。

この四つの関数のグラフをかいて、大小関係を調べてみましょう。

1. $y = a + 3x$
2. $y = b + 2x$
3. $y = c + x$
4. $y = d$


y
x
$y = d$
$y = c + x$
$y = b + 2x$
$y = a + 3x$
$y = d$と$y = c + x$
の交点で $x = d - c$
$y = c + x$と$y = b + 2x$
の交点で $x = c - b$
$y = b + 2x$と$y = a + 3x$
の交点で $x = b - a$
$b - a$
$c - b$
$d - c$


$p(x) = a \odot x^3 \oplus b \odot x^3 \oplus c \odot x \oplus d$ のグラフ
y
$y = d$
$y = c + x$
$y = b + 2x$
$y = a + 3x$
∂
そのHypersurface $V(p)$
$V(p) = \{b - a.\ c - b, d - c\}$
$b - a$ $c - b$ $d - c$
x


y
$y = d$
$y = c + x$
$y = b + 2x$
$y = a + 3x$
x
$b - a$
$c - b$
$d - c$
$d - c \leq x$の時,
$p(x) = d - x$
その範囲で
$p(x)$は線形
$c - b \leq x \leq d - c$の時
$p(x) = c + x$
その範囲で
$p(x)$は線形
$b - a \leq x \leq c - b$の時
$p(x) = b + 2x$
その範囲で
$p(x)$は線形
$x \leq b - a$の時
$p(x) = a + 3x$
その範囲で
$p(x)$は線形
$Hypersurface \quad V(p)$
$V(p) = \{b - a.\ c - b, d - c\}$
の意味

$Hypersurface\ V(p)$ とは？

Tropical 多項式 p は、複数のTropical単項式のTropical和 $\oplus$ で定義される。pを構成するTropical 単項式は、ある線形写像の値を取る。

Tropical和 $\oplus$ の定義から、Tropical 多項式 p の値は、これらの単項式が作る線形写像の組から、最小の値を取るもので決まる。

Hypersurface V(p) は、どの点で、最小の値を持つ線形写像が切り替わるかを示す。この点では、2つの単項式の線形写像は、同じ値を持つ。


Tropical数学入門 (5)
Hypersurface 2
ニューラル・ネットワークの数理 -- Tropical数学入門

二変数Tropical多項式の Hypersurface

このセッションでは、二つの変数を持つTropical多項式の Hypersurface について考えます。

前回のセッションでサンプルとしてみた見た三次のTropical多項式は、変数が一つのものでした。

$$p(x) = a \odot x^3 \oplus b \odot x^2 \oplus c \odot x \oplus d$$

そのHypersurface $V(p)$は、三つの点からなるものでした。

$$V(p) = \{\mathrm{b} - \mathrm{a}.\ \mathrm{c} - \mathrm{d}, \mathrm{d} - \mathrm{c}\}$$

今度は、次のような二変数の2次Tropical多項式を考えます。

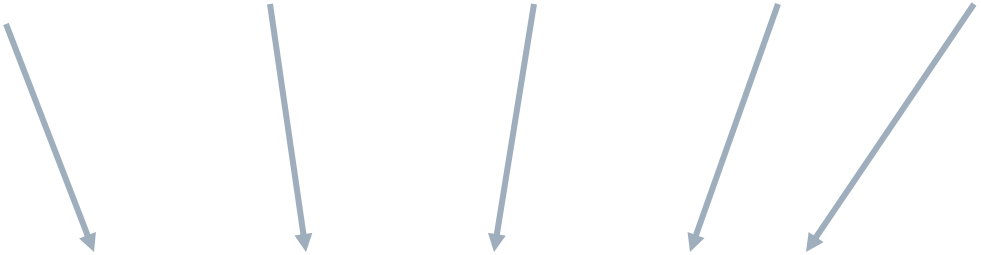
$$p(x,y) = a \odot x^2 \oplus b \odot xy \oplus c \odot x^2 \oplus d \odot y \oplus e$$

このTropical多項式は、次のような値を持ちます。

$$= \min(a + 2x, b + xy, c + 2x, d + y, e)$$

項の対応、チェックしてください。

$$p(x,y) = a \odot x^2 \oplus b \odot xy \oplus c \odot x^2 \oplus d \odot y \oplus e$$



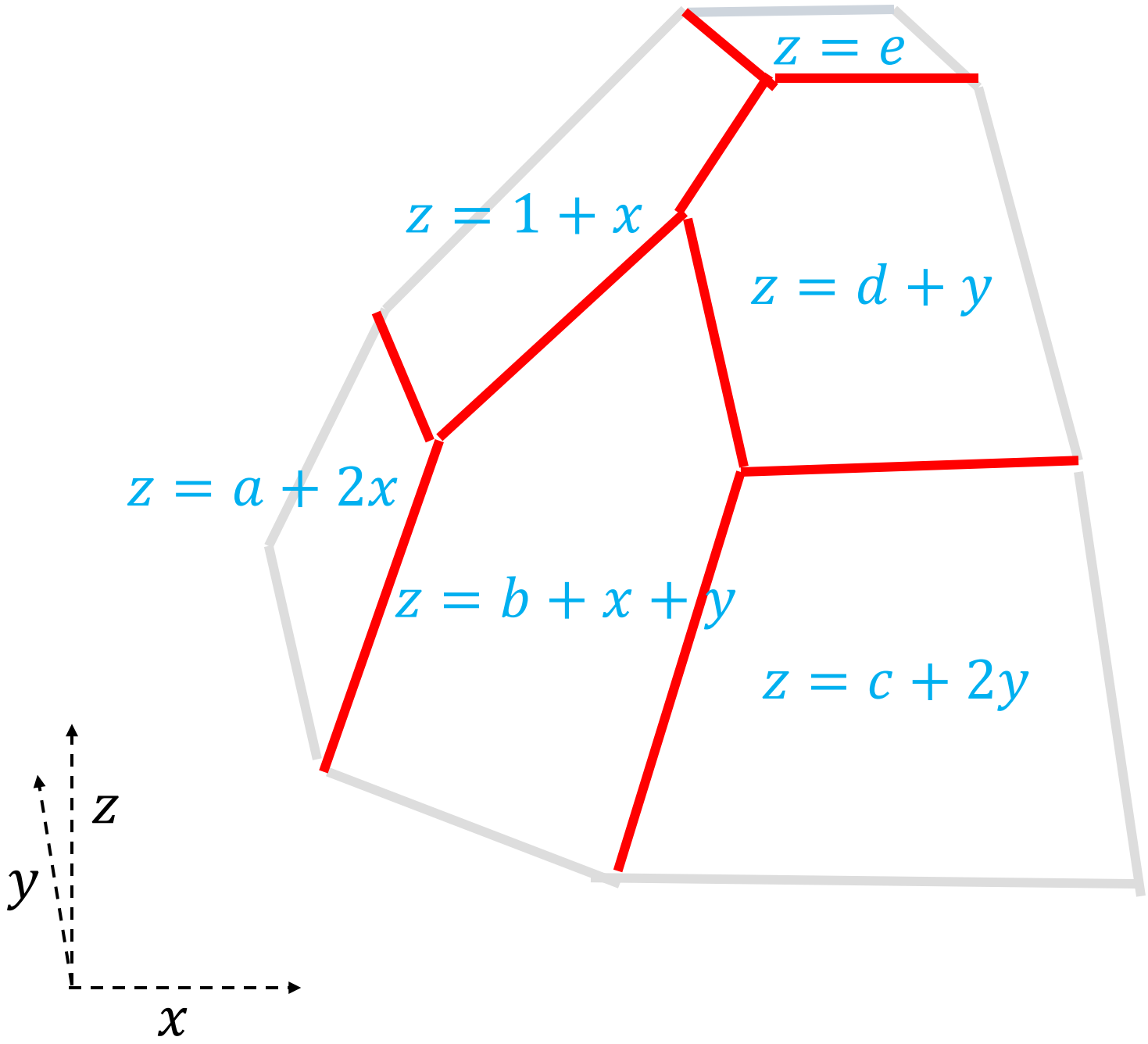
$$= \min(a + 2x, b + xy, c + 2x, d + y, e)$$

$$\min(a+2x, b+xy, c+2x, d+y, e)$$

の計算ですが、5つの項をそれぞれをzとおいて、グラフを書いてみます。

$$z = a + 2x$$
$$z = b + xy$$
$$z = c + 2x$$
$$z = d + y$$
$$z = e$$

次のようになります。


$z = e$
$z = 1 + x$
$z = d + y$
$z = a + 2x$
$z = b + x + y$
$z = c + 2y$
z
y
x

5つのグラフのそれぞれは、平面として広がっています。

先のグラフは、5つの平面の広がりを全て書いているわけではありません。

ここで知りたいのは、どの平面が最小の値を取りうるのかという情報です。ここでは、係数間に次のような関係があると仮定しています。

$$2b < a + c\,;\ 2f < a + e\,;\ 2d < c + e$$

この条件は、以前の一変数3次Tropical多項式でのサンプルで

$$\mathrm{b} - \mathrm{a} \leq \mathrm{c} - \mathrm{b} \leq \mathrm{d} - \mathrm{c}$$

を仮定したのと同じです。

赤い線は何を表すか？

問題は、先のグラフで赤い線が何を表しているかということです。

明らかに、赤い線は二つあるいは三つの平面が交わる線（あるいは線分）を表しています。

前回、Tropical多項式pのHypersurface $V(p)$ を次のように定義しました。これは一変数の場合です。

Hypersurface V(p) は、どの点で、最小の値を持つ線形写像が切り替わるかを示す。この点では、2つの単項式の線形写像は、同じ値を持つ。

二変数の今回の場合、$V(p)$の定義は、次のようになります。

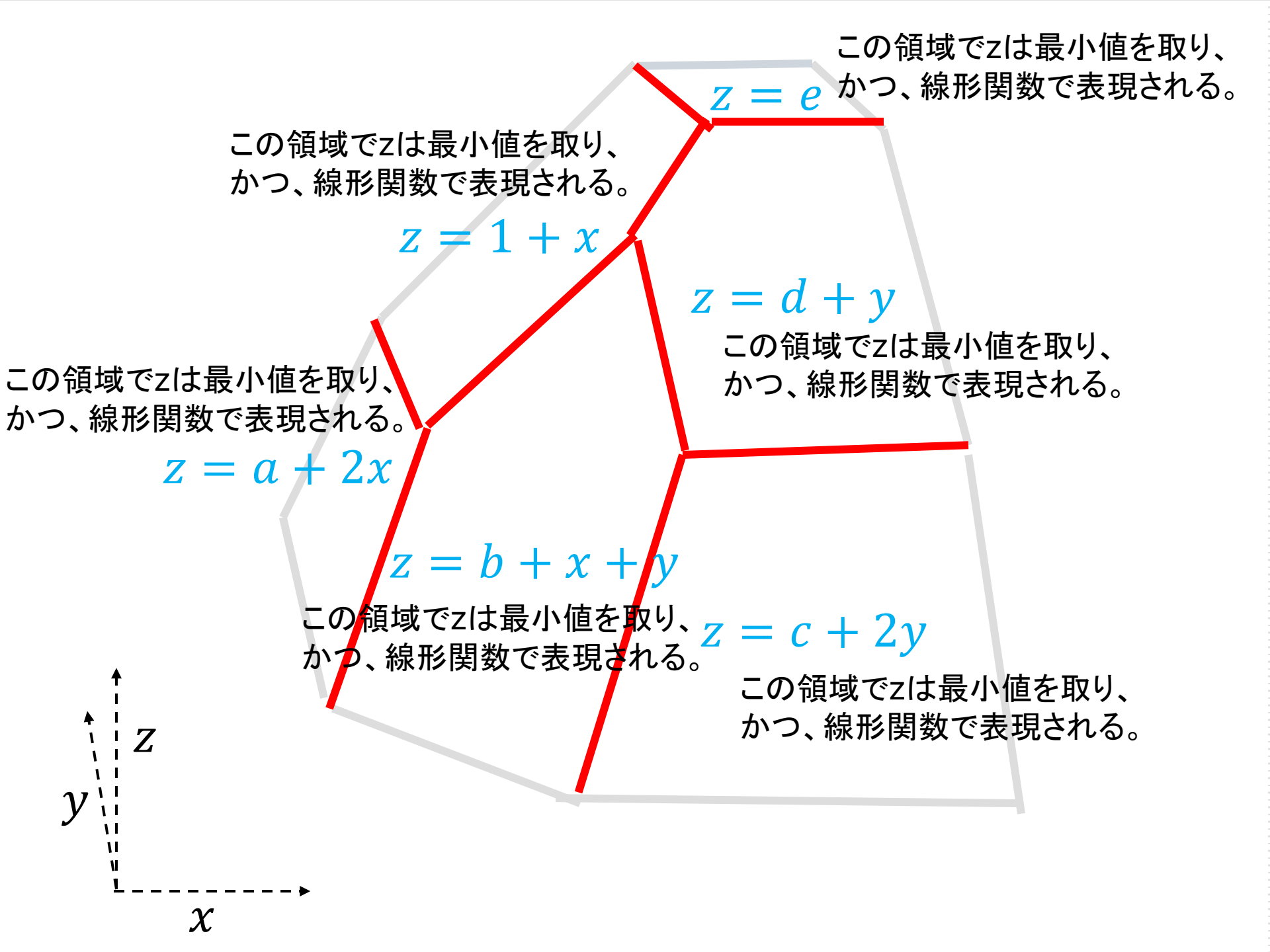
Hypersurface V(p) は、どの線で、最小の値を持つ線形写像が切り替わるかを示す。この線上では、2つあるいは3つの単項式の線形写像は、同じ値を持つ。

そして、赤い線で囲まれた領域は、zが最小となる領域を表しています。また、この領域は、一つの線型写像で表現されます。次の図で、それを示します。

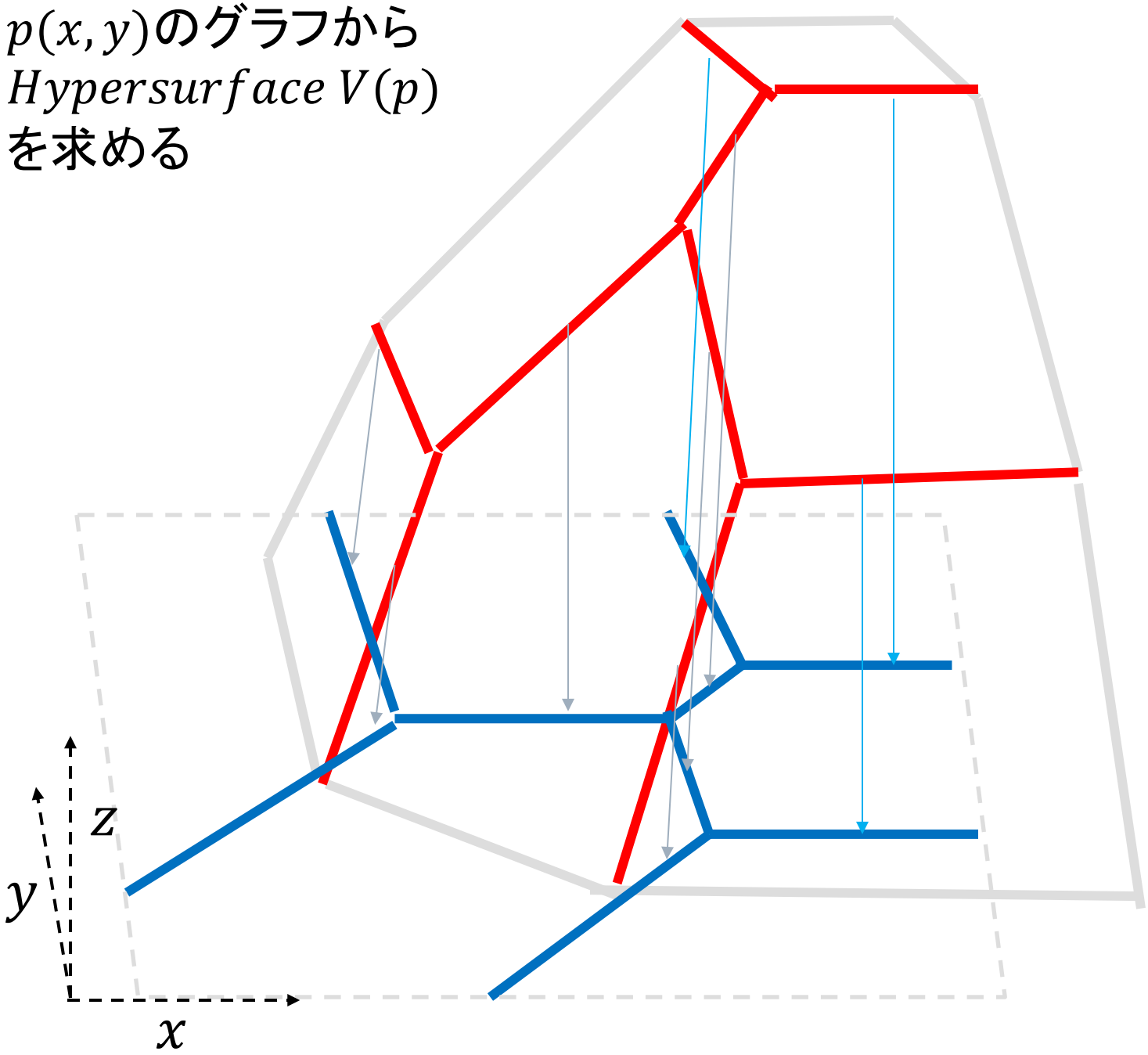
もっとも、この赤いグラフがそのまま$V(p)$を表すわけではありません。$V(p)$は、このグラフのHypersurface なの、もう一次元低いところの xy平面上で表現されます。

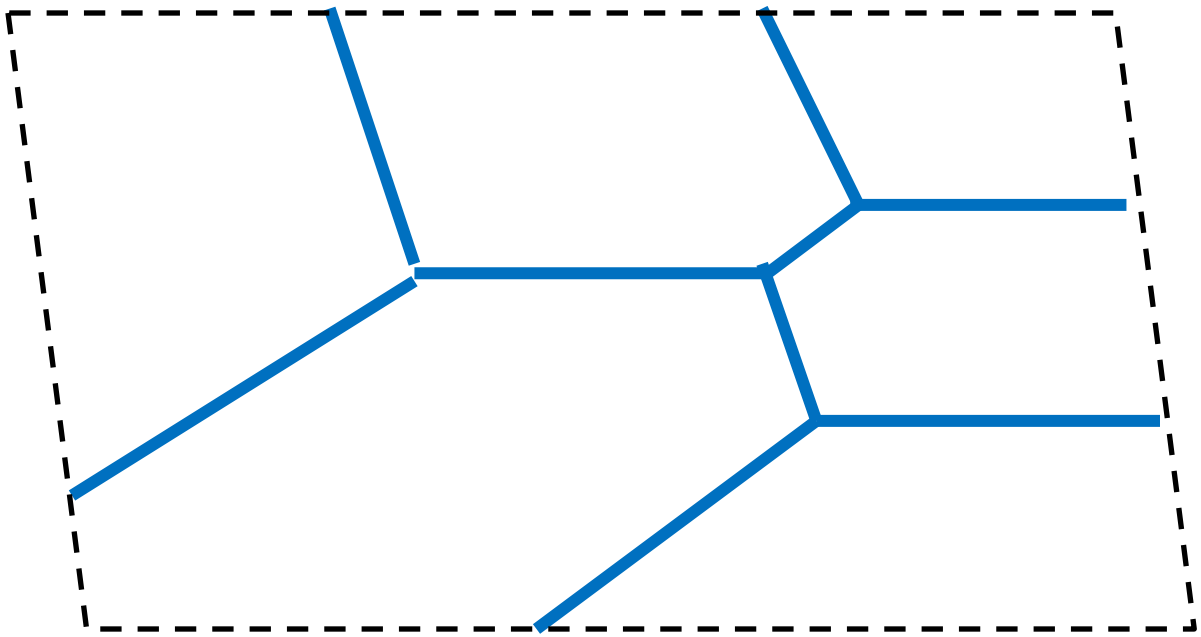
それを求めるのは簡単で、赤いグラフからxy平面上へ足を下ろせばいいのです。（射影する）

次の次の図で、それを示します。


この領域でzは最小値を取り、
かつ、線形関数で表現される。
$z = e$
この領域でzは最小値を取り、
かつ、線形関数で表現される。
$z = 1 + x$
$z = d + y$
この領域でzは最小値を取り、
かつ、線形関数で表現される。
この領域でzは最小値を取り、
かつ、線形関数で表現される。
$z = a + 2x$
$z = b + x + y$
この領域でzは最小値を取り、
かつ、線形関数で表現される。
$z = c + 2y$
この領域でzは最小値を取り、
かつ、線形関数で表現される。
z
y
x

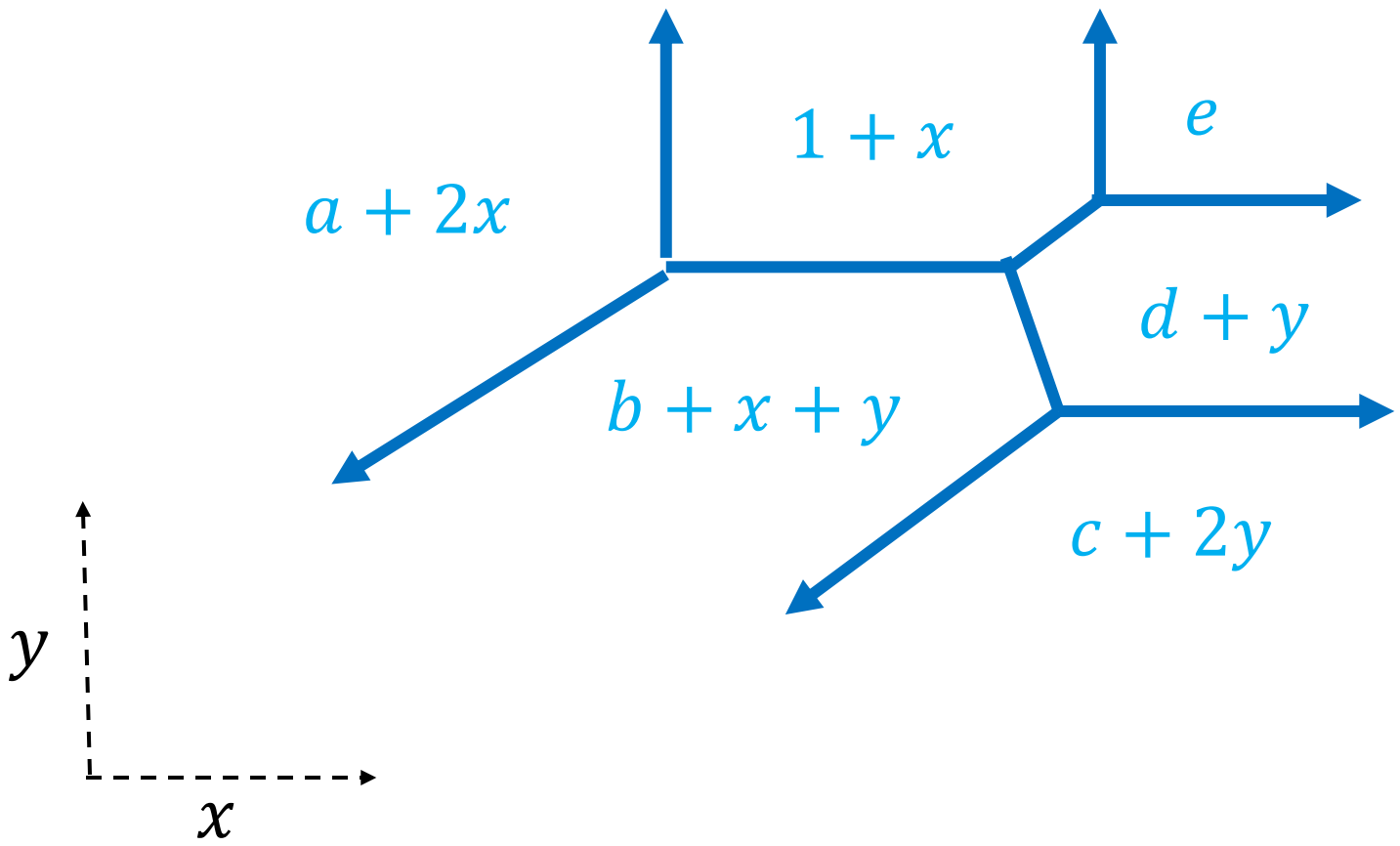
$p(x,y)$のグラフから
$Hypersurface\ V(p)$
を求める





$$p(x,y) = a \odot x^2 \oplus b \odot xy \oplus c \odot x^2 \oplus d \odot y \oplus e \, \mathcal{O}$$

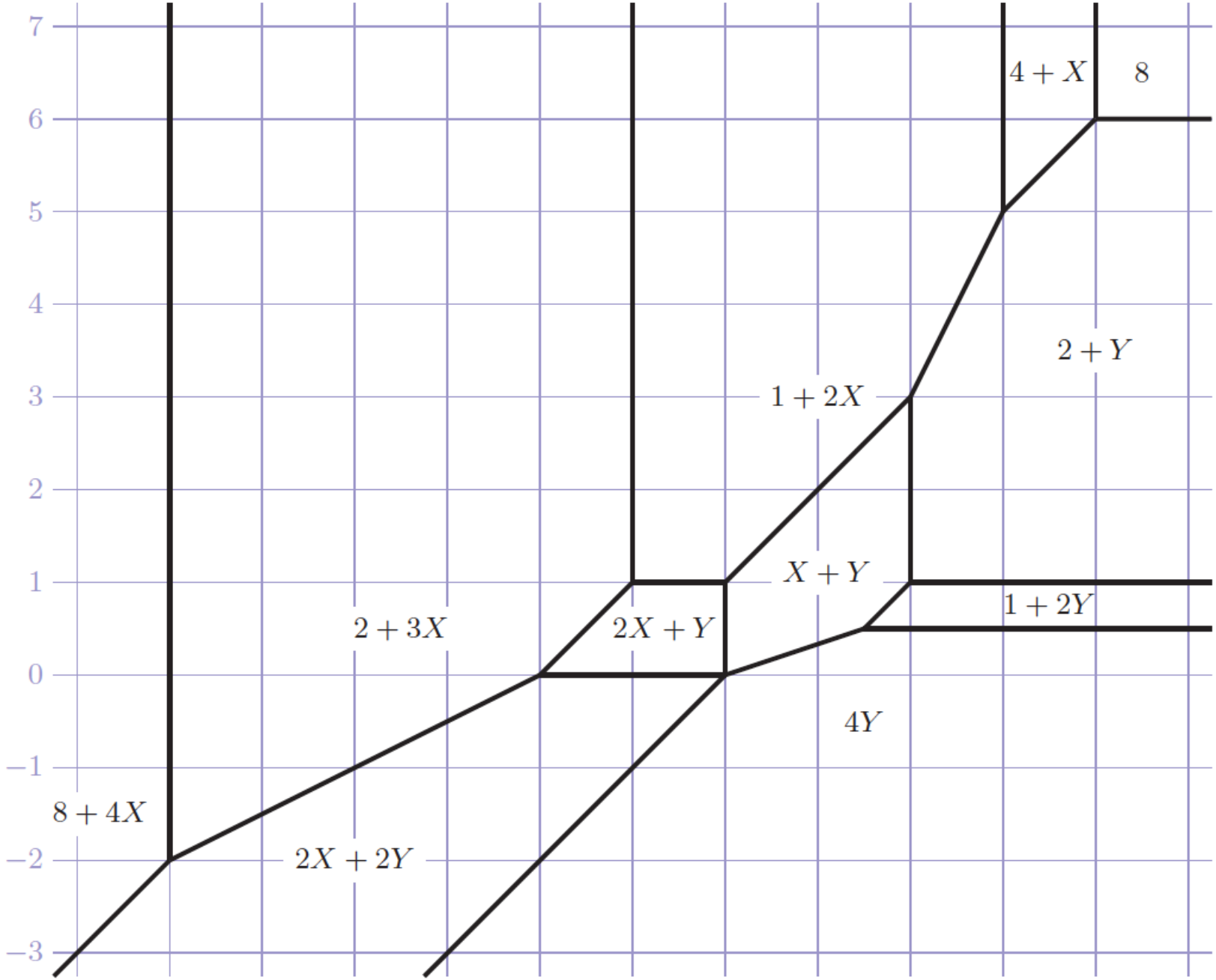
Hypersurface V(p)

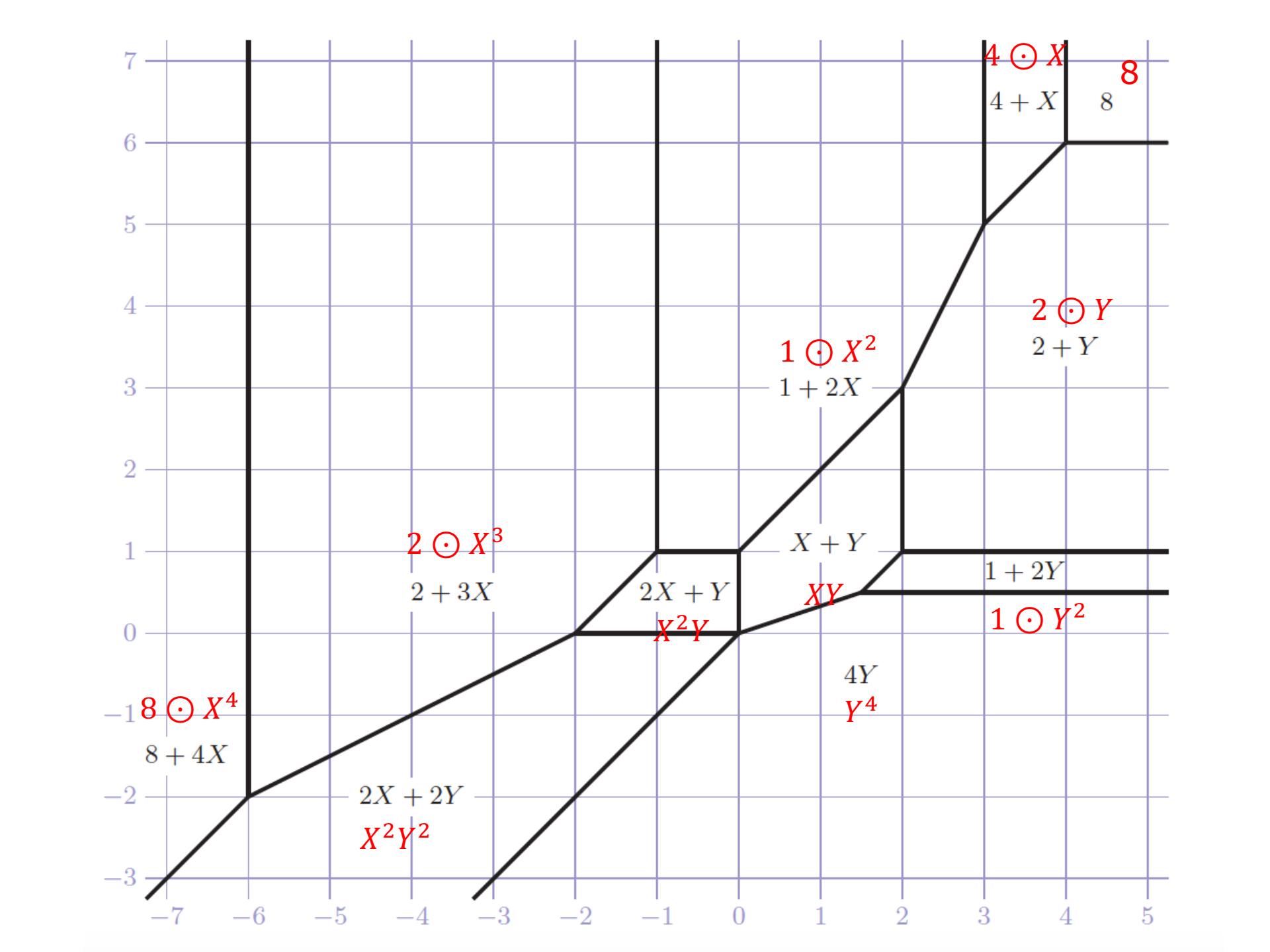


$V(p)$からTropical多項式 pを求める?

逆に、Hypersurface V(p) が与えられた時、Tropical 多項式 p を求めることも可能でしょうか？

次のようなHypersurface V(p) が与えられたとしましょう。


7
6
5
4
3
2
1
0
−1
−2
−3
−7
−6
−5
−4
−3
−2
−1
0
1
2
3
4
5
$4 + X$
8
$2 + Y$
$1 + 2X$
$X + Y$
$1 + 2Y$
$2 + 3X$
$2X + Y$
$4Y$
$8 + 4X$
$2X + 2Y$


$4 \odot X$
$4 + X$
8
8
$2 \odot Y$
$2 + Y$
$1 \odot X^2$
$1 + 2X$
$2 \odot X^3$
$2 + 3X$
$X + Y$
$1 + 2Y$
$2X + Y$
X^2Y
XY
$1 \odot Y^2$
$4Y$
Y^4
$8 \odot X^4$
$8 + 4X$
$2X + 2Y$
X^2Y^2

$$p(x,y) = 8 \oplus 4 \odot X \oplus 2 \odot Y \oplus 1 \odot X^2 \oplus XY \oplus 1 \odot Y^2 \oplus 2 \odot X^3 \oplus X^2Y \oplus 8 \odot X^4 \oplus X^2Y^2 \oplus Y^4$$

$$\min(\, 8, 4 + X, 2 + Y, 1 + 2X, X + Y, 1 + 2Y, 2 + 3X, 2X + Y, 8 + 4X, 2X + 2Y, 4Y \,)$$


Tropical数学入門 (6)
Newton図形
ニューラル・ネットワークの数理 -- Tropical数学入門

Newton図形

これまで、Tropical多項式 p のHypersurface $V(p)$についてみてきました。

pの値は、pを構成する単項式の piecewise（区分的）に分割された領域のな最小値で決まりますので、Hypersurface $V(p)$の情報は役に立ちます。

ただ、Hypersurface $V(p)$を求めるということは、ある区分上ですべての単項式の値を比較することに他なりません。それは、実際にはなかなか手間がかかります。

今回のセッションでは、Tropical なHypersurface $V(p)$ と関う連を持つもう一つの図形 、Tropical多項式 pの「Newton図形」を紹介します。

ちなみに、このNewtonは、あの「万有引力」のIsac Newton です。

昔、Deep LearningのGradient Decentsは、数値計算の「Newton法」の変種だと思って、Newtonの論文を調べたことがありました。ただ、Newtonがやっていたことは、数値計算の手法などではなく、全く代数的なアプローチでした。

それは、僕の想像を遥かに超えて、すべての関数を無限級数に代数的に展開するというものでした。さっぱり理解できず、そっと論文を閉じたのですが。

「Newtonは、僕が思っていたより遥かにスゴイ」

https://www.facebook.com/photo/?fbid=1020471664865359 6&set=a.2252867198672

今回紹介する「Newton図形」は、Newtonの関数の無限級数展開というアプローチに関係したものです。10年前、さっぱりわからなかったことに少し近づけて、少し嬉しいです。

多項式の表示

$\mathbb{R}^d \to \mathbb{R}$である$d$変数の多項式関数 $F(x)$は、一般に次のように表すことができます。

$$F(x) = \sum_{u \in S} a_u x_1^{u_1} x_2^{u_2} \cdots x_d^{u_d}$$

この表示で、少し説明が必要なのは、$x \in \mathbb{R}^d$（すなわち、d個の実数の組、ベクトルと考えても構いません）と考えられていることと、

uは整数（0, $\pm 1, \pm 2,$ ...）の値を走るということです。この$u \in S$であるSを関数Fのsupport（台）と言います。

Tropical多項式の表示

同様の考えで、$\mathbb{R}^d \to \mathbb{R}$である$d$変数のTropical 多項式関数$F(x)$を次のように表すことができます。

$$F(x) = \bigoplus_{u \in S} a_u \odot x_1^{u_1} x_2^{u_2} \cdots x_d^{u_d}$$

この式は、次のように展開できます。（単項式の展開）

$$= \min\{a_u + u_1 x_1 + u_2 x_2 + \cdots + u_d x_d \mid u \in S\}$$

ここで、変数の指数部分 $(u_1, u_2, \cdots, u_d)$と変数$(x_1, x_2, \cdots, x_d)$の双方をベクトルと考えると、先の式は次のように二つのベクトルの内積を使って表現できます。

$$= \min\{a_u + < u, x > \mid \mathrm{u} \in S\}$$

Newton図形と変数の指数部

Newton図形は、先に見た指数部分のベクトルのみに注目することで作ることができます。

Tropical多項式が、次のように与えられているとします。

$$F(x) = \bigoplus_{u \in S} a_u \odot x_1^{u_1} x_2^{u_2} \cdots x_d^{u_d}$$

この時、上の赤い四角で囲った部分$u_1, u_2, \cdots, u_d$ のみに注目します。

こうして得られるベクトル $\mathrm{u} = (u_1, u_2, \cdots, u_d)$ について、$u \in S$の意味を考えておきましょう。

一見すると、多項式を構成する単項式の変数の指数が、すべて同じ $(u_1, u_2, \cdots, u_d)$ で表されているように見えるかもしれませんが、そうではありません。

大事なのは、$u \in S$の方で、こちらは、こうしたuがたくさん（正確にいうと単項式の数だけ）Sに含まれていることを主張しています。

ですので、$(u_1, u_2, \cdots, u_d)$の要素の値は、単項式毎に変わります。

対応する単項式が与えられないと、$(u_1, u_2, \cdots, u_d)$は、

変数x_1の指数は、u_1である。

変数x_2の指数は、u_2である。

……

変数x_dの指数は、u_dである。

という、形式的関係を表しているだけです。

具体的な単項式との対応が与えられて初めて、$(u_1, u_2, \cdots, u_d)$の要素の値は確定します。

Newton図形の作り方

Newton図形は、これまでみてきた Tropical多項式 p の Hypersurface V(p)より、比較的簡単に作ることができます。

第一に、多項式を構成する一つの単項式に注目して、その指数部だけからなる、ベクトル$(u_1, u_2, \cdots, u_d)$をピックアップします。

これを単項式の数だけ繰り返します。先に見たように、単項式の数だけ異なるベクトルが得られるはずです。

第二に、そのベクトルを、d次元の空間にプロットします。

$u_1, u_2, \cdots, u_d$は、すべて整数ですので、d次元空間の格子点にベクトルはプロットされるはずです。

convex hull

第三に、プロットした点をすべて包み込む最小の立体を考えます。プロットした点を含む、おおきな立体はいくらでも存在しうるのですが、「最小」のものは、一つしか存在しません。

イメージ的には、プロットした点を、伸び縮みするゴムの膜で覆って、強く締め上げるといいのです。

こうした立体は、空間的には、凸状の形をしているので、convex hull といいます。(日本語では「凸包」と訳すようです。)

Newton図形のサンプル 1

次の二変数の多項式のNewton図形を作ってみましょう。

$$f = 7x + 8y - 3xy + 4x^2y - 17xy^2 + x^2y^2$$

変数x, yの指数部が見えやすいようにすこし変形し、指数部のベクトルをピックアップします。

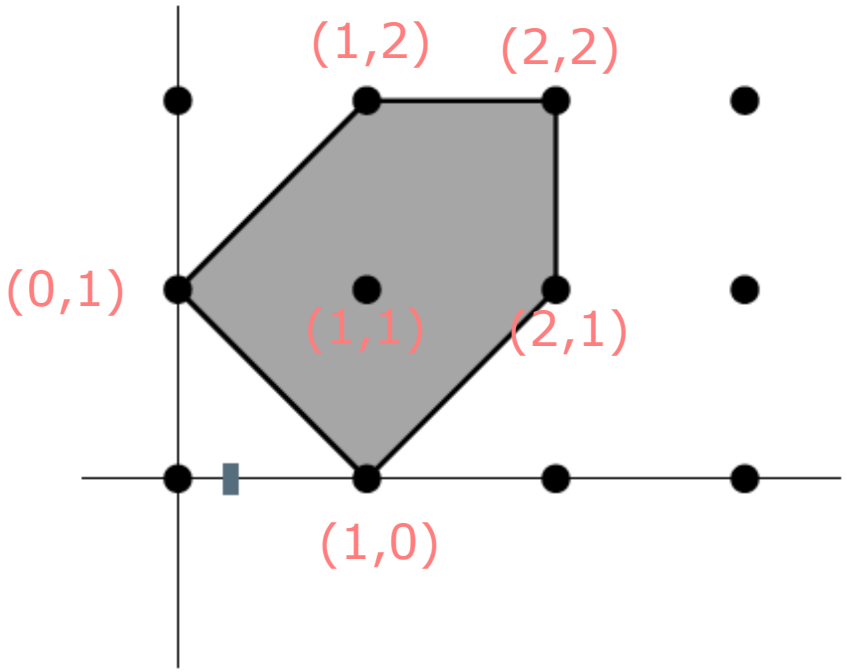
$$= 7x^1y^0 + 8x^0y^1 - 3x^1y^1 + 4x^2y^1 - 17x^1y^2 + x^2y^2$$

(1,0) (0,1) (1,1) (2,1) (1,2) (2,2)

$$f = 7x + 8y - 3xy + 4x^2y - 17xy^2 + x^2y^2$$

のNewton図形

こうして得られたベクトル (1,0); (0,1); (1,1); (2,1); (1,2) ; (2,2) を二次元のxy平面にプロットして、convex hull をとると、こうなります。



Newton図形のサンプル 2

次の二変数の多項式のNewton図形を作ってみましょう。

$$g = x^{-1} - y^{-1} + 3x - 2y + xy$$

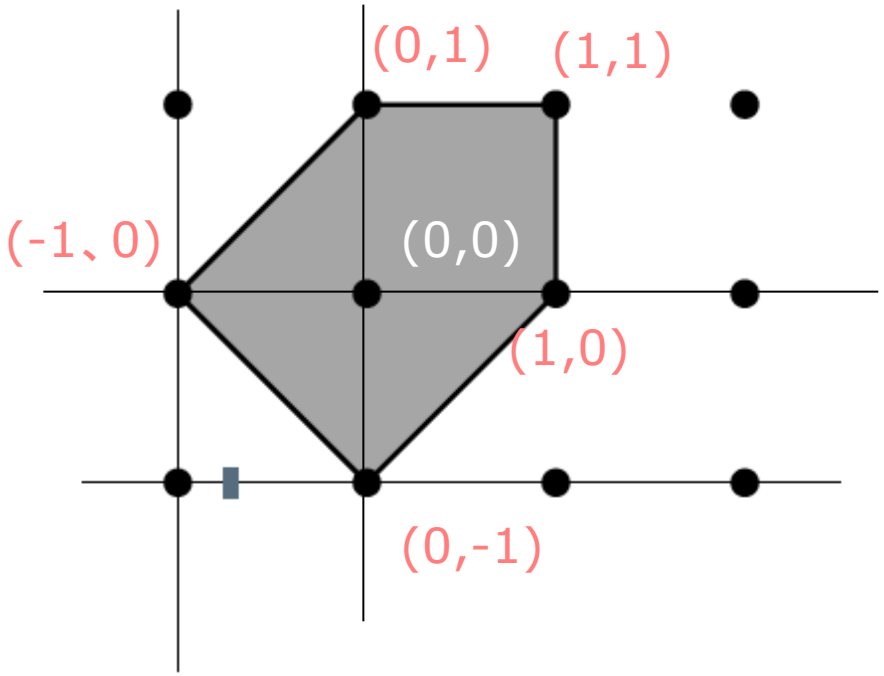
変数x, yの指数部が見えやすいようにすこし変形し、指数部のベクトルをピックアップします。

$$= x^{-1}y^{0} - x^{0}y^{-1} + 3x^{1}y^{0} - 2x^{0}y^{1} + x^{1}y^{1}$$

$(-1, 0)$ $(0, -1)$ $(1,0)$ $(0,1)$ $(1,1)$

$g = x^{-1} - y^{-1} + 3x - 2y + xy$のNewton図形

こうして得られたベクトル $(-1, 0)$; $(0,-1)$; $(1,0)$; $(0,1)$; $(1,1)$を二次元のxy平面にプロットして、convex hullをとると、こうなります。



Newton図形のサンプル 3

次のTropical多項式のNewton図形を作ってみましょう。

$$p = 1 \odot x_1^2 \oplus 1 \odot x_2^2 \oplus 2 \odot x_1 x_2 \oplus 2 \odot x_1 \oplus 2 \odot x_2 \oplus 2$$

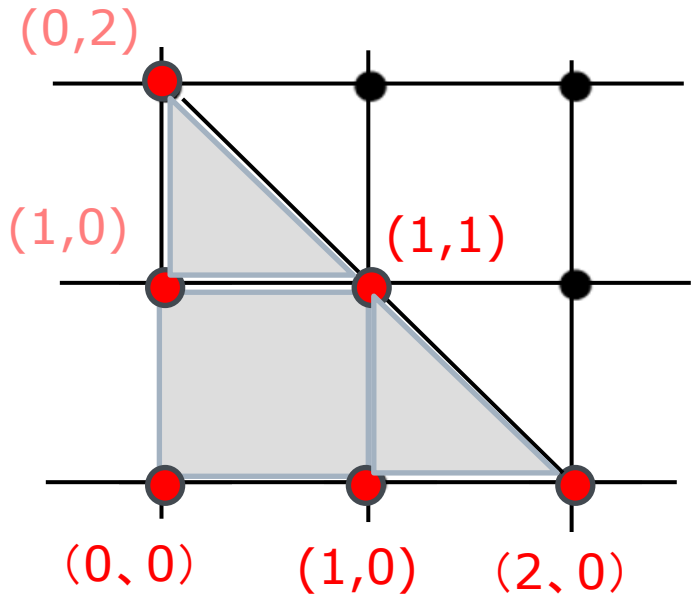
(2,0) (0,2) (1,1) (1,0) (0,1) (0,0)

それぞれの単項式に対応するベクトルは、上のようになります。
（ここは、暗算で）

ここで得られたベクトル
(2,0); (0,2); (1,1); (1,0); (0,1); (0,0);
をプロットしてみましょう。

$p = 1 \odot x_1^2 \oplus 1 \odot x_2^2 \oplus 2 \odot x_1 x_2 \oplus 2 \odot x_1 \oplus 2 \odot x_2 \oplus 2$
のNewton図形

次の灰色部分が、pのNewton図形となります。

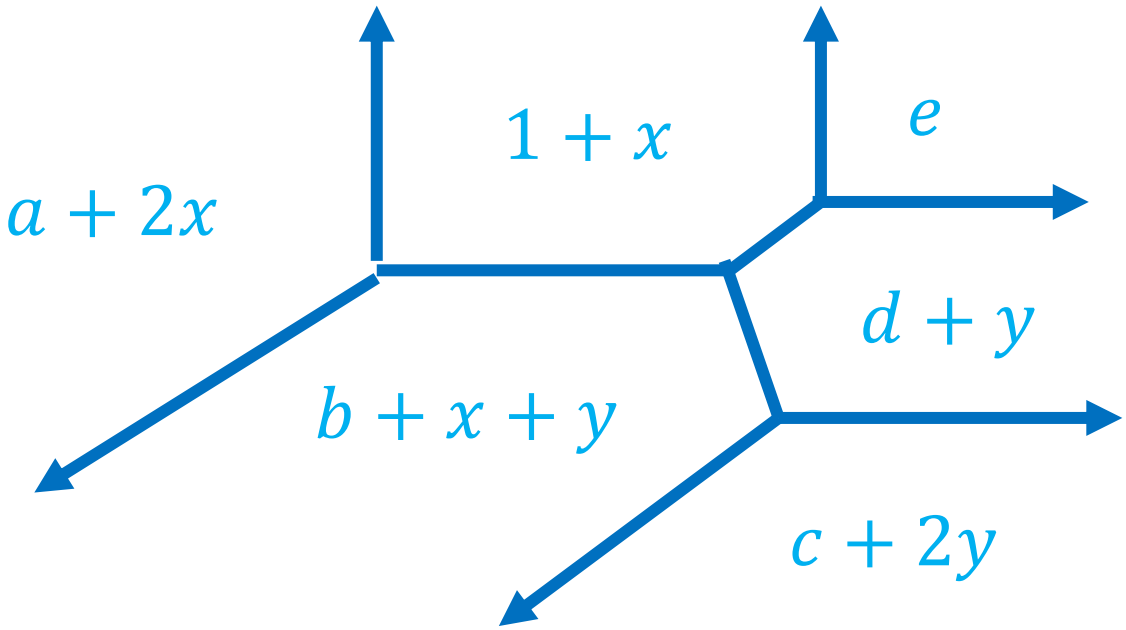


pのHypersurface V(p)と pのNewton図形との対応

ここでは、pのHypersurface V(p)とpのNewton図形とのあいだに対応関係があることを述べようと思います。

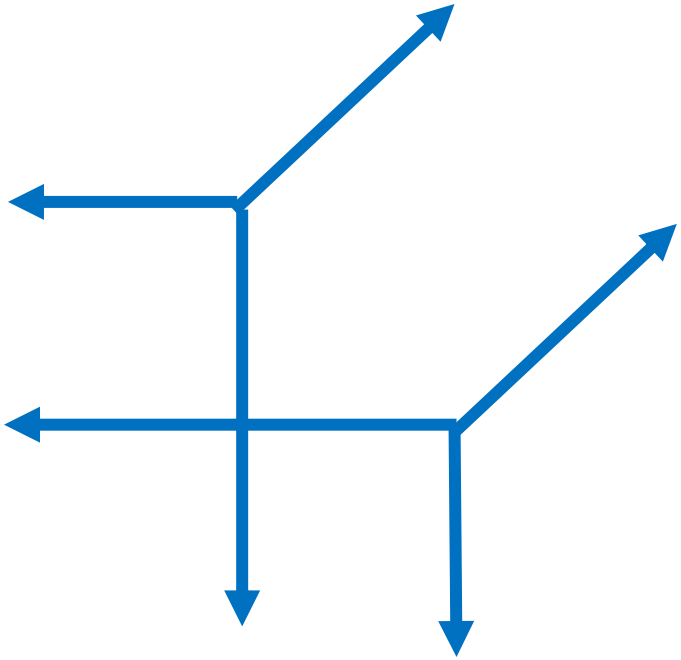
残念ながら、正確な証明ではなく、イメージを持ってもらうことが目標です。

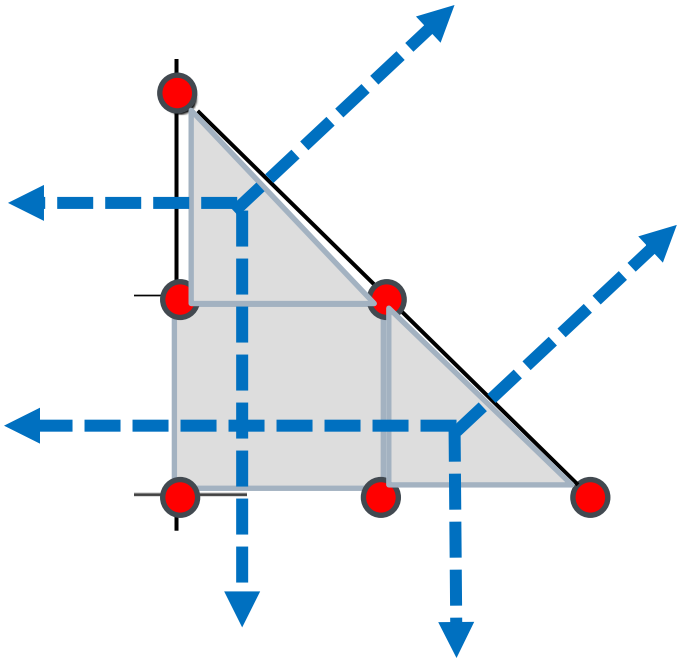
前回、$p(x,y) = a \odot x^2 \oplus b \odot xy \oplus c \odot x^2 \oplus d \odot y \oplus e$ の
Hypersurface V(p)が次のような形になることを見てきました。



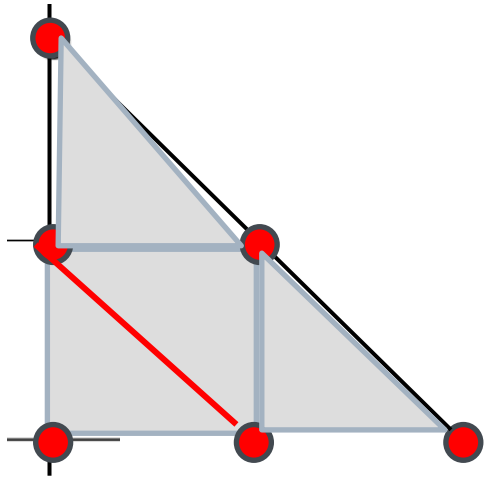
この図形の向きを変え、簡単にして、次のような図形を考えます。

この図形は次のように、先に見たpのNewton図形と対応付けることができます。

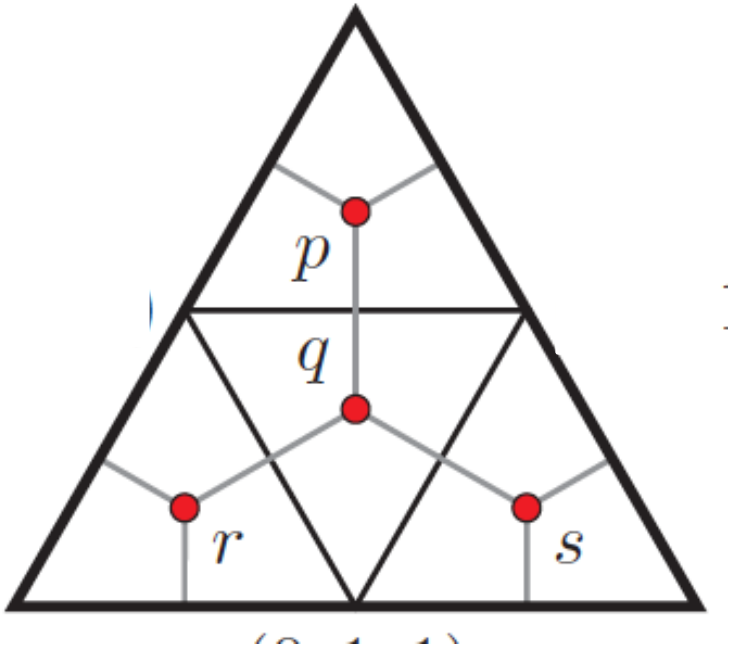




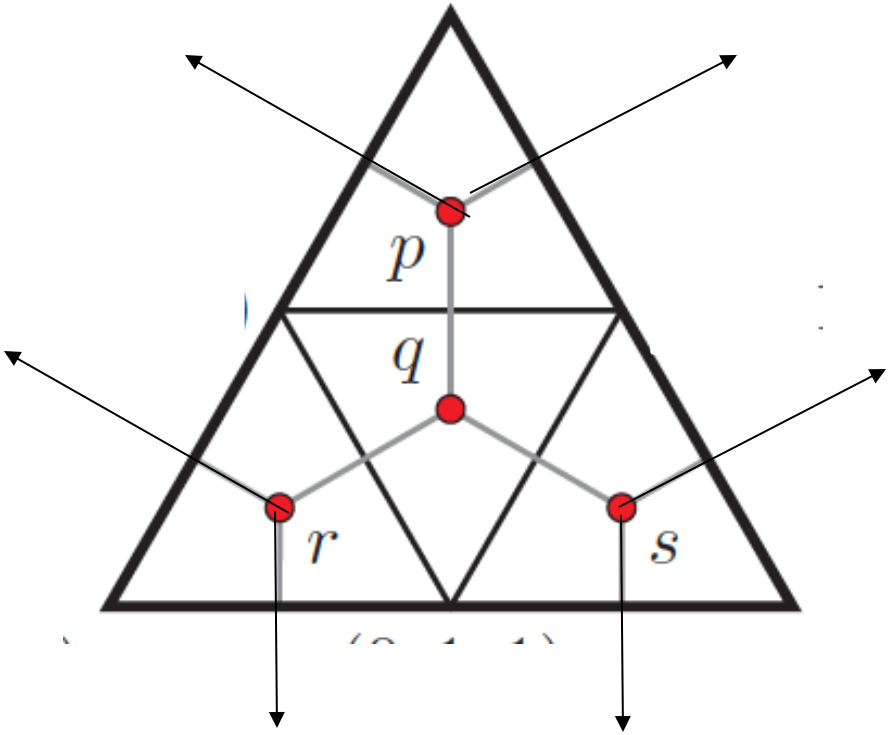
V(p)とNewton図形の対応を、もっと明確に見るには、Newton図形を5つの三角形に分割するのがいいと思います。



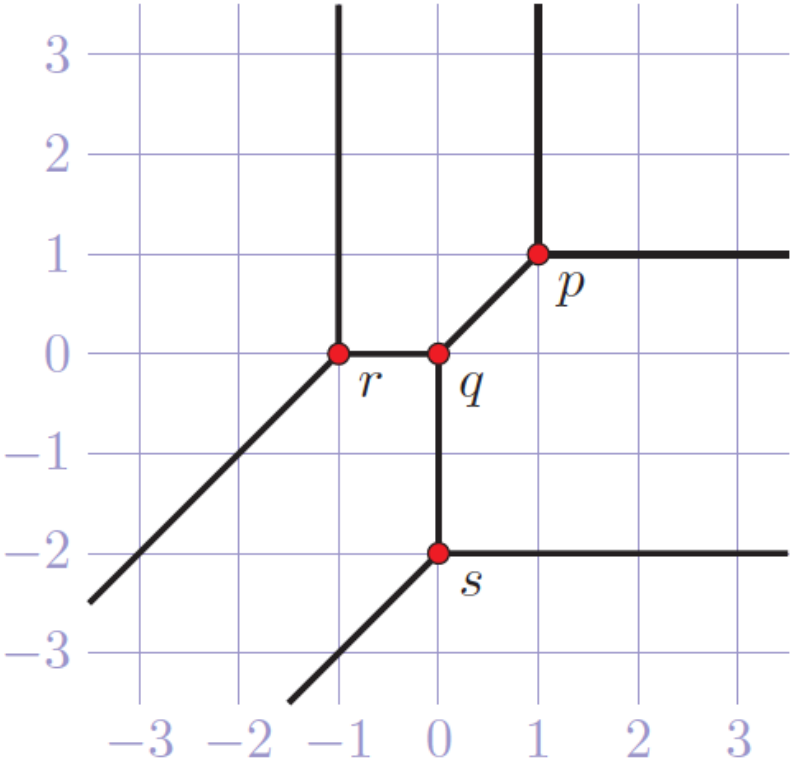
V(p)とNewton図形の対応を、もっと明確に見るには、Newton図形を5つの三角形に分割するのがいいと思います。



V(p)とNewton図形の対応を、もっと明確に見るには、Newton図形を5つの三角形に分割するのがいいと思います。

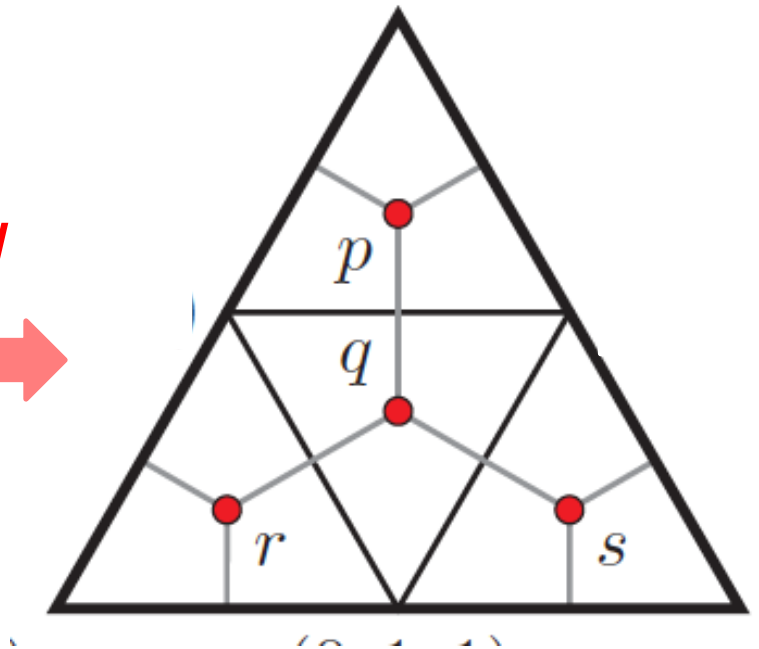


Hypersurface V(p)と Newton図形とは、dual である



$V(p)$

dual



$Newton$図形


Part 3
ニューラル・ネットワークの数理
ニューラル・ネットワークの数理 -- Tropical代数入門

Part 3　ニューラル・ネットワークの数理

- Zhangたちが考えたこと
- mini-plusとmax-plus
- DNNの数学モデルを準備する
- convexな関数の差
- ニューラル・ネットとTropical有理写像
- HypersurfaceとNewton polygonの対応
- Tropical幾何とニューラル・ネットワーク


ニューラル・ネットワークの数理
Zhangたちが考えたこと
ニューラル・ネットワークの数理 -- Tropical代数入門

このセッションでは、論文 “Tropical geometry of deep neural networks” の著者たちが考えたことの概要を、見ていきたいと思います。

「Deep Neural Networkについての我々の理論的理解は、まだ不完全」

まず、彼らの基本的な問題意識を見ておきましょう。

「ディープ・ニューラル・ネットワークは、人工知能、コンピュータ・ヴィジョン、音声認識、自然言語処理など、さまざまな分野のアプリケーションで大きな成功を収め、最近脚光を浴びている。

しかしながら、ディープ・ニューラル・ネットワークの有効性に関する我々の理論的理解がまだ不完全であることもよく知られている。」

例えば、どのような問題があるのでしょう？

「ニューラルネットワークは、学習段階で見たことのないデータに対してもよく汎化する。そのことから、ネットワークには暗黙の秩序化(regularization)の能力がある可能性が示唆されている。

しかし、従来の複雑度の尺度では、この現象を説明できない。Deep Neural Networkの汎化能力を生み出すこの暗黙の秩序化を理解することは、依然として課題である。」

研究の目的

「我々の研究の目的は、ニューラルネットワークとTropical幾何学のつながりを確立することである。それがDeep Neural Networkの仕組みに光を当てることを期待している。

Tropical幾何学は代数幾何学の新領域であり、ここ10年で爆発的な成長を遂げた。ただ、純粋数学以外では比較的無名の分野である。」

成果

「我々は、ReLUを持つfeed forward neural networkに注目し、それらがTropical代数における有理関数、すなわち、2つのTropical多変数多項式f、gの商の類似物であることを示した。

標準多項式や三角多項式では、有理近似（1つの多項式の代わりに2つの多項式の比で近似すること）が次数を上げることなく近似の質を大幅に向上させることが知られている。

ReLU Neural Networkは2つのTropical多項式のTropical比、すなわちTropical有理関数である。

より正確には、Neural Network を関数νと見なせば、各νはTropical有理写像である。

実際、我々は次のことを示す。

ReLU 線形ユニットと整数の重みを持つfeed forward neural networkで表現される関数の族は、まさにTropical有理写像の族である。」

これから導かれること

この関数族にはセミ・フィールド構造が存在することが直ちに導かれる。さらに重要なことは、このことがNeural NetworkとTropical幾何学の架け橋となり、Neural Networkをよく研究されているトロピカル幾何学の対象として見ることができるということである。

この洞察により、Neural Networkの線形領域間の境界をTropical超曲面に密接に関連付けることができる。

こうした洞察により、

分類問題におけるニューラル・ネットワークの決定境界をTropical超曲面として研究することが容易になる。

さらに、ニューラルネットワークの複雑さを表す線形領域の数は、ニューラルネットワークのトロピカル有理表現に関連するポリトープの頂点の数によって境界を定めることができる。

最後に、1つの隠れ層を持つニューラルネットワークは、より深いネットワークの構成要素として機能するゾノトープによって完全に特徴付けることができる。

結論と展望

我々は、ReLUを持つfeed forward neural networkは、些細なことを除けば、Tropicalな有理マップに過ぎないと主張する。

それを理解するためには、しばしば関連するトロピカル幾何学を理解する必要がある。

ニューラル・ネットワークの決定境界、線形領域、深さが表現力に与える影響などに関する疑問は、トロピカル超曲面、ニュートン多角形の双対細分割、ゾノトープから構成される多角形などに関する疑問に変換することができる。

代数幾何学の新しい一分野として、Tropical幾何学の新しさは、代数と幾何の両方、およびそれらの間の相互作用に由来する。

数学の他の多くの分野ともつながりがある。特に、Tropical 線形代数とTropical凸幾何学の二つのTropical 数学がある。

我々は、この豊かな主題のほんの一部にしか触れていないことを強調してもしきれない。このトロピカルな角度からのさらなる研究によって、Deep Neural Networkの他の謎が解明されることを期待している。


ニューラル・ネットワークの数理
mini-plusとmax-plus
ニューラル・ネットワークの数理 -- Tropical代数入門

min-plusとmax-plus
二つのTropical数学

以前、Tropical数学での掛け算と足し算の定義を次のように紹介しました。

$$x \odot y = x + y$$

$$x \oplus y = \min(x, y)$$

あるいは

$$x \oplus y = \max(x, y)$$

掛け算 $\odot$ の定義は一つですが、足し算 $\oplus$ の定義が二つあります。

実は、二つの異なるTropical数学があるのです。もう少し、キチンと書いてみましょう。

$$x \odot y = x + y$$
$$x \oplus y = \min(x, y)$$

あるいは

$$x \odot y = x + y$$
$$x \oplus y = \max(x, y)$$

前者の定義で与えられる Tropical数学を min-plus
後者の定義で与えられる Tropical数学を max-plus
と呼びます

これまでのセッションは、すべて、min-plusのTropical数学をl利用してきたのですが、Zhangらの論文は、max-plusのTropical数学を使っています。

このセッション以降、Zhangらの論文の紹介をするので、特に断りがなければ、max-plusのTropical数学を用います。

なぜ、max-plus?

ニューラル・ネットワークとTropical代数との同型性を論ずる Zhangらが、なぜ、min-plus ではなく max-plus を選んだのでしょう？

それは、ニューラル・ネットワークの代表的なactivator である、ReLUをどう表現するかを考えれば、わかりやすいと思います。

max-plusとReLU

ReLU(x)は、$x < 0$の時 0を返し、$x \geq 0$ の時、xを返します。

これは、max-plusの和 $\oplus$ を使って、次のように表現できます。

$$ReLU(x) = \max\{x, 0\} = x \oplus 0$$

ReLUの表現に、max-plus は向いています。

max-plus 計算練習

max-plusに慣れるため、ちょっと計算を練習してみましょう。
Toropicalな世界での計算はピンクで、普通の世界での計算はライトブルーで表しています。

$4 \odot 5 = 4 + 5 = 9$

$3 \oplus 100 = \max(3, 100) = 100$

$7 \odot 5 \oplus 7 \odot 6$
$= (7 + 5) \oplus (7 + 6) = 12 \oplus 13 = \max(12,13) = 13$

$7 \odot (5 \oplus 6)$
$= 7 \odot \max(5, 6) = 7 \odot 6 = 7 + 6 = 13$

max-plusでの加算と乗算の可換性

加算と乗算の可換性とは、普通の世界では、$x+y=y+x, xy=yx$ であることを言います。

$x \oplus y = y \oplus x$ の証明。

$$x \oplus y = \max(x, y) = \max(y, x) = y \oplus x$$

$x \odot y = y \odot x$ の証明

$$x \odot y = x + y = y + x = y \odot x$$

max-plusでの加算と乗算の分配律

加算と乗算の分配律とは、普通の世界では$x(y+z)=xy+xz$であることを言います。

$x \odot (\mathrm{y} \oplus \mathrm{z}) = x \odot y \oplus \mathrm{x} \odot y$ の証明。

$$
\begin{aligned}
x \odot (\mathrm{y} \oplus \mathrm{z}) &= \mathrm{x} + max(\mathrm{y}, \mathrm{z}) \\
&= \max(\mathrm{x}+\mathrm{y}, \mathrm{x}+\mathrm{z}) \\
&= x \odot y \oplus \mathrm{x} \odot y
\end{aligned}
$$

max-plusでの加算の零元

普通の世界では、任意のxについて $x + 0 = x$ ですので、加算の零元は $\mathbf{0}$です。

max-plusで、任意のxについて、$x \oplus \mathrm{z} = \mathrm{x}$ すなわち、
$x \oplus \mathrm{z} = max(\mathrm{x}, \mathrm{z}) = \mathrm{x}$ を満たすzは存在するでしょうか？

$$z = -\infty$$

とすると、どんなxも$-\infty$ よりは大きいので、max-plusでの加算の零元は、$-\infty$ とおけばいいことがわかります。

max-plusでは、「数」に$-\infty$を加えて、それを加算についての零元とします。

max-plusでの乗算の単位元

普通の世界では、任意のxについて $x\times 1 = x$ ですので、乗算の単位元は 1です。

max-plusで、任意のxについて、$x \odot \mathrm{z} = \mathrm{x}$ すなわち、
$x \odot \mathrm{z} = \mathrm{x} + \mathrm{z} = \mathrm{x}$ を満たすzは存在するでしょうか？

少し意外ですが、$z = 0$ がその条件を満たすことがわかります。

max-plusでの乗算の単位元は、0になります。

$$x \odot 0 = 0 \odot x = x$$

が成り立ちます。これは、min-plusと同じです。

max-plusでのベクトル･行列演算

まず、二つのベクトルの内積について見てみましょう。

$$(u_1, u_2, u_3) \otimes (v_1, v_2. v_3)^T$$

$$= u_1 \otimes v_1 \oplus u_2 \otimes v_2 \oplus u_3 \otimes v_3$$
$$= \max(u_1 \otimes v_1, u_2 \otimes v_2, u_3 \otimes v_3)$$

$$= \max(u_1 + v_1, u_2 + v_2, u_3 + v_3)$$

もう一つの、行列を形成する二つのベクトルの積は、こうなります。

$$(u_1, u_2, u_3)^T \otimes (v_1, v_2. v_3)$$

$$= \begin{pmatrix} u_1 \otimes v_1 & u_1 \otimes v_2 & u_1 \otimes v_3 \\ u_2 \otimes v_1 & u_2 \otimes v_2 & u_2 \otimes v_3 \\ u_3 \otimes v_1 & u_3 \otimes v_2 & u_3 \otimes v_3 \end{pmatrix}$$

$$= \begin{pmatrix} u_1 + v_1 & u_1 + v_2 & u_1 + v_3 \\ u_2 + v_1 & u_2 + v_2 & u_2 + v_3 \\ u_3 + v_1 & u_3 + v_2 & u_3 + v_3 \end{pmatrix}$$

いくつかの計算例

ベクトルのスカラー倍

$$
\begin{aligned}
&2 \otimes (3, -7, 6) \\
&= (2 \otimes 3, 2 \otimes -7, 2 \otimes 6) \\
&= (2 + 3, 2 + (-7), 2 + 6) \\
&= (5, -5, 8)
\end{aligned}
$$

ベクトルの内積

$$
\begin{aligned}
&(-\infty, 0, 1) \otimes (0, 1, -\infty)^T \\
&= (-\infty \otimes 0 \oplus 0 \otimes 1 \oplus 1 \otimes -\infty) \\
&= (-\infty \oplus 1 \oplus -\infty) \\
&= max(\infty . 1, \infty) \\
&= 1
\end{aligned}
$$

行列の和

$$\begin{pmatrix} 3 & 3 \\ 0 & 7 \end{pmatrix} \oplus \begin{pmatrix} 4 & 1 \\ 5 & 2 \end{pmatrix}$$

$$= \begin{pmatrix} 3 \oplus 4 & 3 \oplus 1 \\ 0 \oplus 5 & 7 \oplus 2 \end{pmatrix}$$

$$= \begin{pmatrix} \max(3,4) & max(3,1) \\ max(0,5) & max(7,2) \end{pmatrix}$$

$$= \begin{pmatrix} 4 & 3 \\ 5 & 7 \end{pmatrix}$$

行列の積

$$\begin{pmatrix} 3 & 3 \\ 0 & 7 \end{pmatrix} \otimes \begin{pmatrix} 4 & 1 \\ 5 & 2 \end{pmatrix}$$

$$= \begin{pmatrix} 3 \otimes 4 \oplus 3 \otimes 5 & 3 \otimes 1 \oplus 3 \otimes 2 \\ 0 \otimes 4 \oplus 7 \otimes 5 & 0 \otimes 1 \oplus 7 \otimes 2 \end{pmatrix}$$

$$= \begin{pmatrix} 7 \oplus 8 & 4 \oplus 5 \\ 4 \oplus 12 & 1 \oplus 9 \end{pmatrix}$$

$$= \begin{pmatrix} \max(7,8) & max(4,5) \\ max(4,12) & max(1,9) \end{pmatrix}$$

$$= \begin{pmatrix} 8 & 5 \\ 12 & 9 \end{pmatrix}$$


ニューラル・ネットワークの数理
DNNの数学モデルを準備する

DNNの数学モデルを準備する

このセッションでは、ReLUをactivatorとするfeed-forward network (DNN)と、max-plusのTropical有理関数の同型性を証明するためのいくつかの前提を確認します。

第一に、max-plusのTropical 多項式の性質を見る上で、max-plusでの、冪乗の性質を見ておきます。

第二に、Part 1 で見たDNNを関数の合成として捉える見方を、あらためて確認します。

第三に、Zhangらは、証明にあたってDNNの特徴づけにある条件を設定しているのですが、それを確認します。

max-plusでの冪乗の性質

Tropical数学での冪乗 $x^{\odot a}$をx^aで表すことにします。次の性質が成り立ちます。

- $x, y \in \mathbb{R}$で、$a \in \mathbb{R}$ かつ$a \geq 0$の時

$$(x \oplus y)^a = (\max(x, y))^a = \max(x^a, y^a) = x^a \oplus y^a$$

ただし、$a < 0$の時、一般には、$(x \oplus y) \neq x^a \otimes y^a$ である。

- $x, y \in \mathbb{R}$で、$a \in \mathbb{R}$の時

$$(x \odot y)^a = x^a \odot y^a$$

- $x \in \mathbb{R}$の時

$$x^0 = 0$$

- $x \in \mathbb{R}$で、$a, b \in \mathbb{N}$の時

$$(\mathrm{x}^{\mathrm{a}})^{\mathrm{b}} = x^{ab}$$

- $x \in \mathbb{R}$で、$a, b \in \mathbb{Z}$の時

$$x^a \odot x^b = x^{a+b}$$

- $x \in \mathbb{R}$で、$a, b \in \mathbb{Z}$の時

$$x^a \oplus x^b = x^b \odot \left(x^{a-b} \oplus 0\right) = x^a(0 \oplus x^{b-a})$$

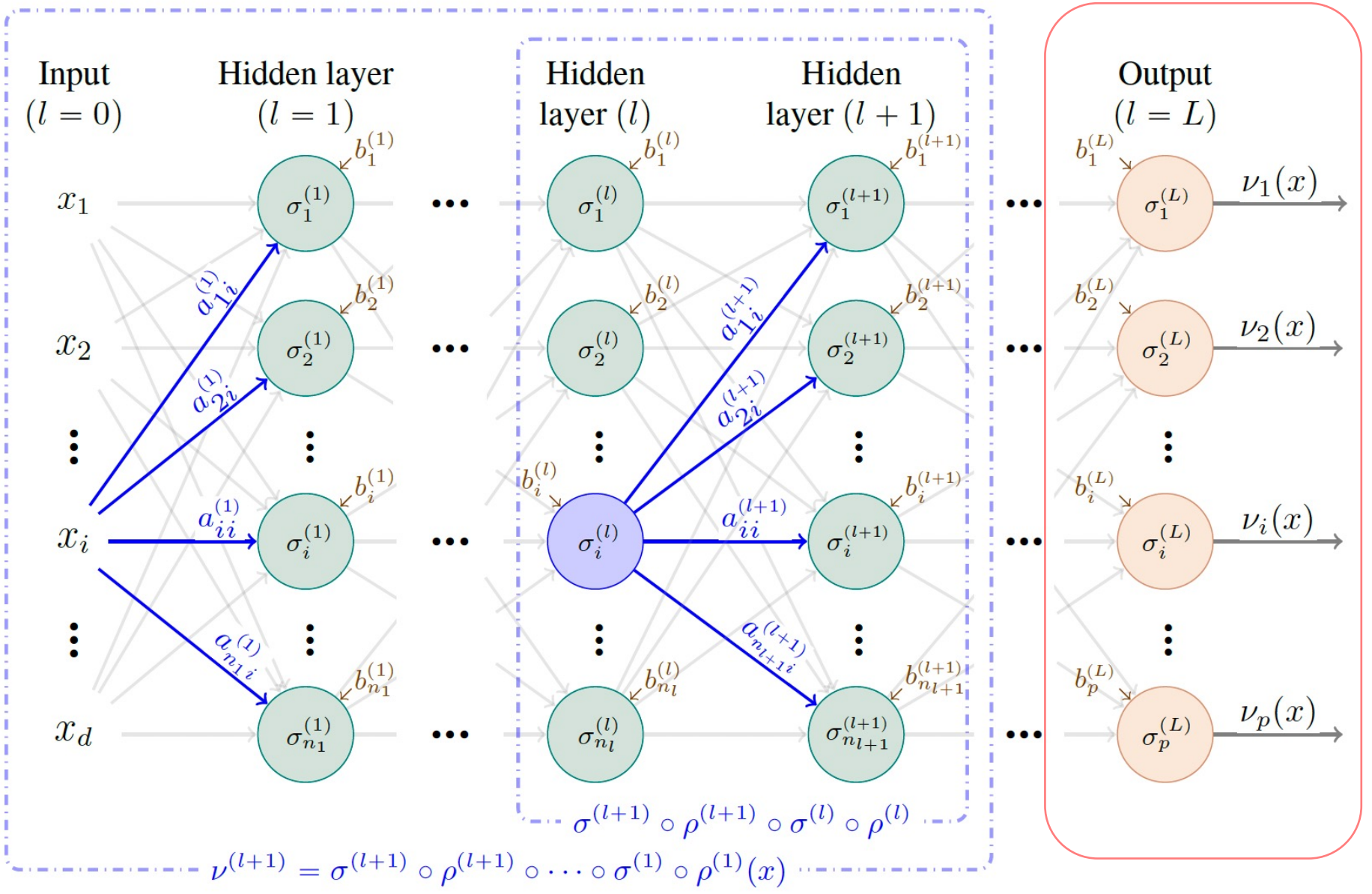
基本的な式のまとめ

ここでは、Part 1の「Deep Neural Netを関数の合成で表わす」の内容を確認します。

- pdf: https://drive.google.com/file/d/1OKEC5heaEFUm2c2prn_sY0lLcyg6gAty/view?usp=sharing
- video: https://youtu.be/GKVWlwHbjmA?list=PLQIrJ0f9gMcOI0QVy2PA1LT_IuuLp97Qs

を、参照ください。

DNNのグラフ


Input
$(l = 0)$
Hidden layer
$(l = 1)$
Hidden
layer (l)
Hidden
layer $(l + 1)$
Output
$(l = L)$
x_1
x_2
x_i
x_d
$\nu_1(x)$
$\nu_2(x)$
$\nu_i(x)$
$\nu_p(x)$
$\sigma^{(l+1)} \circ \rho^{(l+1)} \circ \sigma^{(l)} \circ \rho^{(l)}$
$\nu^{(l+1)} = \sigma^{(l+1)} \circ \rho^{(l+1)} \circ \cdots \circ \sigma^{(1)} \circ \rho^{(1)}(x)$

関数の合成としてのDNN $\nu: \mathbb{R}^d \to \mathbb{R}^p$

抽象的に捉えれば、L個の層からなる feed-forward network (DNN)は、次の関数の合成で与えられる写像 $\nu: \mathbb{R}^d \to \mathbb{R}^p$ である。

$$\nu = \sigma^{(L)} \circ \rho^{(L)} \circ \sigma^{(L-1)} \circ \rho^{(L-1)} \circ \cdots \circ \sigma^{(1)} \circ \rho^{(1)}$$

activation前の関数 $\rho^{(1)}, \ldots, \rho^{(L)}$は決定されるべきaffine変換で(線形変換)で、activation関数 $\sigma^{(1)}, \ldots, \sigma^{(L)}$は事前に選ばれて固定されている。

l 番目の層の出力 $\nu^{(l)}$

$width$ すなわち l 番目の層のノードの数を n_l で表す。ここに、$l = 1, \dots, L-1$ である。ネットワークの入力と出力の次元を、それぞれ $n_0 := d, n_L := p$ と置く。この時、l 番目の層からの出力$\nu^{(l)}$は、次の式で表される。

$$\nu^{(l)} = \sigma^{(l)} \circ \rho^{(l)} \circ \sigma^{(l-1)} \circ \rho^{(l-1)} \circ \cdots \circ \sigma^{(1)} \circ \rho^{(1)}$$

すなわち、写像 $\nu^{(l)}: \mathbb{R}^d \to \mathbb{R}^{n_l}$ である。

便宜的に、$\nu^{(0)}(x) := x$ とする。

パラメーター

affine関数 $\rho^{(l)}: \mathbb{R}^{n_{l-1}} \to \mathbb{R}^{n_l}$ は、重み行列 $A^l \in \mathbb{Z}^{n_l \times n_{l-1}}$とバイアス・ベクトル $b^{(l)} \in \mathbb{Z}^{n_l}$ によって与えられる。

$$\rho^{(l)}\left(\nu^{(l-1)}\right) := A^{(l)}\nu^{(l-1)} + b^{(l)}$$

A^l の(i,j)成分を $a_{ij}^{(l)}$ で、$b^{(l)}$ のi番目の成分を $b_i^{(l)}$ で表す。

これらは、l層のパラメータを構成する。

activator と score 関数

入力ベクトル $x \in \mathbb{R}^{n_l}$について、$\sigma^{(l)}$は成分ごとに作用すると考えるので、$\sigma: \mathbb{R}^{n_l} \to \mathbb{R}^{n_l}$ である。

ネットワークの最終出力 $\nu(x)$ は、アプリケーションによって異なるscore関数 $s: \mathbb{R}^{p} \to \mathbb{R}^{m}$に渡される。m個のカテゴリーへの分類なら、sにはsoft-maxやsigmoidといった関数が選ばれる。

score関数は、しばしば、ニューラル・ネットワークの最終層と見なされるのだが、それは便宜的なことで、我々はそれを仮定していない。

ZhangらがDNNに課した条件

Zhangらは、feed-forward network に次のような条件を課しています。

a. 重み行列 $A^{(1)}, \ldots, A^{(L)}$は、整数値を取る。

b. バイアス・ベクトル $b^{(1)}, \ldots, b^{(L)}$は、実数値を取る。

c. activation関数 $\sigma^{(1)}, \ldots, \sigma^{(L)}$は、次の形をしている。
$\sigma^{(l)}(x) := \max\{x, t^{(l)}\}$,
この$t^{(l)} \in (\mathbb{R} \cup \{-\infty\})^{n_l}$は、閾値ベクトルと呼ばれる。

条件 a について

条件 bは、一般的なものですが、条件 a は、少し強いものです。
これについて、Zhang らは、次のような説明を行なっています。

- 実数値の重みは、望む精度まで有理数で近似できる。
- この有理数値の重みに、分母たちの最小公倍数をかければ、「分母を払う」ことができる。
- 重みとバイアスを、同じ正の定数でスケールさせることは、ニューラル・ネットワークの仕事の障害にはならない。

条件 c と ReLUネットワークについて

条件 c を満たす activator には、ReLU ($t^{(l)}=0$)と、特殊なものでは同一射($t^{(l)}=-\infty$)が含まれています。

この研究では、ReLUネットワークをニューラルネットワークの最も単純で標準的なモデルとみなし、そこから特定のタスクでより効果的な別のモデルを導き出す。

我々が求めているのは一般的な理論的洞察であり、特定の実用的効果ではないことを考えると、この最も単純なケースに限定することは理にかなっている。

さらに、ReLUネットワークは、より広範なニューラルネットワークに共通する最も重要な要素（および神秘的な要素）のいくつか（例えば、普遍的な近似、指数関数的な表現力）をすでに具現化している。


ニューラル・ネットワークの数理
convexな関数の差
ニューラル・ネットワークの数理 -- Tropical代数入門

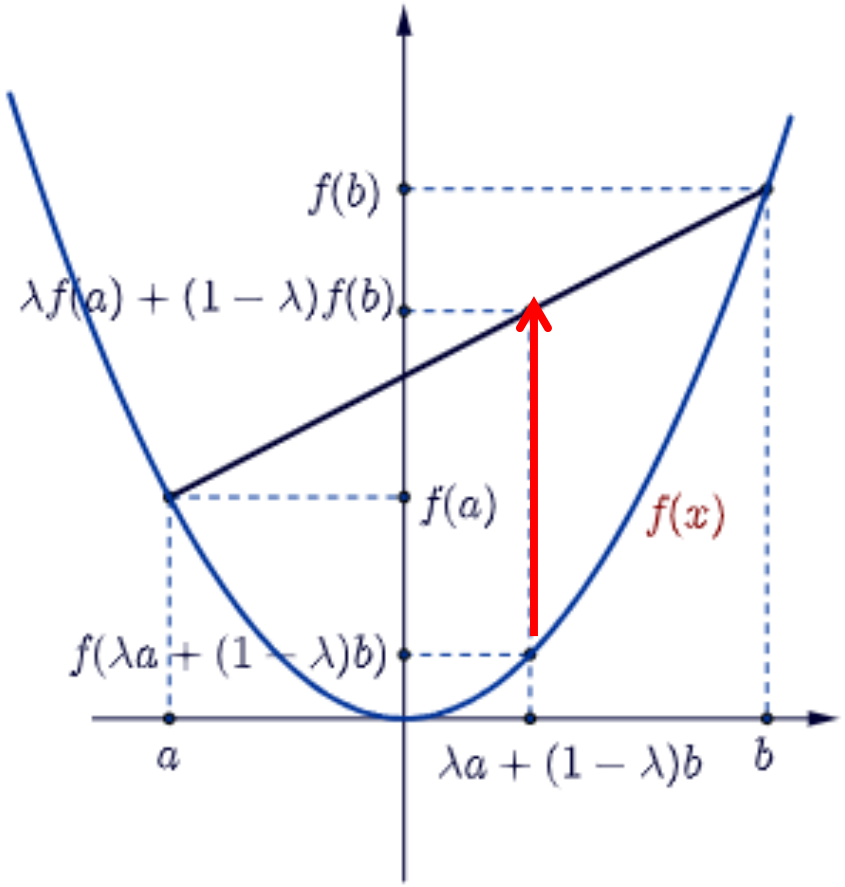
このセッションでは、convexな関数の性質を取り上げます。
Tropical多項式の定める関数もconvex関数です。

ただ、DNNの出力は、一般にはconvexではありません。

重要なのは、任意の連続関数は、二つのconvexな関数の差として近似できるということです。

DNNの数学的モデルとして、Tropical有理関数が登場するのには、そうした理由があります。

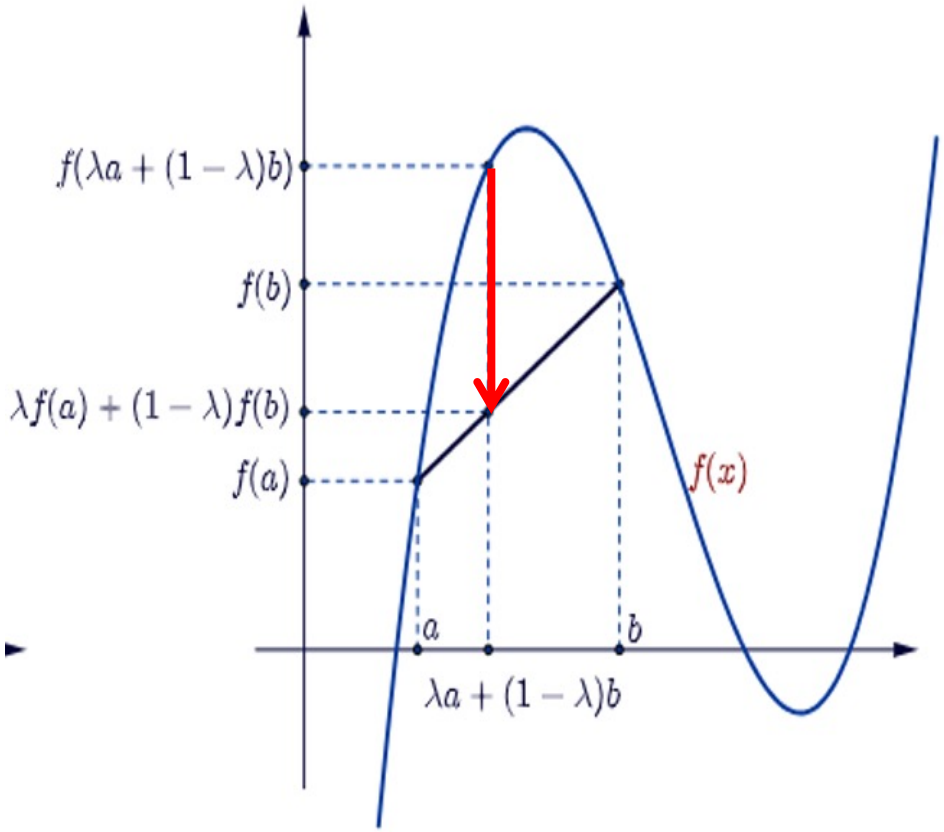
convex(下に凸)な関数



$$f(\lambda a + (1-\lambda)b \leq \lambda f(a) + (1-\lambda) f(b)$$

$$f(\frac{a+b}{2}) \leq \frac{1}{2}(f(a) + f(b))$$

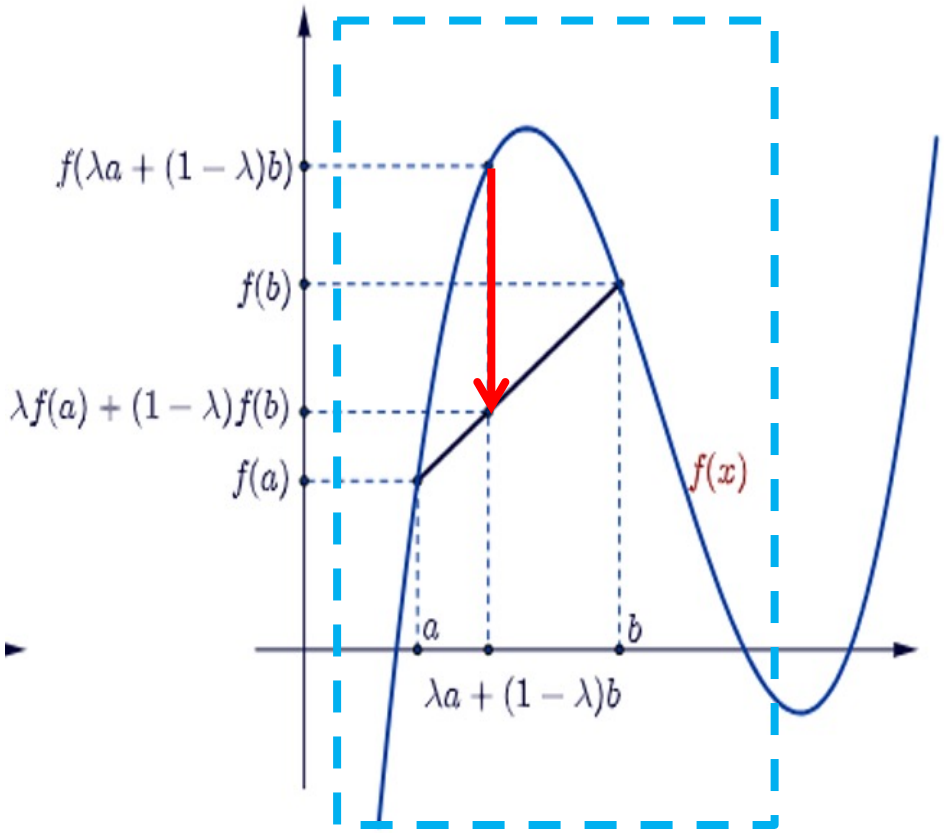
non-convex function



$$f(\lambda a + (1-\lambda)b \geq \lambda f(a) + (1-\lambda) f(b)$$

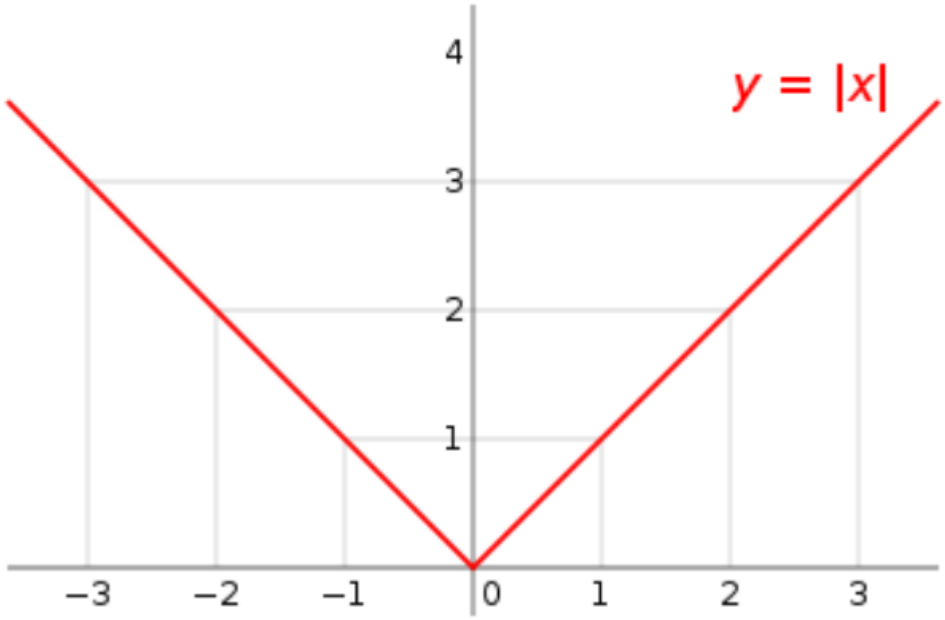
non-convex function

concave（上に凸）



$$f(\lambda a + (1-\lambda)b \geq \lambda f(a) + (1-\lambda) f(b)$$

$f(x) = |x|$ は、convex



convex関数の例

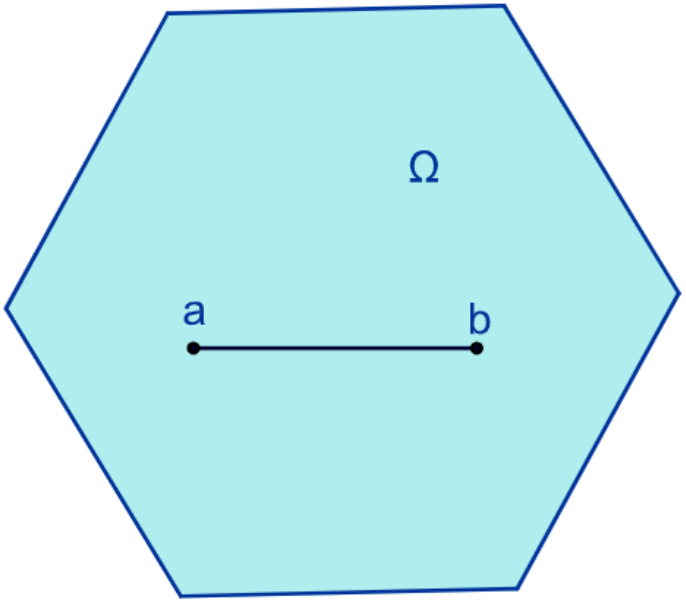
$f(x) = x^2 + ax + c$

$f(x) = e^x$

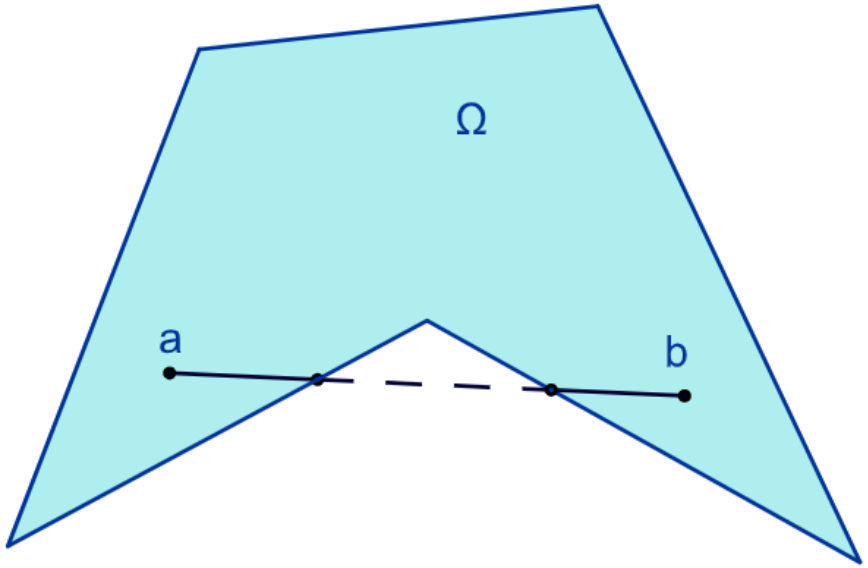
$f(x) = -\ln(x)$

convex set と non convex set

convex



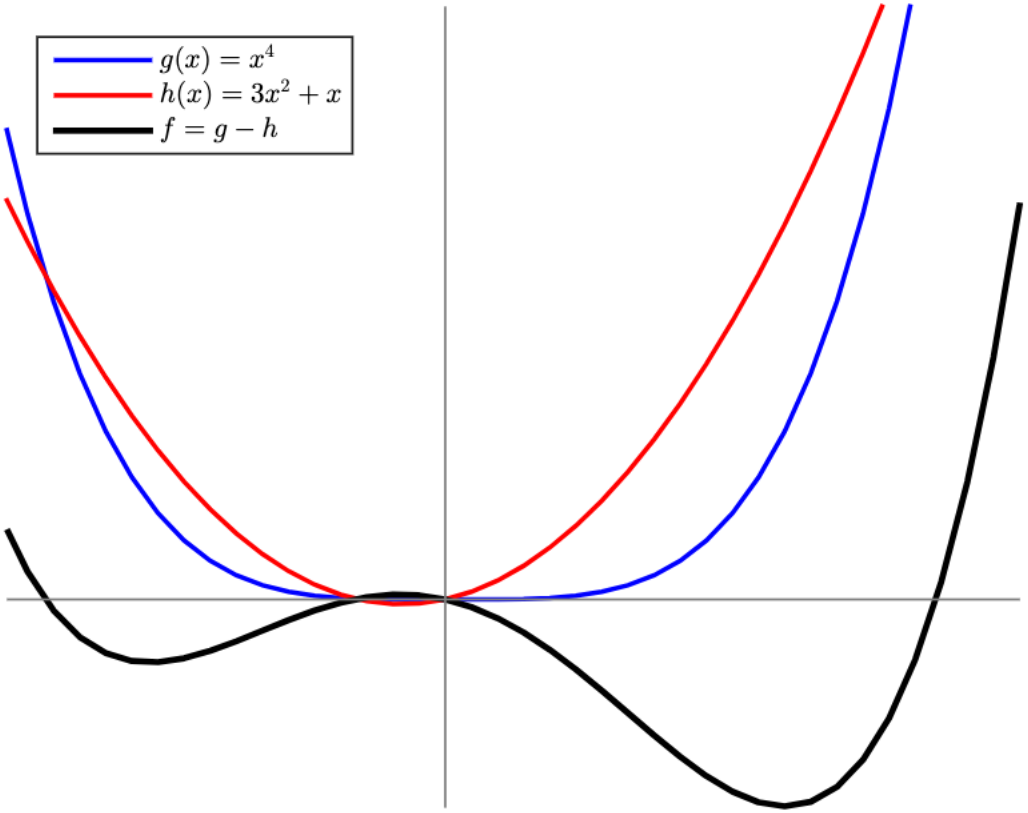
non-convex



二つのconvex関数の差の例

$$f = g - h$$

We call $g - h$ a DC decomposition of f



DC function

dc関数とは、2つのconvex関数の差として表現できる関数のことである。dc集合とは、2つのconvex集合の差として表現できる集合のことである。

dc関数とdc集合のいくつかの重要な性質について論じる。特に、

(1) 任意の連続関数は、dc関数によって望みどおりの精度で近似できる
(2) 任意の$\mathbb{R}^n$上のclosedな集合は、$\mathbb{R}^{n+1}$上のdc集合の射影である。
(3) dc関数のクラスは、線形操作や有限の上包または下包の下で安定である。

後者の性質により、任意の有限なdc不等式の系は、等価的に単一のdc不等式として書き直すことができる。

このことは、線形最適化問題をいくつかの基本的なクラスに分類することを可能にし、体系的な研究に非常に便利である。

次に、どの関数が線形であるか、線形関数の効果的な表現をどのように見つけるか、線形関数の最小化子をどのように認識するか、について議論する。

Convex Analysis and Global Optimization
https://link.springer.com/chapter/10.1007/978-3-319-31484-6_4

Tropical 有理関数

論文の「定義 2.4」

Tropical有理関数は、二つのTropical 多項式 $f(x), g(x)$の Tropical商である。それは、二つのTropical 多項式 $f(x), g(x)$ の「差」である。

$$f(x) - g(x) = f(x) \oslash g(x)$$

fとgがTropical多項式関数の時、$f \oslash g$で、Tropical 有理関数を表す。

Tropical多項式とTropical有理関数の関係

d-変数のTropical多項式を $\mathbb{T}[x_1, \ldots, x_d]$ で表し、
d-変数のTropical有理関数を $\mathbb{T}(x_1, \ldots, x_d)$ で表す。

Tropical多項式 $f = f \oslash 0$ であるので、Tropical多項式は、
Tropical有理関数の、特別の場合である。

$$\mathbb{T}[x_1, \ldots, x_d] \subset \mathbb{T}(x_1, \ldots, x_d)$$

よって、Tropical有理関数について成り立つ命題は、すべて
Tropical多項式についても成り立つ。

Tropical有理関数は、DC function

d-変数のTropical多項式は、$f: \mathbb{R}^d \to \mathbb{R}$ である関数 $f(x)$ を定義するが、この $f(x)$は convex関数である。なぜなら、convex関数について maxと和をとる操作は、convex性を保存するからである。

d-変数のTropical有理関数 $f \oslash g$: $\mathbb{R}^d \to \mathbb{R}$ は、二つのconvex関数 f, g の差であるので、DC functionである。

ベクトル値をとるTropical有理関数

論文の「定義 2.5」

$$F: \mathbb{R}^d \to \mathbb{R}^p, x = (x_1, \dots x_d) \longmapsto \Big(f_1(x), \dots, f_p(x)\Big)$$

それぞれの $f_i: \mathbb{R}^d \to \mathbb{R}$ が Tropical多項式のとき、Fを、Tropical多項式写像と呼び、$Pol(d,p)$と表す。

それぞれの $f_i: \mathbb{R}^d \to \mathbb{R}$ が Tropical有理関数のとき、Fを、Tropical有理写像と呼び。$Rat(d,p)$と表す。

$$Pol(d,1) = \mathbb{T}[x_1, \dots, x_d]$$
$$Rat(d,1) = \mathbb{T}(x_1, \dots, x_d)$$

なぜ、Tropical有理写像なのか？

多層のfeed-forward ニューラル・ネットワークは一般にconvexではない。しかし、Tropical多項式は常にconvexである。

ただし、ほとんどのconvexではない関数は2つのconvexな関数の差で表すことができる。それゆえ、多層のfeed-forward ニューラル・ネットワークを、convexな2つのTropical多項式の差、すなわちTropical有理関数であると考えるのが妥当であろう。


ニューラル・ネットワークの数理
ニューラル・ネットとTropical有理写像
ニューラル・ネットワークの数理 -- Tropical代数入門

ReLUを持つfeed-forward neural networkは、Tropical有理写像で表現できる

このセッションでは、ReLUを持つfeed-forward neural networkは、Tropical有理写像、すなわちTropical多項式関数の差で特徴付けられるという、論文の基本的な部分を紹介します。

重み行列Aの分割

$$A = A_+ - A_-$$

行列Aを成分a_{ij}の正負に応じて、二つの行列A_+と$A_=$に分割し、

$$A = A_+ - A_-$$

とする。

行列A_+, A_-の成分をa_{ij}^+, a_{ij}^-とすると、

$$a_{ij}^+ = \max\left(a_{ij}, 0\right), a_{ij}^- = \max(-a_{ij}, 0)$$

とおけばいい。

$$\begin{pmatrix} 3 & -2 & 1 \\ -1 & 2 & 3 \\ 4 & -1 & -3 \end{pmatrix} = \begin{pmatrix} 3 & 0 & 1 \\ 0 & 2 & 3 \\ 4 & 0 & 0 \end{pmatrix} - \begin{pmatrix} 0 & 2 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 3 \end{pmatrix}$$

max関数の変形

$$\max(A, B, C) = \max(A + X, B + X, C + X) - \mathrm{X}$$

$$\begin{aligned}
\rho(x) &= \max\{Ax + b, t\} \\
&= \max\{(A_+ - A_-)x + b, t\} \\
&= \max\{A_+x - A_-x + b, t\} \\
&= \max\{A_+x + b - A_-x, t\} \\
&= \max\{A_+x + b - A_-x + A_-x, t + A_-x\} - A_-x \\
&= \max\{A_+x + b, A_-x + t\} - A_-x
\end{aligned}$$

Proposition 5.1

l層のactivate後の出力$\nu^{(l)}$が、Tropical有理写像として、次のように表されたとする。

$$\nu^{(l)}(x) = F^{(l)}(x) \oslash G^{(l)}(x) = F^{(l)}(x) - G^{(l)}(x)$$

ここに、$F^{(l)}(x), G^{(l)}(x)$ は、Tropical 多項式関数である。

このとき、

(l+1)層のactivate以前の出力$\rho^{(l+1)}$も、
(l+1)層のactivate後のの出力$\nu^{(l+1)}$も、

Tropical有理写像として表現できる。

$\rho^{(l+1)}$ は、Tropical有理写像

$$\rho^{(l+1)}(x) = A\rho^{(l)}(x) + b$$

これは、線形変換の適用である。

$$= A\left(F^{(l)}(x) - G^{(l)}(x)\right) + b$$

重み行列を分割する。

$$= (A_+ - A_-)\left(F^{(l)}(x) - G^{(l)}(x)\right) + b$$

$$= A_+F^{(l)}(x) + A_-G^{(l)}(x) + b - (A_+G^{(l)}(x) + A_-F^{(l)}(x))$$

ここで、

$$A_+F^{(l)}(x) + A_-G^{(l)}(x) + b = H^{(l+1)}(x)$$

$$A_+G^{(l)}(x) + A_-F^{(l)}(x) = G^{(l+1)}(x)$$

とおくと、

$$\rho^{(l+1)}(x) = H^{(l+1)}(x) - G^{(l+1)}(x)$$

$\rho^{(l+1)}$ は、Tropical有理写像である。

$\nu^{(l+1)}(x)$はTropical有理写像

$$\nu^{(l+1)}(x) = \max(\nu^{(l+1)}(x), t)$$

これは、ReLUの適用である。

ここで、先に見た、$\rho^{(l+1)}(x) = H^{(l+1)}(x) - G^{(l+1)}(x)$を使う。

$$= \max(H^{(l+1)}(x) - G^{(l+1)}(x), t)$$

$$= \max(H^{(l+1)}(x) - G^{(l+1)}(x) + G^{(l+1)}(x), \mathrm{t} + G^{(l+1)}(x)) - G^{(l+1)}(x)$$

$$= \max\left(H^{(l+1)}(x), G^{(l+1)}(x) + t\right) - G^{(l+1)}(x)$$

$$\max\left(H^{(l+1)}(x), G^{(l+1)}(x) + t\right) = F^{(l+1)}(x)$$

とおけば、

$$\nu^{(l+1)}(x) = F^{(l+1)}(x) - G^{(l+1)}(x)$$

$\nu^{(l+1)}(x)$はTropical有理写像である。

結果を整理する

$$\rho^{(l+1)}(x) = H^{(l+1)}(x) - G^{(l+1)}(x)$$
$$\nu^{(l+1)}(x) = F^{(l+1)}(x) - G^{(l+1)}(x)$$

ここに、

$$F^{(l+1)}(x) = \max\left(H^{(l+1)}(x), G^{(l+1)}(x) + t\right)$$
$$G^{(l+1)}(x) = A_{+}G^{(l)}(x) + A\_F^{(l)}(x)$$
$$H^{(l+1)}(x) = A_{+}F^{(l)}(x) + A\_G^{(l)}(x) + b$$

Tropical多項式としての表記

$F^{(l)}, G^{(l)}, H^{(l)}$の$i$番目成分を、$f_i^{(l)}, g_i^{(l)}, h_i^{(l)}$として、Tropical多項式として表しておこう。

$$F^{(l+1)}(x) = \max\left(H^{(l+1)}(x), G^{(l+1)}(x) + t\right)$$
$$f_i^{(l+1)} = h_i^{(l+1)} \oplus \left(g_i^{(l+1)} \odot t_i\right)$$

$$G^{(l+1)}(x) = A_+ G^{(l)}(x) + A_- F^{(l)}(x)$$
$$g_i^{(l+1)} = \left\{ \bigodot_j \left(f_j^{(l)}\right)^{a_{ij}^-} \right\} \odot \left\{ \bigodot_j \left(g_j^{(l)}\right)^{a_{ij}^+} \right\}$$

$$G^{(l+1)}(x) = A_+ F^{(l)}(x) + A_- G^{(l)}(x) + b$$
$$h_i^{(l+1)} = \left\{ \bigodot_j \left(f_j^{(l)}\right)^{a_{ij}^+} \right\} \odot \left\{ \bigodot_j \left(g_j^{(l)}\right)^{a_{ij}^-} \right\} \odot b$$

Theorem 5.2
ニューラル・ネットワークのTropicalな特徴づけ

先に見た仮定のもとで、feed-forward neural networkは、関数$\nu: \mathbb{R}^d \to \mathbb{R}^p$である。

それと同等のものは、F, GをTropical多項式写像とする、次のTropical 有理関数である。

$$\nu(x) = F(x) \oslash G(x) = F(x) - G(x)$$

こうして、νは、Tropical 有理写像である。


ニューラル・ネットワークの数理
HypersurfaceとNewton polygonの対応
ニューラル・ネットワークの数理 -- Tropical代数入門

max-plus の Tropical Geometry

このセッションでは、Zhangらの論文の展開に沿って、あらためてHypersurface, Newton図形等のTropical幾何の概念を紹介したいと思います。

これらについては、Part 2 でも触れています。ただ、少し違いもあります。Part 2で紹介したのは、min-plusなTropical代数上の幾何学でした。このセッションで扱うのは max-plusなTropical代数上の幾何学です。あと、ノテーションも少し異なっているところがあります。

ただ、これらの二つの幾何学の考え方は、基本的には同じものです。

max-plusでのTropical 多項式の値

まず、max-plusでのTropical 多項式の値を見ておきましょう。

Tropical 多項式 $f(x) = c_1 x^{\alpha_1} \oplus \cdots \oplus c_r x^{\alpha_r}$は、max-plus Tropical代数では次のように展開されます。

$$f(x) = c_1 x^{\alpha_1} \oplus \cdots \oplus c_r x^{\alpha_r}$$
$$= \max\{c_1 x^{\alpha_1}, \ldots, c_r x^{\alpha_r})$$
$$= \max\{c_1 + \alpha_1^T x, \cdots, c_r + \alpha_r^T x)$$

$f(x) = \max\{c_1 x^{\alpha_1}, \ldots, c_r x^{\alpha_r})$に留意してください。
これが、次の Hypersurfaceの定義の基礎になります。

Tropical Hypersurface

この論文でのhypersurfaceの定義を見てみましょう。

Tropical 多項式 $f(x) = c_1 x^{\alpha_1} \oplus \cdots \oplus c_r x^{\alpha_r}$ の
Tropical hypersurface $\mathcal{T}(f)$ は、次のように定義される。

$$\mathcal{T}(f) \coloneqq \{x \in \mathbb{R}^d : \boldsymbol{c_i x^{\alpha_i}} = \boldsymbol{c_j x^{\alpha_j}} = \boldsymbol{f(x)} \quad for\ some\ \alpha_i \neq \alpha_j \ \}$$

すなわち、多項式 f を構成する二つ（あるいはそれ以上の）単項式が、その上では同じ値を持つ点の集まりです。

$f(x) = \max\{c_1 + \alpha_1^T x, \cdots, c_r + \alpha_r^T x)$ という表示は、maxの中の項がすべて、xについての線形変換であることを表しています。

Tropical curve

Tropical単項式$c_j x^{\alpha_j}$が他の全ての単項式$c_j x^{\alpha_j}$に対して最大値をとるという条件は、$\{x \in \mathbb{R}^d : c_j + \alpha_j^T x \geq c_i + \alpha_i^T x\}$ と表現できます。

直観的には、$\mathcal{T}(f)$は、$f(x)$の単項式の表す値が「線形」である領域の境界になっています。この$\mathcal{T}(f)$の性質は、Part 2で見たhypersurface $V(f)$ と同じものです。

二変数の（すなわち、$\mathbb{R}^2$の平面上）の多項式のhypersurfaceをTropical curve と呼びます。

一般に、Tropical Hypersurfaceは、fの定義域をfが線形となる凸な領域(Cell)に分割します。これらの領域(Cell)は凸多面体であり、整数係数の線形不等式 $\{ x \in \mathbb{R}^d : Ax \leq b \}$で定義されます。

Newton Polygonとその分割

Zhangたちは、Newton 図形を次のように定義します。

Tropical 多項式 $f(x) = c_1 x^{\alpha_1} \oplus \cdots \oplus c_r x^{\alpha_r}$ の
Newton Polygon $\Delta(f)$ は、$\mathbb{R}^d$の点と見做した $\alpha_1, \cdots, \alpha_r$の
convex hull である。

$$\Delta(f) := Conv\{\alpha_i \in \mathbb{R}^d : \alpha_i \neq -\infty, i = 1, \ldots, r\}$$

これは、Part 2 で見た Newton図形の定義と同じです。

Newton図形では、単項式$c_i x^{\alpha_i}$の係数c_iは無視されています。

Newton Polygonの分割

Part 2では、きちんと説明しなかったのですが、fのhypersurfaceとfのNewton polygonの対応を考える時、Newton Polygonの「分割」が意味を持つのですが、その分割の導き方を説明しようと思います。

まず、先に無視した係数c_iを含んだ、次のような集合$\mathcal{P}(f)$を作ります。

$$\mathcal{P}(f) := Conv\{(\alpha_i, c_i) \in \mathbb{R}^d \times \mathbb{R} : i = 1, \dots, r\}$$

次に、$\mathcal{P}(f)$の上面(upper face) $UF(\mathcal{P}(f))$に、
射影 $\pi: \mathbb{R}^d \times \mathbb{R} \to \mathbb{R}^d$ を作用させて、追加したc_iの項を落とします。
それを、f によって決定されるNewton polygonのdualな部分分割 $\delta(f)$ と言います。

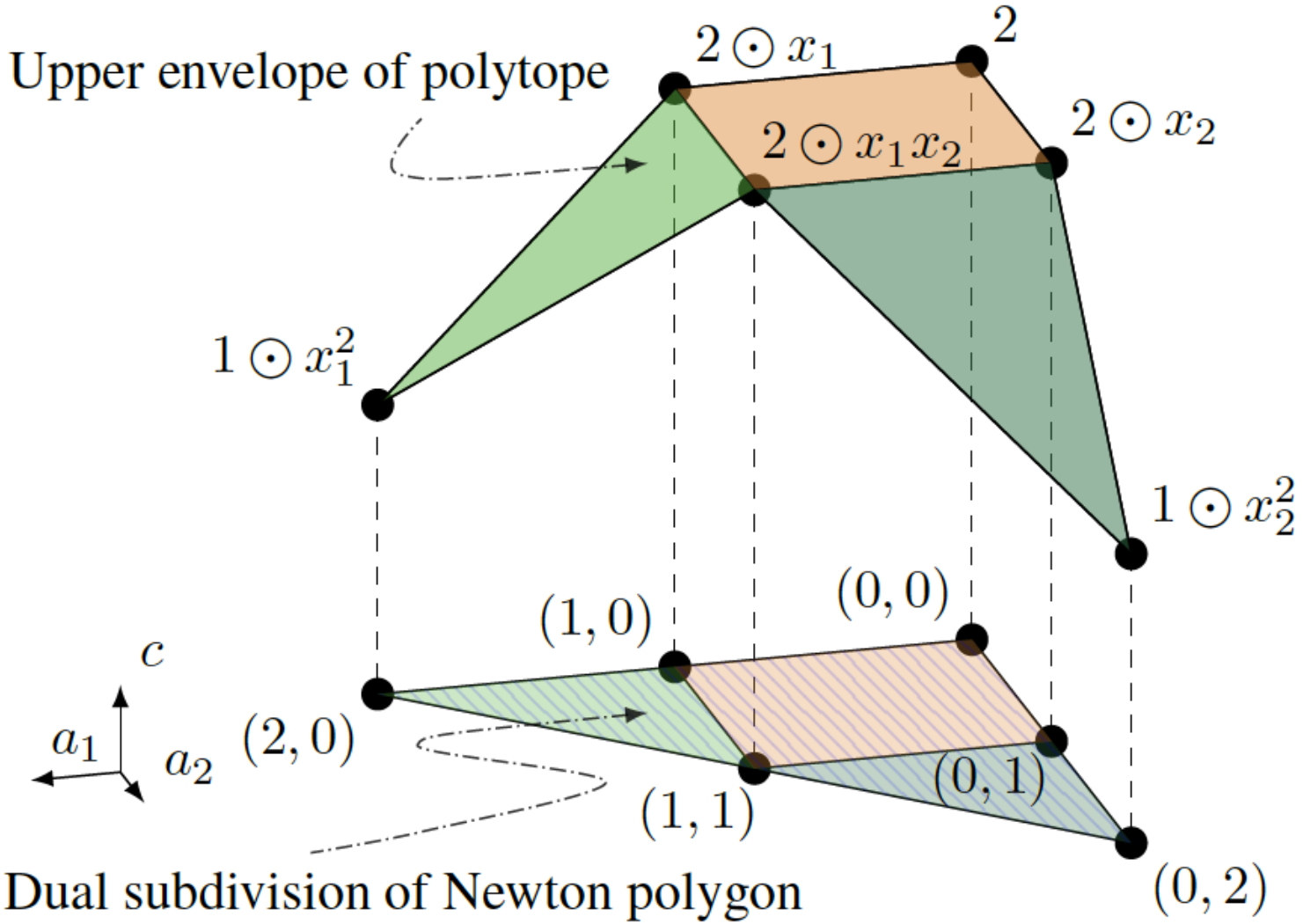
$$\delta(f) \coloneqq \{\pi(p) \subset \mathbb{R}^d : p \in UF(\mathcal{P}(f))\}$$

次の図は、Tropical多項式

$$f(x_1, x_2) = 1 \odot x_1^2 \oplus 1 \odot x_2^2 \oplus 2 \odot x_1 x_2 \oplus 2 \odot x_1 \oplus 2 \odot x_2 \oplus 2$$

のNewton polygonとdualな部分分割を示したものです。

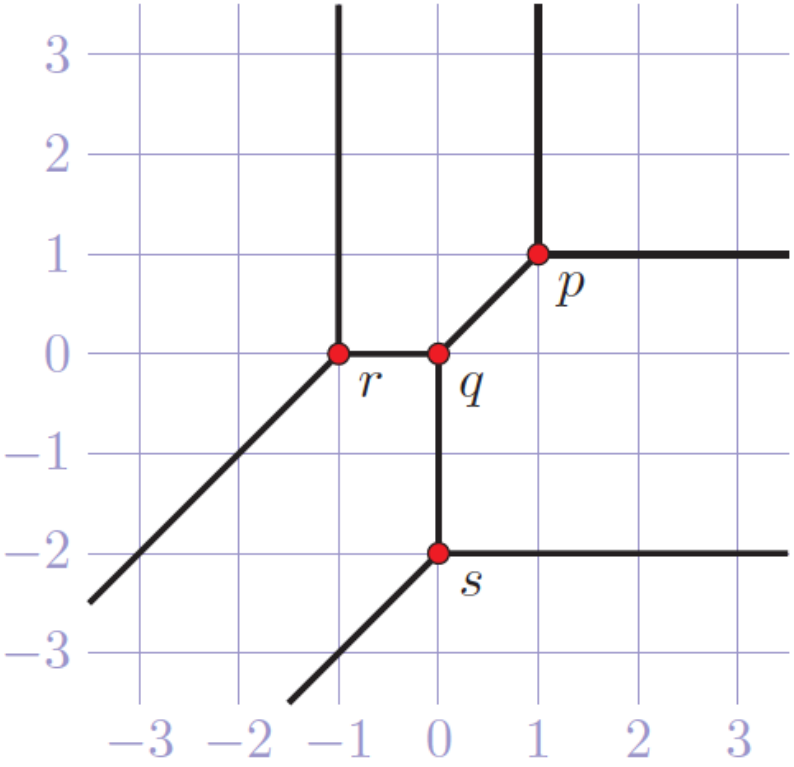
$$f(x_1, x_2) = 1 \odot x_1^2 \oplus 1 \odot x_2^2 \oplus 2 \odot x_1 x_2 \oplus 2 \odot x_1 \oplus 2 \odot x_2 \oplus 2$$



HypersurfaceとNewton Polygonの「dualな対応」の意味

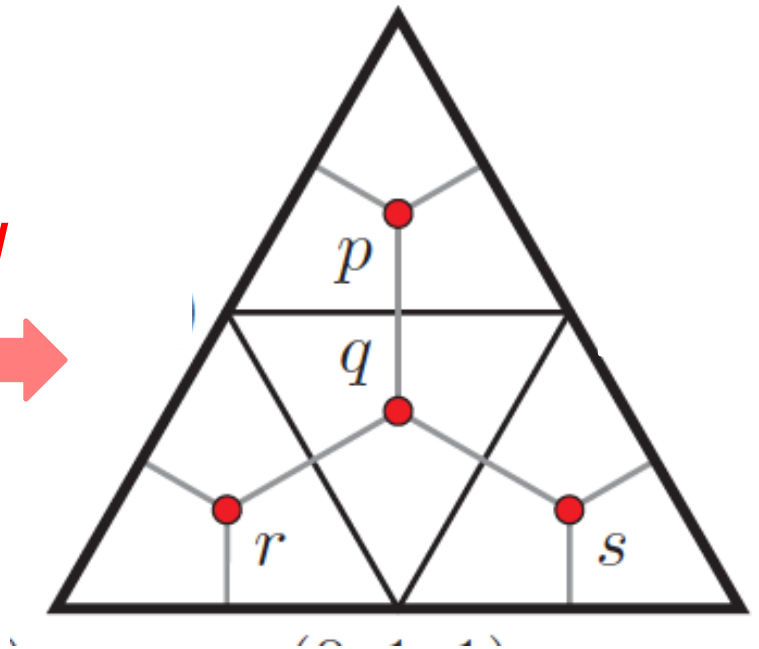
Part 2 で、次のような図を紹介したのだが、ここでは、HypersurfaceとNewton Polygonの「dualな対応」の意味 を、もう少し詳しく見ておこう。

Hypersurface V(p)と Newton図形とは、dual である



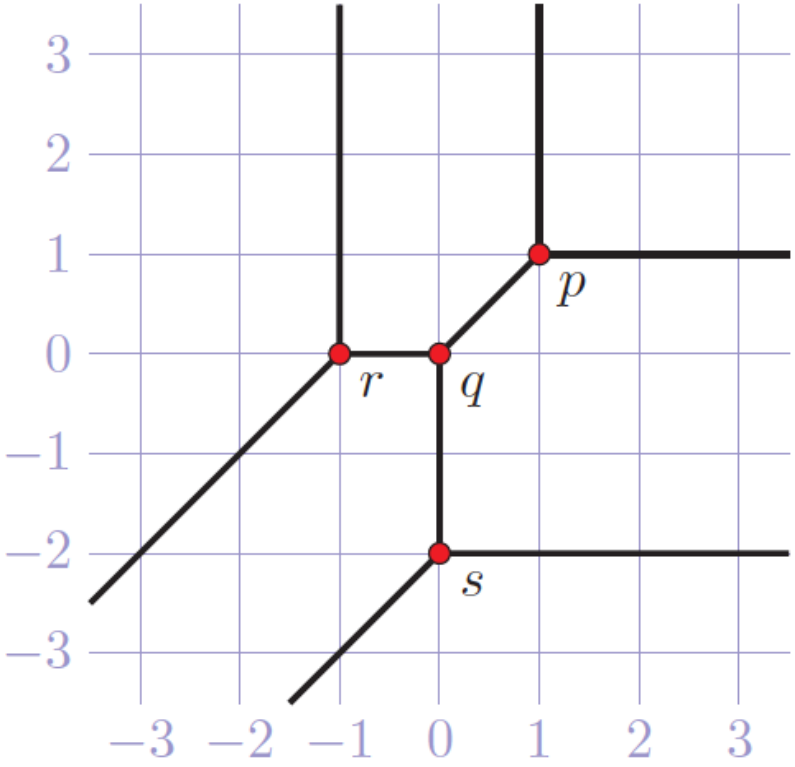
$V(p)$

dual

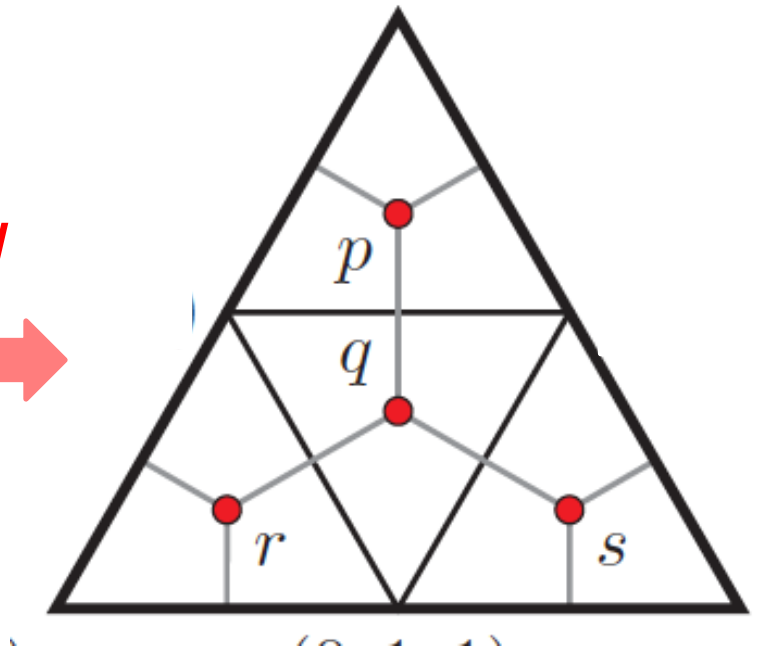


$Newton$図形

Hypersurface $\mathcal{T}(f)$と Newton polygon $\Delta(f)$ とは、dual である



dual

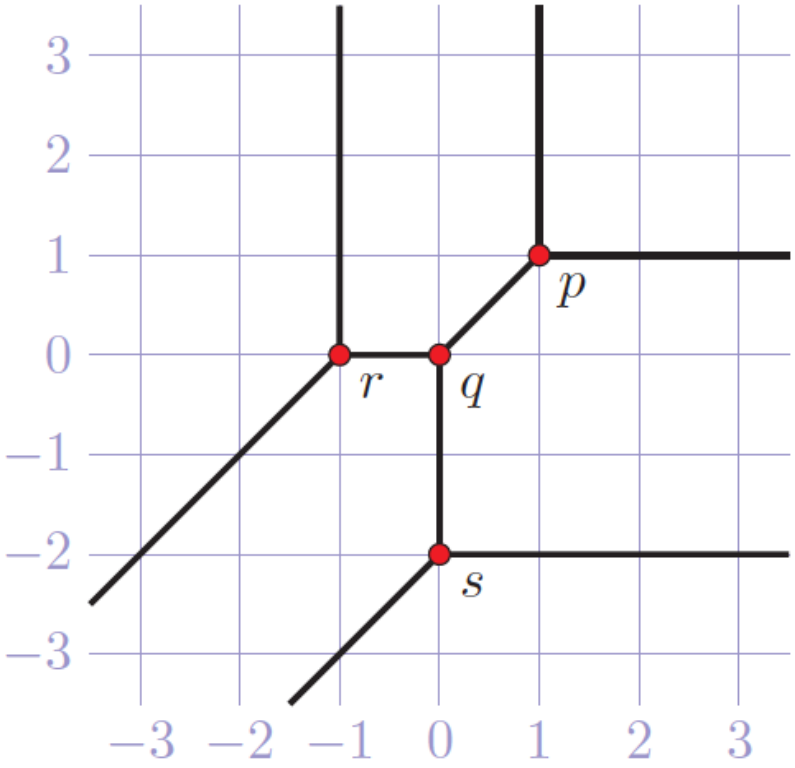


C

$Hypersurface\ \mathcal{T}(f)$

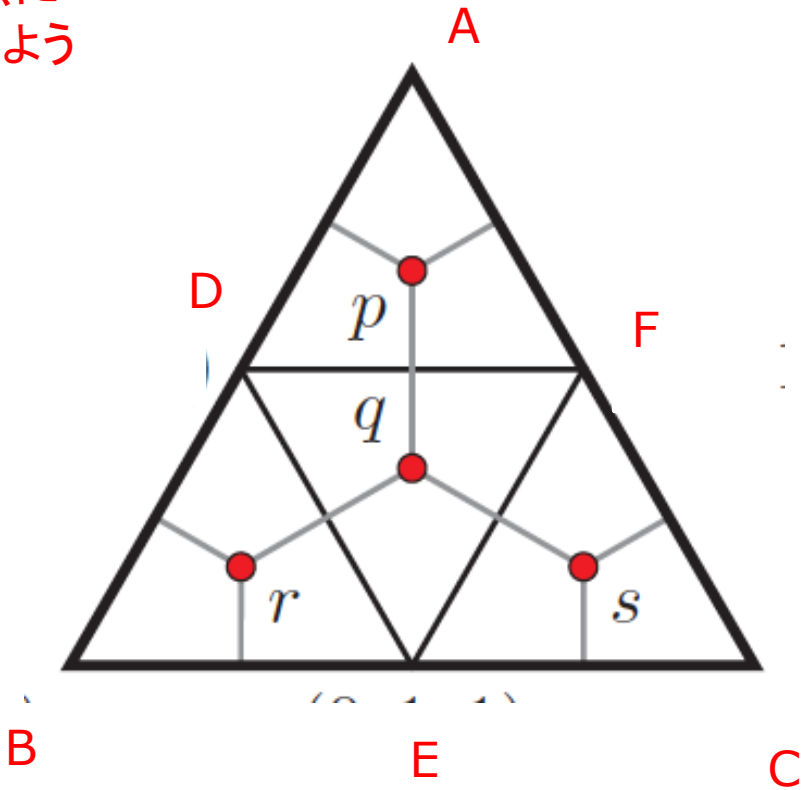
$Newton\ polygon\ \Delta(f)$

Hypersurface $\mathcal{T}(f)$と Newton polygon $\Delta(f)$ とは、dual である



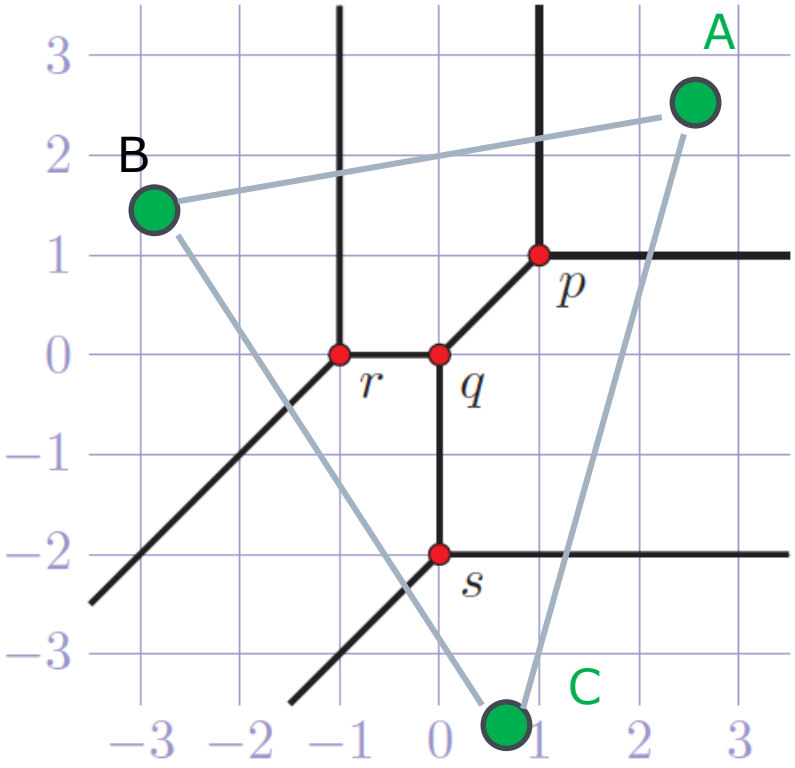
$Hypersurface\ \mathcal{T}(f)$

$\Delta(f)$上の点に
名前をつけよう



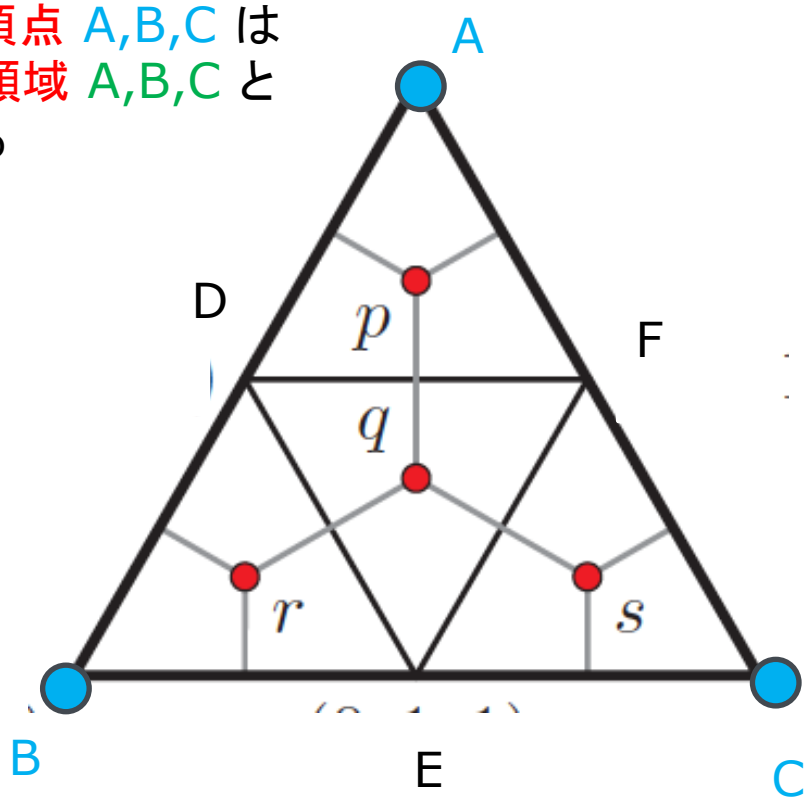
$Newton\ polygon\ \Delta(f)$

Hypersurface $\mathcal{T}(f)$と Newton polygon $\Delta(f)$ とは、dual である



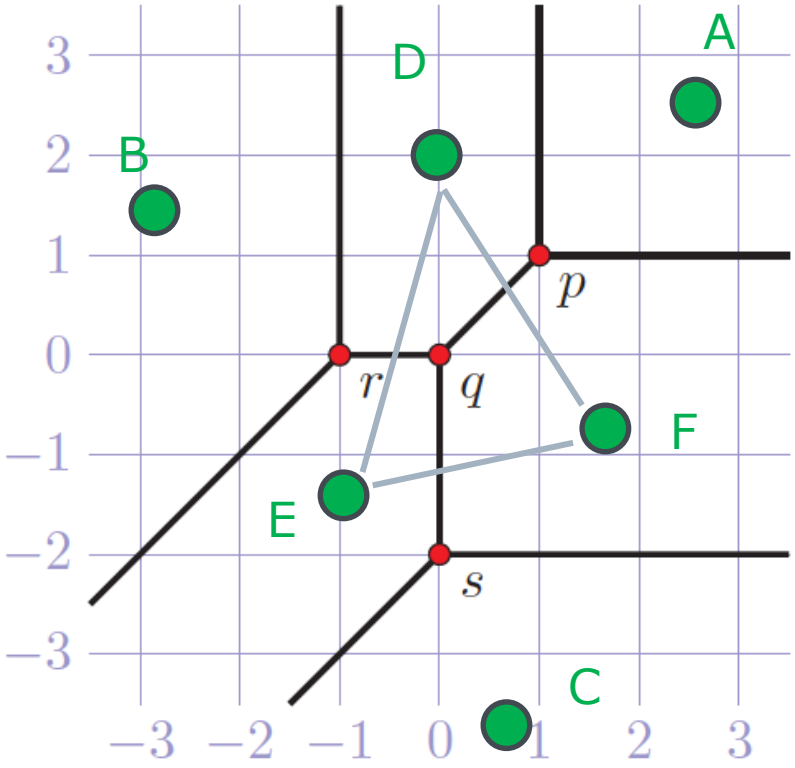
$Hypersurface\ \mathcal{T}(f)$

$\Delta(f)$の頂点 A,B,C は
$\mathcal{T}(f)$の領域 A,B,C と
対応する



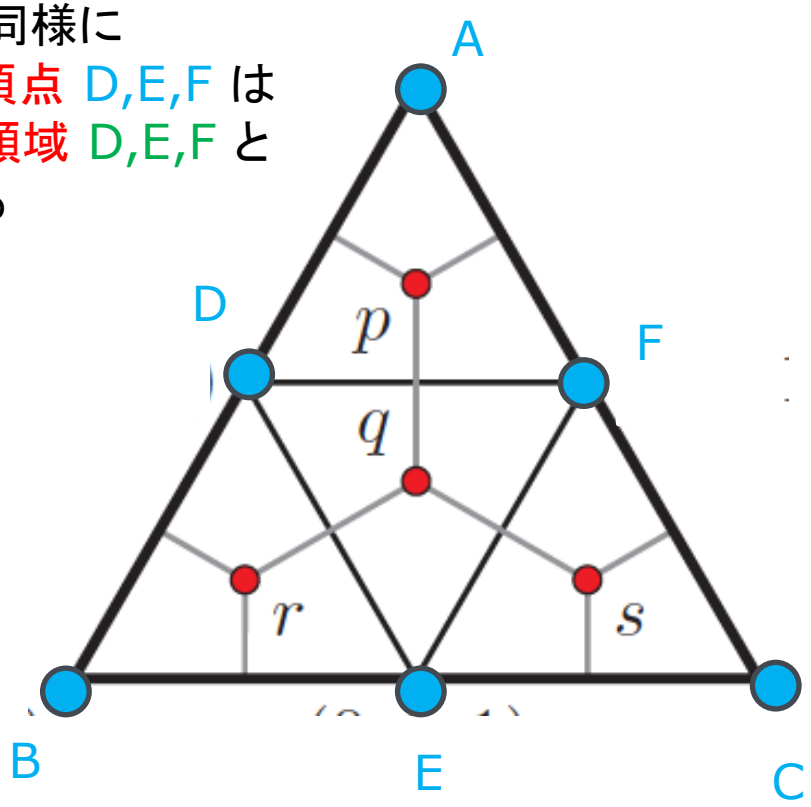
$Newton\ polygon\ \Delta(f)$

Hypersurface $\mathcal{T}(f)$と Newton polygon $\Delta(f)$ とは、dual である



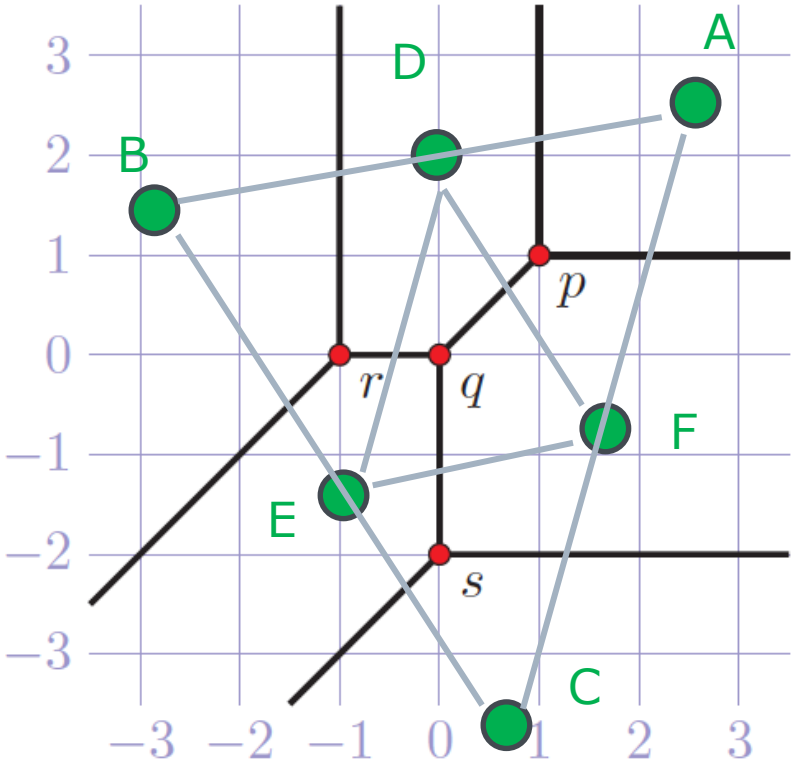
$Hypersurface\ \mathcal{T}(f)$

同様に
$\Delta(f)$の頂点 D,E,F は
$\mathcal{T}(f)$の領域 D,E,F と
対応する

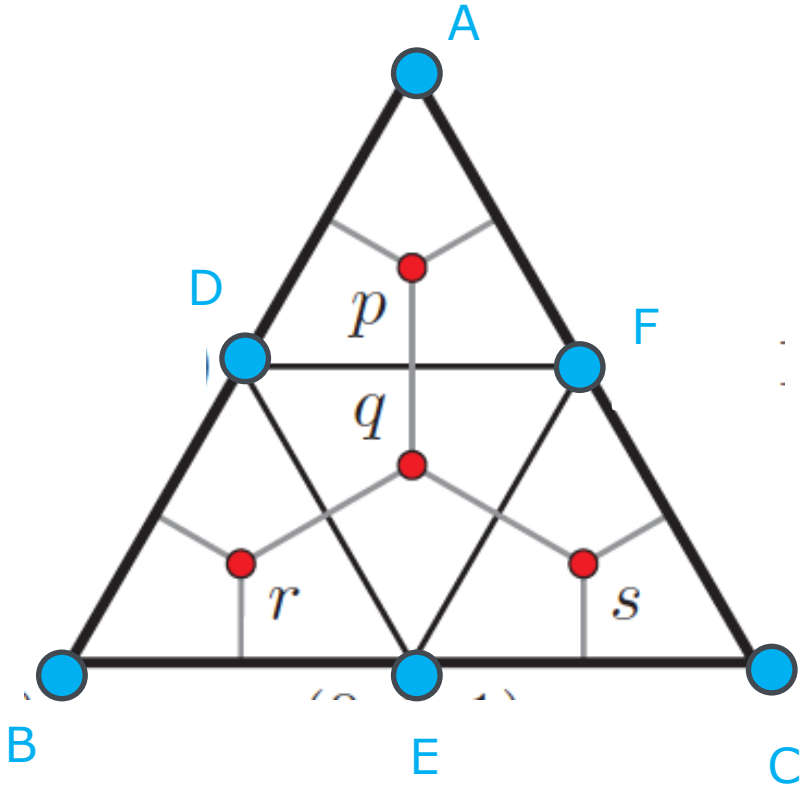


$Newton\ polygon\ \Delta(f)$

Hypersurface $\mathcal{T}(f)$ の領域と Newton polygon $\Delta(f)$ の頂点は対応する

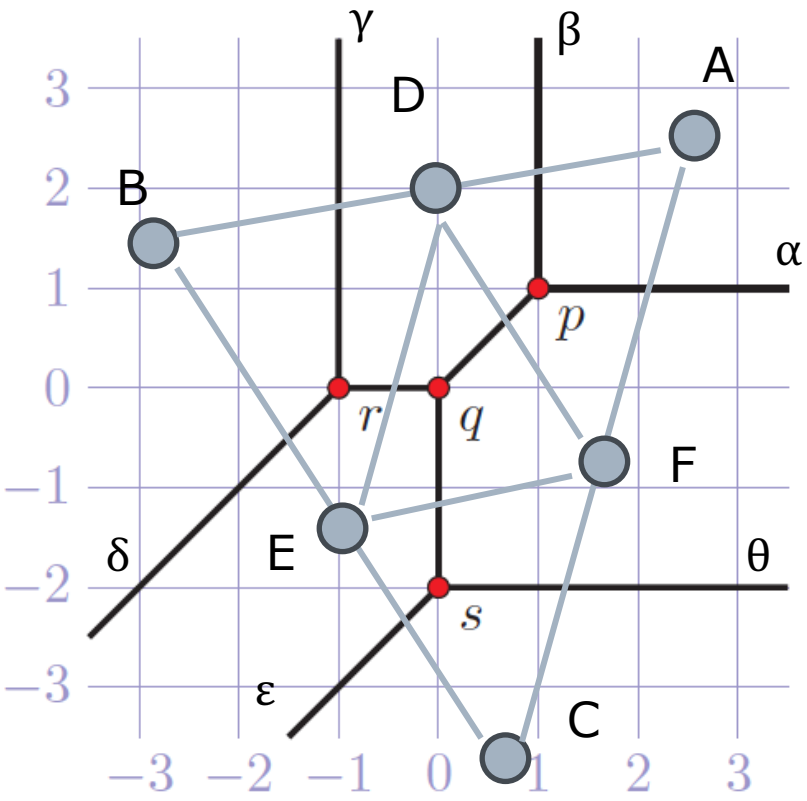


Hypersurface $\mathcal{T}(f)$

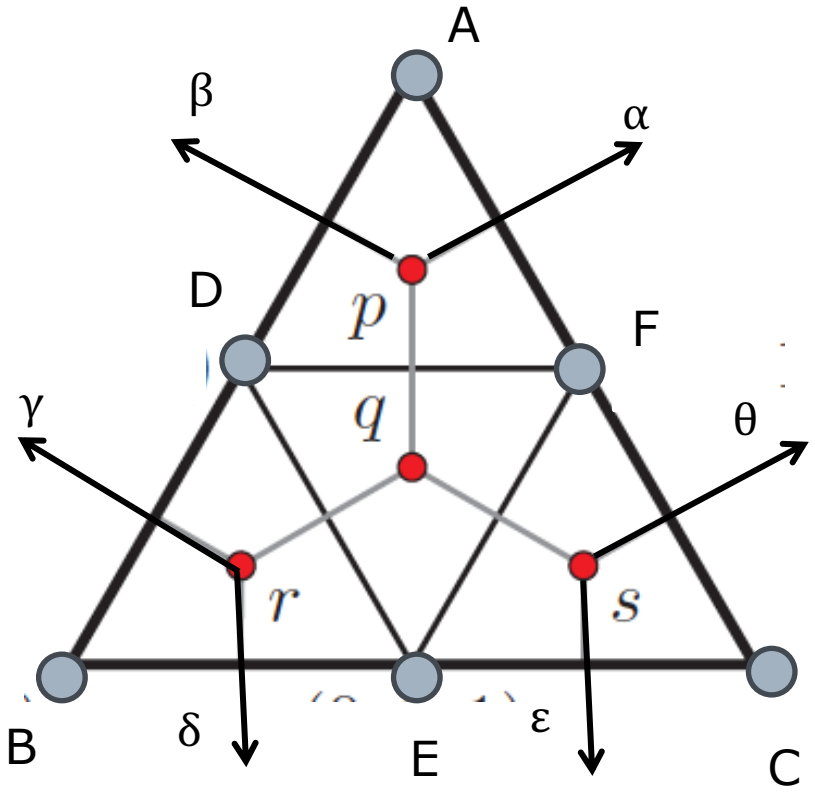


Newton polygon $\Delta(f)$

Hypersurface $\mathcal{T}(f)$と Newton polygon $\Delta(f)$ とは、dual である



Hypersurface $\mathcal{T}(f)$



Newton polygon $\Delta(f)$

Dualな対応

- Hypersurface $\mathcal{T}(f)$ の領域と
  Newton polygon $\Delta(f)$ の頂点は対応する

- Hypersurface $\mathcal{T}(f)$ の半直線 $\alpha,\beta,\gamma,\delta,\epsilon,\theta$と
  Newton polygon $\Delta(f)$ の半直線 $\alpha,\beta,\gamma,\delta,\epsilon,\theta$は対応する

- Hypersurface $\mathcal{T}(f)$ の線分 $p\text{-}q,\ r\text{-}q,\ s\text{-}q$と
  Newton polygon $\Delta(f)$ の線分 $p\text{-}q,\ r\text{-}q,\ s\text{-}q$は対応する