# ICUHS 数学ツアー2019

**H. Suzuki（鈴木寛）**

**2019年8月28日**

# 1 はじめに

# 1.1 数学ツアー案内

**2019年8月29日・30日**

タイトル：NAND回路の設計とラグランジュの補間公式
サブタイトル：〜数学的思考とその応用〜

電子回路は、ゲートといわれるものを組み合わせて作ります。ゲートのひとつが NAND と呼ばれるもので、２つの入力と、１つの出力からなり、２つの入力が両方とも ON（通常1であらわします） のときのみ、OFF（通常0であらわします）、それ以外は、ON（1）を出力します。この NAND ゲートだけをいくつか使い線でつないで、足し算を計算する回路を作ることを考えたいと思います。
LOGIC LAB: http://www.neuroproductions.be/logic-lab/ （ブラウザーによっては、設定で Flash Player を使えるようにする必要があります）

平面上に二点が与えられているとその二点を結ぶ直線が一本あります。三点が与えられていると、その三点を通る、二次関数は、いつでもありますか。それは、ひとつだけですか。点の数を、四点、五点と増やしていくとどうなるのでしょうか。考えてみましょう。

この２つの全くことなると思われる問題の背後にある、数学の考え方のいくつかを、一緒に学ぶことができればと考えています。

# 1.2 コンピュータ・ツールについて

ツールが必要な場合には、そのときに説明します。核となる部分は、数学ですから、コンピュータ・ツールは使わなくても学習できますが、使うことで視覚化でき、理解もしやすくなりますし、発想の幅も広がります。ただ、なれるのには、多少の時間がかかりますので、少し、触っておくことはたいせつです。二日間で、考えられることは時間の関係で限られますが、ツールにもなれておくことで、自分で問題を作って、考えることもでき、発展性もあります。得意な人もそうでない人も、少し、世界を広げるといった気持ちで、この機会に、使ってみてはいかがでしょうか。

大きく分けて三種類のツールを紹介します。１つ目は、論理回路に関係するもの、２つ目は、グラフを描くためのもの、３つ目は、文書を書くためのものです。３つ目は、みなさんは、基本的に使いませんが、わたしがどのようにして、数式などを書いているかを知ることは、有益だと思うのと、数学やデータ・サイエンスを利用する分野で文章を書く時には、基本的なツールですので、簡単に紹介しておきます。

すべてフリーのソフトで、かつインターネット上で、オンラインで使えるものです。アカウントを作ったほうが便利なものもありますが、１つ目と２つ目に関しては、それも必要ありません。

# 1.2.1 論理回路設計

理解を助けるツールです。たくさんありますが、二種類、紹介します。

- Logic Lab: http://www.neuroproductions.be/logic-lab/ (http://www.neuroproductions.be/logic-lab/)
- logic.ly: https://logic.ly (https://logic.ly)
    - Online Trial Page: https://logic.ly/demo (https://logic.ly/demo)

英語のサイトですが、オンラインで使えますので、コンピュータにインストールする必要もなく、かつ直感的に使えます。入力（input）は、ON または OFF （1 または 0）と考えても構いません。論理ゲート（logic gate, logic port）を通ると、入力によって変化します。出力（output）で、実際に ON なのか、OFF なのかを確認できます。基本的にこれら３つのものを線でつないで作ります。

Logic Lab と Logic.ly で多少ことなりますが、殆ど同じ機能を持っています。（Logic Lab は、Adobe の Flash Player を使っていますので、特に、iOS すなわち、Apple 系の Tablet や、Cell Phone などでは、うまく動作しないかもしれません、そのときは、Logic.ly を使ってみてください。ただし、無償版では保存はできません。）Logic Lab は、古くからある Web Application で、わたしは、長く使っています。自分で設計した回路を保存することもできます。Logic.ly のほうが新しく、発展性もあるようですが、Online 版では保存はできません。わたしは、（8月に入ってから）30日間の無償トライアル版を使っています。Windows 版と、MacOS 版があり、30日間は、すべての機能が使えるようで、保存もできます。

数学ツアーでは、NAND ゲートだけを使って、加算器を作ることを目標としていますが、他のゲート（Logic Lab では Port と呼ばれています）もぜひ使ってみてください。

Flip-Flops も使えるようになっています。今回は、時間の関係で触れることはできないと思いますが、基本的なものですので、Flip-Flop とはどのようなもので、何に使われるのか、調べてみるのもよいでしょう。

論理ゲートや、Flip-Flop や、実際のゲートが電子的にどのような構造になっているのかなどの情報は、インターネットで調べることができます。二箇所だけ上げておきます。

- TOSHIBA e-Learning: https://toshiba.semicon-storage.com/jp/design-support/e-learning/micro_intro/chap1/1274740.html (https://toshiba.semicon-storage.com/jp/design-support/e-learning/micro_intro/chap1/1274740.html)
- 頼れるエンジニアの知恵袋：https://www.chip1stop.com/sp/knowledge/top (https://www.chip1stop.com/sp/knowledge/top)

## 1.2.2 グラフ・ツール

関数のグラフを描画するツールです。二つ上げておきます。

- Desmos: https://www.desmos.com/calculator (https://www.desmos.com/calculator)
- GeoGebra: https://www.geogebra.org/graphing (https://www.geogebra.org/graphing)

Desmos は、関数のグラフの描画に特化したもので、GoeGebra は幾何にも使えるようになっています。今回は、Desmos をほんの少し使う予定ですが、殆どのことは、どちらでも可能です。機能も充実しており、学習にも、教育にも有効だと思います。

## 1.2.3 文書作成ツール

数学の文章は、数式を整えて見やすく書く必要があること、文書全体の論理構造がきっちりしていることがたいせつですが、数学者の D. Knuth が開発し、無償公開した、$TEX$（テック） という組版システムが、数学関連の論文や書籍の殆どで使われています。$TEX$ は、本体だけでなく、フォントを作成し管理するシステムや、$TEX$ プログラム自体を改善し使い易くする、マクロと呼ばれるものが最初の設計段階から整備されていたり、説明をも加えられる、web という形式で書かれたこともあり、Reproducible Research（研究の再現性）や、Literate Programming（プログラムの文書化） という、研究や、ソフトウェア開発にも欠かせない分野に大きな影響を与えました。

$TEX$ を使いやすくするために、やはり、数学者の L. Lamport によるマクロを付け加えた、$LATEX$（ラテック） の名前で呼ばれることもあります。

この文章は、web の考え方を発展させ、データ・サイエンスで主要なソフトとして使われている、R などとの連携を考えて開発された、Rmarkdown という形式で書かれています。この形式で書くと、ホームページなどを記述する、HTML（hypertext markup language）形式のほか、Microsoft Word や、PDF にも簡単に出力できるようになっています。Rmarkdown は、中にプログラムを書き込み、その実行結果とともに、文書を作成できるようになっており、プログラムもいろいろな言語に対応しています。プログラムを文書に加え、その結果を出力するだけでなく、そのプログラムの解説も、同時に作成することができ、データを書き換えても、グラフなども同時に書き換えることができるので、データ・サイエンスにおける、再現性（Reproducibility）にはなくてはならないものになっています。今回は、データ分析も、プログラミングも行いませんが、いろいろな形式の文書として、書き出すことができる利点を考えて、Rmarkdown で書き始めています。Rmarkdown は、テキストで文書を作成しますが、RStudio と連携させると、特に、データ分析のときには、便利です。他に似たものとしては、Python というコンピュータ言語に付随して開発された、Jupyter と呼ばれる、Python Notebook が有名です。

- RStudio: https://www.rstudio.com (https://www.rstudio.com)
- RMarkdown: https://rmarkdown.rstudio.com (https://rmarkdown.rstudio.com)
- $TEX$: https://texwiki.texjp.org (https://texwiki.texjp.org)
- Online $LATEX$: https://cloudlatex.io/ja (https://cloudlatex.io/ja)
- Cocalc: https://cocalc.com (https://cocalc.com)
- SageCell: https://sagecell.sagemath.org (https://sagecell.sagemath.org)

Cocalc は、$LATEX$ も Rmarkdown も Jupiter も 使えますし、数式の計算もできるもので、最初は、SageMath の一部として開発されたものです。Cocalc は、Collaborative Calculation からとったもので、共同作業に適した環境を提供しています。最後の SageCell は、電卓のようにして使える、SageMath（商用の Mathematica や Maple などの代わりになるものとして開発されました） です。

Rmarkdown の中でも、数式は、$TEX$ 形式で書いています。

# 1.3 問題について

## 1.3.1 論理回路の問題

The Logic Lab で説明してみます。ページの下に、Logic Ports と書いてあるものがあります。それは、この [リンク] (http://www.neuroproductions.be/logic-lab/index.php?id=104983) のようになっています。すべてにスイッチがついていて、白になっていれば、OFF、赤になっていれば ON です。それを動かして、AND, NOR, NOT, NAND, XOR, OR, XNOR それぞれで、入力がどうな

っているときに、電気が赤くつくかを調べてみてください。案内に書いたように、NAND は、両方が、OFF のときのみ ON になっていることが確かめられると思います。他のものは、それぞれどうなっていますか。

Logic.ly の場合には、ある回路を選択すると、Truth Table（真理表）を作ってくれる機能もあります。上で書いた、ON または 1 が TRUE で、OFF または 0 が FALSE に対応しています。入力の値によって、出力がどうなるかが読み取れるようになっています。

加算器を作りたいのですが、２進演算を考えます。簡単なものから始めます。

- $0+0=0, 0+1=1, 1+0=1, 1+1=10_2$ です。最後だけ二桁になっています。添字の2は、2進であることを示したものです。繰り上がりを Carry と言います。ですから、みな二桁だと考えて、
- $0+0=00_2, 0+1=01_2, 1+0=01_2, 1+1=10_2$ と出力を二桁にしてみましょう。
- これを、いくつかのゲート（ポート）を組み合わせて作ってみてください。
- Output に2進４桁のカウンターもありますから、数字で答えを出すことも可能です。Input も数字で出せるとわかりやすいですね。
- 時間があれば、最終的には、２進三桁＋２進三桁を計算できる加算器を作りたいと思います。２進三桁は、10進では $0, 1, 2, 3, 4, 5, 6, 7$ ですね。
- ちょっと難しいのは、これを、NAND ゲート（Port）だけで作ってみようという部分ですが、まずは、他のゲートも使って、作ってみてください。
- NAND だけでできたら、NAND ゲート を使う数を一番少なくするにはどうしてらよいか考えてみてください。

どのように考えていったらよいのでしょうか。ひとに説明できるように、考え方も整理できるとよいですね。

## 1.3.2 いくつかの点を通る多項式関数

- 平面上の二点 $(a,b)$ と $(c,d)$ を通る直線は、いつでも一本ありますね。その方程式はどうなりますか。
- 三点 $(a,b)$ と $(c,d)$ と $(e,f)$ のときは、どうでしょうか。二次関数をつかうと、この三点を通るものがいつでも見つけられますか。

  $$y=\alpha x^2+\beta x+\gamma$$

  として、$\alpha, \beta, \gamma$ はいつでも見つかりますか。
- 二次関数が見つかる条件は何でしょうか。それは、ただ一つに決まるのでしょうか。
- 二点を通る直線は、1次関数に関係しており、三点を通る直線は、2次関数に関係しているとすると、点の数を、4点、5点、6点とどんどん増やしていくとどうなると思いますか。
- すっきりした形で記述する方法はあるでしょうか。
- $y=\alpha x^3+\beta x^2+\gamma x+\delta$ は、$\alpha \neq 0$ のとき、三次関数と呼びます。三点 $(a,b)$ と $(c,d)$ と $(e,f)$ を通る、三次関数をすべて求めることはできるでしょうか。
- どのように結果を整理（定理として記述）したら良いでしょうか。
- $y=\alpha x^3+\beta x^2+\gamma x+\delta$ は、$\alpha \neq 0$ の左辺は、右辺の $x$ の値によって決まりますから、$x$ の関数とよび、$f(x)=\alpha x^3+\beta x^2+\gamma x+\delta$ と書きます。$f(x)$ は三次関数です。$c$ を実数としたとき、

  $$f(x)=\alpha x^3+\beta x^2+\gamma x+\delta=(\alpha' x^2+\beta' x+\gamma')(x-c)+r$$

  と書けることが知られてます（数学II）。$\alpha'=\alpha$ と書けることはわかりますね。上のように書けている時、$f(x)$ を $x-c$ で割って、商が $\alpha' x^2+\beta' x+\gamma'$ 余りが、$r$ であるともいいます。
- 上の式で、$r=f(c)$ となっていることに注意してください。特に、$f(c)=0$ のときは、

  $$f(x)=(\alpha' x^2+\beta' x+\gamma')(x-c)$$

  と書くことができます。
- Desmos で遊んでみましょう。
  - 多項式関数 (https://www.desmos.com/calculator/kgbvajslwl) 係数の部分をスライドさせると、グラフの形が変わります。最初の表の値を変えて、それを通る関数を見つけることができますか。
  - 左上に三本線がありますが、それをクリックすると、例が出てきます。少しずつ変更してどんなことができるか、やってみましょう。

# 1.4 数学を復習しておきたい人のために

## 1.4.1 2進数の演算

1. 10進数の 0-15 を2進数で表すと
2. 10進数を2進数で表す方法
3. 2進数を10進数で表す方法
4. 2進数の足し算 例：$101_2 + 110_2$
5. 2進数の掛け算 例：$101_2 \times 110_2$
6. 2進数で1より小さい小数を表す方法

## 1.4.2 直線と二次関数

1. $(0, -2)$ を通り、傾きが $3$ の直線の方程式。
2. $(0, b)$ を通り、傾きが $a$ の直線の方程式。
3. 平面上の直線で、$y = ax + b$ とは表せないもの。
4. $ax + by + c = 0$ をみたす平面上の点 $(x, y)$ 全体が直線となる条件。
5. $(a, 0)$ と $(0, b)$ を通る直線の方程式。
6. $(a, b)$ と $(c, d)$ を通る直線の方程式。
7. $(0, 0), (1, -1), (3, 3)$ を通る二次関数。
8. $(1, a), (2, b), (3, c)$ と通る二次関数（か、１次関数か…)。

# 2 NAND ゲートの加算器

最初に、1.3.1 にあることから確認していきたいと思います。

## 2.1 クラスで考えてみたいこと

1. $p, q, r$ がすべて $0$ か $1$ とすると、三組 $(p, q, r)$ は

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

   全部で8通りあります。入力が、８種類の $(p, q, r)$ のときに、出力が、それぞれ、 たとえば、

   $$(0,1,1,0,1,0,0,1), \quad (0,0,0,1,0,1,1,1)$$

   となるような回路は作れますか。（$(0,0,0)$ のとき $0$, $(0,0,1)$ のとき $1$, $(0,1,0)$ のとき、$1$ などです。）加算器の問題とは、どのように関係しているのでしょうか。
2. 1で出力の8個組がなんであっても、それを出力する回路は作れるでしょうか。
3. 2の回路は、NAND だけでつくれるでしょうか。
4. 基本的なものをまず作って、それを組み合わせてできないだろうか。

## 2.2 授業 I: 論理回路設計（１）

### 2.2.1 2進数の演算

答えだけ書いておきます。

1. 10進数の 0-15 を2進数で表すと
   - $0000_2, 0001_2, 0010_2, 0011_2, 0100_2, 0101_2, 0110_2, 0111_2, 1000_2, 1001_2, 1010_2, 1011_2, 1100_2, 1101_2, 1110_2, 1111_2$. これらを16進記法で、$0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F$ と書く場合もあります。
2. 10進数を2進数で表す方法
   - 10進数を $n$ としたとき、2 で割ってあまりを計算していく方法と、$2^m \leqq n < 2^{m+1}$ を満たす、$m$ を見つけて順に引いていく方法があります。
   - $n = a_m 2^m + a_{m-1} 2^{m-1} + \cdots + a_1 2^1 + a_0 2^0$ （$a_m, a_{m-1}, \ldots, a_1, a_0$ は $0$ または $1$）ですから、上の方法は、$a_0, a_1, \ldots$ の順番に見つけていく方法と、$a_m, a_{m-1}, \ldots$ と見つけていく方法と考えることもできます。
   - $n = a_m a_{m-1} \ldots a_1 a_{0\,2}$ または、$n = (a_m a_{m-1} \ldots a_1 a_0)_2$ などと書きます。
3. 2進数を10進数で表す方法
   - 上の2の説明を逆にたどれば、よいですね。
4. 2進数の足し算 例：$101_2 + 110_2$
   - 10進の5と6ですから、和は11、すなわち、$1011_2$ となります。
   - 10進の筆算のように繰り上がりにも注意すれば、以下のようになります。

$$
\begin{array}{cccc}
 & & 1 & 0 & 1 \\
+ & & 1 & 1 & 0 \\
\hline
 & 1 & 0 & 1 & 1
\end{array}
$$

5. 2進数の掛け算 例：$101_2 \times 110_2$

$$
\begin{array}{cccccc}
 & & & 1 & 0 & 1 \\
\times & & & 1 & 1 & 0 \\
\hline
 & & 1 & 0 & 1 & \\
 & 1 & 0 & 1 & & \\
\hline
 & 1 & 1 & 1 & 1 & 0
\end{array}
$$

6. 2進数で1より小さい小数を表す方法
    - 基本的には、10進数の場合と同じです。$0.5 = 2^{-1} = .1_2$ などとなります。$n$ を1より小さい小数とすると、$n = b_1 2^{-1} + b_2 2^{-2} + b_3 2^{-3} + \cdots$（$b_1, b_2, \ldots$ は、$0$ または $1$） となります。ただし、10進数では、小数点以下有限であっても、無限小数になる場合もあります。今回は、使いませんが、考えてみるとよいと思います。
    - $0.1, 0.2, 0.3$ などはどうなりますか。$0.625$ などはどうなりますか。

## 2.2.2 論理回路

２進演算で、加算器を作ることを考えましょう。単純なものから考えます。すなわち、二進一桁を一ビット（bit）と呼びますが、１ビット＋１ビットの加算器です。$A$と$B$であらわしましょう。

- $0+0=0, 0+1=1, 1+0=1, 1+1=10_2$ です。最後だけ二桁になっています。添字の2は、2進であることを示したものです。繰り上がりを Carry と言います。ですから、みな二桁だと考えて、
- $0+0=00_2, 0+1=01_2, 1+0=01_2, 1+1=10_2$ と出力を二桁にしてみましょう。
- 一桁目（$2^0$ の位）を $A \oplus B$ 繰り上がりを $C$ であらわしましょう。すると次のようになっていることがわかります。

| $A$ | $B$ | $A \oplus B$ | $C$ |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 |

- 入力が $A$ と $B$ の二つ。出力が、$A \oplus B$ と $C$ の二つということになります。出力は一つ一つ作っていくことにしましょう。
- これを、いくつかのゲート（ポート）を組み合わせて作りたいので、それぞれの、ゲートが入力に対して、どのような出力をしているか見てみましょう。

| $A$ | $B$ | $A \oplus B$ | $C$ | $A$ NAND $B$ | $A$ AND $B$ | $A$ OR $B$ | NOT $A$ | $A$ XOR $B$ | $A$ NOR $B$ |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 0 |
| 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 |

- どんなことがわかりますか。
    - $A \oplus B \equiv A$ XOR $B$
    - $C \equiv A$ AND $B$
- NAND だけを使うということでなければ、これですでに加算器はできているようです。確認してみてください。[“Simple Half Adder” へのリンク] (http://www.neuroproductions.be/logic-lab/index.php?id=104988)
- もう一つ確認しておきたいことがあります。

| $A$ | $B$ | NOT | ($A$ AND $B$) | NOT($A$) | OR | NOT($B$) | $A$ NAND $B$ |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 1 | 0 | 1 | 1 | 1 | 1 |
| 0 | 1 | 1 | 0 | 1 | 1 | 0 | 1 |
| 1 | 0 | 1 | 0 | 0 | 1 | 1 | 1 |
| 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 |

- 入力が何であっても、出力が同じであるとき、同値といい、≡の記号を使って、次のように書きます。
  - $\mathrm{NOT}(A \mathrm{AND} B) \equiv A \mathrm{\ NAND\ } B$
  - $\mathrm{NOT}(A) \mathrm{\ AND\ } \mathrm{NOT}(B) \equiv A \mathrm{\ NAND\ } B$
  - $\mathrm{NOT}(A \mathrm{AND} B) \equiv \mathrm{NOT}(A) \mathrm{\ AND\ } \mathrm{NOT}(B)$
  - 公式のようなもので、最後のものは、ド・モルガンの法則とも言われ、高校で習った人もいると思います。
- 複雑になると、簡単な記号を使ったほうがわかりやすいこともありますので、一般的に使われる（論理）記号を紹介しておきます。
  - $\mathrm{NOT}(A)$: $\bar{A}, \neg A, \sim A$ 日本の高校では 1 つ目を使っているので、ここでもそれを使うことにします。
  - $A \mathrm{\ AND\ } B$: $A \wedge B$
  - $A \mathrm{\ OR\ } B$: $A \vee B$
  - $A \mathrm{\ XOR\ } B$: $A \oplus B, A \veebar B$
  - $A \mathrm{\ NAND\ } B$: 一般的な記号はありません。$A \uparrow B$
  - $A \mathrm{\ NOR\ } B$: 一般的な記号はありません。$A \downarrow B$
- 記号をつかうと、次のように書くことができます。
  - $A \uparrow B \equiv \overline{A \wedge B} \equiv \bar{A} \vee \bar{B}.$
- 二桁以上の場合も一桁ずつ計算しますが、桁上り（Carry）の部分も計算しないといけませんから、入力が $A, B, C$ で出力が $X, Y$ のようになっています。$Y$ を桁上り分としましょう。
- $A = B = C = 1$ であっても、桁上りをふくめて二桁で収まることを確認しておきましょう。するとどうなるでしょうか。

| $A$ | $B$ | $C$ | $X$ | $Y$ | $Z$ |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | |
| 0 | 0 | 1 | 1 | 0 | |
| 0 | 1 | 0 | 1 | 0 | |
| 0 | 1 | 1 | 0 | 1 | |
| 1 | 0 | 0 | 1 | 0 | |
| 1 | 0 | 1 | 0 | 1 | |
| 1 | 1 | 0 | 0 | 1 | |
| 1 | 1 | 1 | 1 | 1 | |

- これを実現する回路を全加算器（Full Adder）といいます。さきほどの、入力に、桁上りを考えないものを、半加算器（Half Adder）といいます。
- いろいろなゲートを用いて、全加算器を作り、あとから、上のような公式を使って、NAND 回路にしていく方法も一つです。そのことは、次の時間で説明します。
- 上の公式では、NAND を他のもので書き換えてみましたが、逆に、他のゲートを、NAND で置き換えることはできないでしょうか。下の右辺を NAND だけでかけますか。NOT, AND, OR, XOR, NOR は使わないということです。
  - $\bar{A} \equiv$
  - $A \wedge B \equiv$
  - $A \vee B \equiv$
  - $A \oplus B \equiv$
- 問題をまとめておきます。
  1. 上の表の $X$ や $Y$ または、$Z$ に $0, 1$ がどのように並んでいても、いろいろな論理記号を使って、値がちょうどそのようになるものを作れるか。
  2. 他の記号をつかわず、NAND だけで書くことができるか。
  3. 二桁以上になったときに、どのように組み合わせていけばよいか。

# 2.3 授業 II: 論理回路設計（２）

最後に述べた問題の 1 と 2 について考えてみたいと思います。

具体的には、次の二つの項目について話します。

1. 表の $Z$ に 0, 1 がどのように並んでいても、入力の $A, B, C$ と NOT, AND, OR だけを組み合わせて、$Z$ が出力になるようにすることができる。(Conjunctive Normal Form, CNF)
2. NOT, AND, OR いずれも、NAND で書くことができる。(Functional Completeness, FC)

注：XOR については、述べていませんが、$A \oplus B$ の値を、$Z$ に書いておけば、そのように、NOT, AND, OR だけで書けるのですから、心配いらないことがわかります。

# 2.3.1 NAND の完全性

- $\bar{A} \equiv A \uparrow A$
- $A \wedge B \equiv \overline{A \uparrow B}$：NOT を使っていますが、上を使えば、NOT を使わない形に変形できます。
- $A \vee B \equiv \bar{A} \uparrow \bar{B}$：上と同様

Hint: $Z$ がどんな式であっても、$\bar{\bar{Z}} = Z, A \uparrow B \equiv \overline{A \wedge B} \equiv \bar{A} \vee \bar{B}$

# 2.3.2 Conjunctive Normal Form, CNF

| $X$ | $Y$ | $Z$ | $X \wedge Y \wedge Z = E_8$ | $X \vee Y \vee Z$ | $S$ | $C$ | $E_2$ | $E_3$ | $E_5$ |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 0 |
| 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 |
| 1 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |

- $S \equiv E_2 \vee E_3 \vee E_5 \vee E_8$
- $E_8 \equiv X \wedge Y \wedge Z$
- $E_2 \equiv \bar{X} \wedge \bar{Y} \wedge Z$
- $E_3 \equiv$
- $E_5 \equiv$

注：日本語では、連言標準形というらしいです。他方、Disjuntive Noraml Form (DNF) は、選言標準形。

# 2.3.3 回路の設計と課題

- 原理的には、全加算器の回路ができることを示しました。
- かなり複雑ですから、全加算器をNAND で簡単に書く方法を考えてください。
- できれば、３桁＋３桁の計算までできる加算器を作ってください。
- 設計した加算器と工夫したこと、考え方などを、説明してもらおうと思います。
- NAND ではなく、他の一種類のゲートだけで、書くことはできないのでしょうか。考えてみてください。

# 2.3.4 練習問題

1. $A \Rightarrow B$ （$A$ IM $B$ ともかく）ゲートを下の表で定義する。空欄を埋めよ。

| $A$ | $B$ | $A \Rightarrow B$ | $\bar{A} \vee B$ | $\overline{A \wedge \bar{B}}$ | $\bar{B} \Rightarrow \bar{A}$ |
|---|---|---|---|---|---|
| 0 | 0 | 1 | | | |
| 0 | 1 | 1 | | | |
| 1 | 0 | 0 | | | |
| 1 | 1 | 1 | | | |

   このことから、次がわかる。

$$A \Rightarrow B \equiv \bar{A} \vee B \equiv \overline{A \wedge \bar{B}} \equiv \bar{B} \Rightarrow \bar{A}.$$

   $\bar{B} \Rightarrow \bar{A}$ は対偶（contrapositive）と呼ばれる。

2. 次の二つの式で表される論理回路の出力が同じではない（同値でない）ことを示せ。

$$(A \wedge B) \vee C, \quad A \wedge (B \vee C)$$

   （回路を作って示しても、表を書いて示しても、他の方法でも構いません。）

3. $A \Rightarrow (B \vee C) \equiv A \wedge (\bar{B}) \Rightarrow C$ すなわち、どちらの論理回路も $(A, B, C)$ のすべての値について同じ出力となること（同値であること）を、二つの方法で示せ。

    a. 表を完成することで確認せよ。
    b. 実際に、論理回路を作って確認せよ。
    c. 練習問題 1 などの公式を使った変形により示せ。
4. 論理記号 ↑ NAND だけを用い（NAND 回路で）次を表わせ。NOT も使ってはいけません。

    a. $A \wedge B$
    b. $A \vee B$
    c. $A \oplus B$
    d. $A \Rightarrow B$
    e. $A \downarrow B$
5. $(A \wedge \bar{B}) \Rightarrow C$ を以下の指示にしたがって変形せよ。

    a. $\vee$ と NOT（否定）だけを使い、$\wedge$、$\Rightarrow$ などを用いない。
    b. ↑ だけを使い、$\wedge, \vee, \Rightarrow$ NOT（否定）などを用いない。
6. $(A, B, C)$ の入力が $(0, 0, 0), (0, 1, 0), (1, 0, 1)$ のときだけ $1$ でそれ以外、$0$ を出力する回路を作成せよ。

    a. AND, OR, NOT ゲートだけを用いよ。
    b. NAND ゲートだけを用いよ。

# 3 いつくかの点を通る多項式関数

最初に 1.3.2 に書いてあることから確認していきたいと思います。

# 3.1 クラスで考えてみたいこと

1. たくさんの点を考える時、どのような記号がよいだろうか。
2. 基本的なものをまず作って、それを組み合わせることはできないだろうか。
3. すべての点で、$0$ という値をとるものは、どのようになっているだろうか。
4. 一つの点では、1で他では、$0$ という値をとるものは、どのようなものが考えられるだろうか。
5. ただ一つしかないということは、どのようにしたら示すことがでるでしょうか。

# 3.2 R and SageMath Code

下のプログラムと結果は、何をあらわしているのでしょうか。

```
# R Program
library(polynom)
x<-c(0,2,3,4)
y<-c(7,11,28,63)
poly.calc(x,y)
```

```
## 7 - 2*x + x^3
```

“SageMathCell” へのリンク (https://sagecell.sagemath.org)

次の二行を copy して、SageMathCell の窓に paste して、Evaluate くだみてさい。

```
R = PolynomialRing(QQ, 'x')
R.lagrange_polynomial([(0,7),(2,11),(3,28),(4,63)])
```

$(0, 7), (2, 11), (3, 28), (4, 63)$ の部分を変更するとどうなるでしょうか。

## 3.2.1 いくつかの点を通る（多項式）関数について

まず、$c_0, c_1, \ldots, c_n$ を数とし、$x$ を変数としたとき、次の式を考える。

$$y = c_n x^n + c_{n-1} x^{n-1} + \cdots + c_1 x + c_0$$

右辺の$x$にある数を代入すると、$y$ の値が決まるので、$x$ の値によって、$y$ の値が一つ定まることを　$y$ は $x$ の関数（function）であるといい、$y = f(x)$ などと書く。$f$ は function からとっている。右辺は、$x$ の整式ともよばれるが、ここでは、多項式（polynomial）と

呼ぶことにする。高校では、一項しかないとき、単項式、複数項のとき、多項式と区別することもあるが、ここでは、すべて多項式と呼ぶことにする。ここで、$c_n \neq 0$ であるとき、$f(x)$ の次数（degree）は $n$ であるといい、$\deg f(x) = n$ と書く。右辺が $0$ のとき、すなわち、ゼロ多項式の次数の次数もいくつか決め方があるが、ここでは、$0$ でない多項式のときだけ、次数を考えることにする。定数たとえば、$f(x) = c_0 = 2$ の次数は、$0$ である。右辺が次数$n$ の多項式の時、$y = f(x)$ を次数$n$（または $n$次）の多項式関数という。

1. $y = 2x - 1$ は1次の多項式関数。
2. $y = x^2 - 2x + 1$ は $2$ 次の多項式関数である。
3. 一般の2次の多項式関数は、$y = c_2x^2 + c_1x + c_0$ で、$c \neq 0$ と書けている。
4. $y = f(x) = c_2x^2 + c_1x + c_0$ で $b = f(a)$ となっているとき、関数 $y = f(x)$ は $(x, y)$ 平面の点 $(a, b)$ を通るという。
5. 点 $(0, 0), (1, 1), (2, 4)$ を通る二次関数はあるでしょうか。直観的に、$y = x^2$ と答えられる人が多いと思います。確かに、$f(x) = x^2$ は、この3点を通ります。他に、この3つの点を通る二次関数はありませんか。
6. $y = f(x) = c_2x^2 + c_1x + c_0$ と置くと、

$$
\begin{aligned}
0 = f(0) &= 0 \cdot c_2 + 0 \cdot c_1 + c_0 = c_0 \\
1 = f(1) &= 1 \cdot c_2 + 1 \cdot c_1 + c_0 = c_2 + c_1 + c_0 = c_2 + c_1 \\
4 = f(2) &= 4 \cdot c_2 + 2 \cdot c_1 + c_0 = 4 \cdot c_2 + 2 \cdot c_1 = 2 \cdot c_2 + 2(c_2 + c_1) = 2 \cdot c_2 + 2
\end{aligned}
$$

   これより、$c_2 = 1, c_1 = 0, c_0 = 0$ が得られます。つまり他にはありません。
7. 3点を通る二次関数は、いつでも唯一存在するでしょうか。ここからスタートして、一般の場合を考えたいと思います。

「1.4 数学を復習しておきたい人のために」の「直線と二次関数」の問題を眺めながら、考えてみてください。

# 3.3 授業 III: いくつかの点を通る多項式関数（１）

まず、「1.4 数学を復習しておきたい人のために」の「直線と二次関数」の問題の答えを書いておきます。

## 3.3.1 直線と二次関数

1. $(0, -2)$ を通り、傾きが 3 の直線の方程式。
   - $y = 3x - 2$
2. $(0, b)$ を通り、傾きが $a$ の直線の方程式。
   - $y = ax + b$
3. 平面上の直線で、$y = ax + b$ とは表せないもの。
   - $x = a$ のように、$y$ 軸と平行な直線。
4. $ax + by + c = 0$ をみたす平面上の点 $(x, y)$ 全体が直線となる条件。
   - $a \neq 0$ または $b \neq 0$。($(a, b) \neq (0, 0)$ とも書きます。)
5. $(a, 0)$ と $(0, b)$ を通る直線の方程式。
   - $\dfrac{x}{a} + \dfrac{y}{b} = 1$ ($a \neq 0, b \neq 0$ のとき)　それ以外は、$x$ 軸 $y = 0$ または、$y$ 軸 $x = 0$ となります。
   - なお、一点になるときは、原点を通る直線がすべて条件を満たします。
6. $(a, b)$ と $(c, d)$ を通る直線の方程式。
   - $a \neq c$ のとき、$y = b \cdot \dfrac{x - c}{a - c} + d \cdot \dfrac{x - a}{c - a} = \dfrac{d - b}{c - a}(x - a) + b$
   - $a = c$ のとき、$x = a$ ただし、$(a, b) = (c, d)$ のときは、$y = m(x - a) + b$ で $m$ は任意となります。
7. $(0, 0), (1, -1), (3, 3)$ を通る二次関数。
   - $y = (x - 1)^2 - 1 = x^2 - 2x$
8. $(1, a), (2, b), (3, c)$ を通る二次関数（か、１次関数か...)。
   - $y = a \cdot \dfrac{(x - 2)(x - 3)}{(1 - 2)(1 - 3)} + b \cdot \dfrac{(x - 1)(x - 3)}{(2 - 1)(2 - 3)} + c \cdot \dfrac{(x - 1)(x - 2)}{(3 - 1)(3 - 2)}$
   - $a, b, c$ の値によっては、一次関数の場合も、定数関数の場合もありますね。例を挙げられますか。

## 3.3.2 ラグランジュの補間公式

- $y = f(x)$ の形の多項式関数だけを考えることにすると、$a = b$ のとき、$f(a) = f(b)$ となるので、いくつか、点をとったとき、$x$ 座標は異なることが必要です。
- 一般的に書きたいので、$m$ 個の点などとしたいので、$(a, b), (c, d)$ ではなく、$(a_1, b_1), (a_2, b_2), \ldots, (a_m, b_m)$ とします。添字は、index といいます。
- $m$ 個の点、$(a_1, b_1), (a_2, b_2), \ldots, (a_m, b_m)$ が与えられた時、

$$y = f(x) = c_n x^n + c_{n-1} x^{n-1} + \cdots + c_1 x + c_0$$

で

$$f(a_1) = b_1,\ f(a_2) = b_2, \ldots, f(a_m) = b_m$$

となるものを見つけたいのですが、$n$ は何にすればよいでしょうか。

- 二点、すなわち $m = 2$ では、1次、すなわち、$n = 1$、三点、$m = 3$ では、$n = 2$ ですから、$m$ 点では、$n = m - 1$ のようです。

先に進む前に、$m = 3$ すなわち、三点与えられた場合、$n = 2$ 二次関数で三点を通るものがあるかを考えておきましょう。

- $(1, a), (2, b), (3, c)$ を考えましたが、今の記号を使い、

$$(a_1, b_1), (a_2, b_2), (a_3, b_3)$$

この三点を通る関数を作ってみましょう。

$$y = b_1 \cdot \frac{(x - a_2)(x - a_3)}{(a_1 - a_2)(a_1 - a_3)} + b_2 \cdot \frac{(x - a_1)(x - a_3)}{(a_2 - a_1)(a_2 - a_3)} + b_3 \cdot \frac{(x - a_1)(x - a_2)}{(a_3 - a_1)(a_3 - a_2)}$$

２次以下ではありますが、2次とは限りません。

- $p(x) = (x - a_1)(x - a_2) \cdots (x - a_m)$
- $q_1(x) = p(x)/(x - a_1), q_2(X) = p(x)/(x - a_2), \ldots, q_m(x) = p(x)/(x - a_m)$
- $r_1(x) = q_1(x)/q_1(a_1), r_2(x) = q_2(x)/q_2(a_2), \ldots, r_m(x) = q_m(x)/q_m(a_m)$.
- このとき、

$$h(x) = b_1 \cdot r_1(x) + b_2 \cdot r_2(x) + \cdots + b_m \cdot r_m(x)$$

とすると、

$$h(a_1) = b_1,\ h(a_2) = b_2, \ldots, h(a_m) = b_m$$

となっていることが確かめられます。

- 定理の形に書いてみましょう。
- $r_1(x), r_2(x), \ldots, r_m(x)$ は、論理回路のときの、$E_1, E_2, \ldots, E_8$ と似ていませんか。

次の授業では、他の表し方はないのかなど、考えてみようと思います。

# 3.4 講義 IV: いくつかの点を通る多項式関数（２）

## 3.4.1 因数定理

$f(x) = \alpha x^3 + \beta x^2 + \gamma x + \delta$ としたとき、実数 $c$ について、$f(c) = 0$ ならば、

$$f(x) = (\alpha' x^2 + \beta' x + \gamma')(x - c)$$

と書くことができると書きました。これを一般化したものを考えましょう。

- $f(x)$ を下のような $n$ 次の多項式関数とします。

$$f(x) = c_n x^n + c_{n-1} x^{n-1} + \cdots + c_1 x + c_0,\ c_n \neq 0$$

このとき、

$$f(x) = (b_{n-1} x^{n-1} + b_{n-2} x^{n-2} + \cdots + b_1 x + b_0)(x - c) + r = q(x)(x - c) + r$$

となる、定数 $r$ と $n - 1$ 次多項式

$$q(x) = b_{n-1} x^{n-1} + b_{n-2} x^{n-2} + \cdots + b_1 x + b_0$$

が、ただ一組きまります。

- $q(x)$ の係数と、$r$ は組立除法（synthetic division）で決まります。
- 特に、$f(c) = r$ で、$f(c) = 0$ ならば

$$f(x) = (b_{n-1} x^{n-1} + b_{n-2} x^{n-2} + \cdots + b_1 x + b_0)(x - c)$$

と書けます。

## 3.4.2 一般解

$f(x) = c_n x^n + c_{n-1} x^{n-1} + \cdots + c_1 x + c_0$, $c_n \neq 0$ で、次数に制限をつけず、

$$f(a_1) = b_1,\ f(a_2) = b_2, \ldots, f(a_m) = b_m$$

となるものを、すべて見つけてみましょう。

- 前に作った、$h(x)$ は

  $$h(a_1) = b_1,\ h(a_2) = b_2, \ldots, h(a_m) = b_m$$

  を満たしていました。
- $g(x) = f(x) - h(x)$ とすると、

  $$g(a_1) = 0,\ g(a_2) = 0, \ldots, g(a_m) = 0.$$

  であることがわかります。
- 因数定理を順に繰り返し使うと、

  $$g(x) = q(x)(x - a_1)(x - a_2) \cdots (x - a_m)$$

  で、あることがわかります。
- したがって、

  $$f(x) = h(x) + q(x)p(x),\ p(x) = (x - a_1)(x - a_2) \cdots (x - a_m).$$

- 次数を考えると、$\deg f(x) \leqq m - 1$ のときは、$f(x) = h(x)$. $\deg f(x) \geqq m$ のときは、$\deg q(x) = \deg(x) - m$ となります。
- これで、どのような定理ができたのでしょうか。理解するとともに考えてみましょう。

## 3.4.3 練習問題

1. 次数が3以下の多項式 $f(x)$ で $f(0) = 4, f(1) = 3, f(2) = -6, f(3) = 1$　を満たすものは何か。
2. 次数が3以下の多項式 $r(x)$、$f(x)$ で $r(-1) = 1, r(0) = r(1) = r(2) = 0$、 および $f(-1) = 2, f(0) = -1, f(1) = 3, f(2) = -6$　となるものを求めよ。
3. 多項式 $f(x)$ は $f(1) = 2,\ f(2) = 7,\ f(3) = 1,\ f(4) = 8$ を満たすものとする。
   a. 次数が3以下である $f(x)$ を求めよ。
   b. 次数が丁度 4 である $f(x)$ を求めよ。
4. 多項式 $r(x)$ は $r(-5) = r(0) = r(5) = r(10) = 0$ and $r(15) = 1$ を満たすものとする。次数が4 のものと5 のものを一つずつ求めよ。
5. 多項式 $f(x)$ は $f(1) = -2,\ f(2) = 2,\ f(3) = 14,\ f(4) = 40$ を満たすものとする。
   a. 次数が 3 以下の多項式 $g(x)$ で $g(1) = 1, g(2) = g(3) = g(4) = 0$ となるものを求めよ。
   b. 次数が 3 以下の $f(x)$ を求めよ。
6. 多項式、$f(x)$ は $f(1) = a_1, f(2) = a_2, f(3) = a_3, f(4) = a_4$ を満たすものとする。多項式 $g(x)$ で $g(1) = a_1$, $g(2) = a_2$, $g(3) = a_3$, $g(4) = a_4$, $g(5) = a_5$ を満たすものを $f(x)$ を利用して求めたい。
   a. 多項式 $h(x)$ で $g(x) = f(x) + h(x)(x - 1)(x - 2)(x - 3)(x - 4)$ となるものが、あることを説明せよ。
   b. もし $h(x)$ が $h(5) = (a_5 - f(5))/(5 - 1)(5 - 2)(5 - 3)(5 - 4) = (a_5 - f(5))/24$　を満たせば (a) の $g(x)$ は条件を満たすことを示せ。

# 4 まとめ

- 背景にある数学の考え方について、確認したいと思います。

a. ある条件のもとで必ず成り立つ「定理」を目指す。
b. 単純（で基本的）な場合をよく調べ、それを組み合わせて、複雑な問題の解決を図る。
c. 複雑な問題を、単純な問題が組み合わさったものとして分解して考える。
d. 目標地点から逆にたどって、そこに行き着く道を探る。
e. 実験を繰り返し、予想を立てる。

# 5 おわりに

- 関連問題として、どのようなことが考えられるでしょうか。
  - 一次関数（直線の方程式）：$ax + by + c = 0$
  - 二次関数（円錐曲線の方程式）：$ax^2 + bxy + cy^2 + dx + fy + g = 0$

- 疑問におもったことや、考えたいことを話し合えればと思います。

数学の問題を理解し、考え、英語を利用し、コンピュータ・ツールも利用し、自律的に、能動的に思考し、かつ他者から学ぶ謙虚さを持ち続けることは、どのような、分野に進んでも、世界を広げる大きな力となると思います。わたしも、みなさんと、一緒に学びを楽しみ続けたいと願っています。

# 6 備考

# 6.1 スケジュール

## 6.1.1 8月29日(木)

- 9:00　ICU（大学）N館ラウンジ集合ののち、教室N-232 に移動
- 9:10〜 先生の紹介
- 9:15〜 講義① 論理回路設計（１）
- 10:15〜 休憩（N307に移動）
- 10:30〜 コンピュータの説明と個人での確認 （N307）
- 11:00 〜　講義② 論理回路設計（２）および 質疑応答（N307）
- 12:00〜 昼食休憩（大学の学食に移動）
- 13:00 集合（N307）
- 13:10〜 研究室見学
- 14:10〜 グループワーク（4〜5名の3グループに分かれて)（N307）
- 15:30〜 発表について・予告（N307）
- 16:00 終了

## 6.1.2 8月30日(金)

- 9:00　ICU（大学）N館教室N-232 に直接に集合
- 9:05〜 講義③ いくつかの点を通る多項式関数（１）（N232）
- 10:05〜 質疑応答（N232）
- 10:15〜 休憩
- 10:30〜 講義④ いくつかの点を通る多項式関数（２）（N232）
- 11:30〜 質疑応答・発表について（N232）
- 12:00〜 昼食休憩（大学の学食に移動）
- 13:00 集合（N232）
- 13:10〜 グループワーク（N232, N307）
- 13:40〜 グループワークと平行して、研究室見学
- 15:00〜 グループの発表、各グループ15分（N232）
- 15:55〜 講評と補足（N232）
- 16:15 終了

# 6.2 練習問題の略解

## 6.2.1 練習問題 2.3.4

1. $A \Rightarrow B$ （$A$ IM $B$ ともかく）ゲートを下の表で定義する。空欄を埋めよ。

| $A$ | $B$ | $A \Rightarrow B$ | $\bar{A} \vee B$ | $\overline{A \wedge \bar{B}}$ | $\bar{B} \Rightarrow \bar{A}$ |
|---|---|---|---|---|---|
| 0 | 0 | 1 | 1 | 1 | 1 |
| 0 | 1 | 1 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 | 0 | 0 |
| 1 | 1 | 1 | 1 | 1 | 1 |

このことから、次がわかる。

$$A \Rightarrow B \equiv \bar{A} \vee B \equiv \overline{A \wedge \bar{B}} \equiv \bar{B} \Rightarrow \bar{A}.$$

$\bar{B} \Rightarrow \bar{A}$ は対偶（contrapositive）と呼ばれる。

2. 次の二つの式で表される論理回路の出力が同じではない（同値でない）ことを示せ。

$$(A \wedge B) \vee C, \quad A \wedge (B \vee C)$$

（回路を作って示しても、表を書いて示しても、他の方法でも構いません。）

| $A$ | $B$ | $C$ | $(A \wedge B) \vee C$ | $A \wedge (B \vee C)$ |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 1 | 1 | 1 |
| 1 | 1 | 0 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 |

A, B が OFF（または 0）で、C が ON（または 1）のとき、$(A \wedge B) \vee C$ はON（または 1）になりますが、$A \wedge (B \vee C)$ は、OFF（または 0）になり、等しく（同値では）ありません。Aが OFF で、B, C が ON の場合も食い違っていますが、同値ではないことを示すためには、一カ所食い違っていることをしめせば、十分であることに気をつけてください。

3. $A \Rightarrow (B \vee C) \equiv A \wedge (\bar{B}) \Rightarrow C$ すなわち、どちらの論理回路も $(A, B, C)$ のすべての値について同じ出力となること（同値であること）を、三つの方法で示せ。

   a. 表を完成することで確認せよ。

| $A$ | $B$ | $C$ | $A \Rightarrow (B \vee C)$ | $A \wedge (\bar{B}) \Rightarrow C$ |
|---|---|---|---|---|
| 0 | 0 | 0 | 1 | 1 |
| 0 | 0 | 1 | 1 | 1 |
| 0 | 1 | 0 | 1 | 1 |
| 0 | 1 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 1 | 1 | 1 |
| 1 | 1 | 0 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 |

   b. 実際に、論理回路を作って確認せよ。
   ヒント：$\Rightarrow$ のゲートはありませんが、$X \Rightarrow Y$ の形の回路は、問題1をもちいて、$\bar{X} \vee Y$ などに置き換えれば良いことを注意しておきます。

   c. 練習問題 1 などの公式を使った変形により示せ。

$$A \Rightarrow (B \vee C) \equiv \bar{A} \vee (B \vee C) \equiv (\bar{A} \vee B) \vee C \equiv \overline{(\bar{A} \vee B)} \Rightarrow C \equiv A \wedge (\bar{B}) \Rightarrow C$$

   それぞれ、どの公式を使って変形したか確認してください。

4. 論理記号 ↑ NAND だけを用い（NAND 回路で）次を表わせ。NOT も使ってはいけません。

   a. $A \wedge B \equiv (A \uparrow B) \uparrow (A \uparrow B)$
   b. $A \vee B \equiv (A \uparrow A) \uparrow (B \uparrow B)$
   c. $A \oplus B \equiv (A \uparrow (B \uparrow B)) \uparrow ((A \uparrow A) \uparrow B)$
   d. $A \Rightarrow B \equiv A \uparrow (B \uparrow B)$
   e. $A \downarrow B \equiv ((A \uparrow A) \uparrow (B \uparrow B)) \uparrow (A \uparrow A) \uparrow (B \uparrow B))$

5. $(A \wedge \bar{B}) \Rightarrow C$ を以下の指示にしたがって変形せよ。

   a. $\vee$ と NOT（否定）だけを使い、$\wedge$、$\Rightarrow$ などを用いない。

$$(A \wedge \bar{B}) \Rightarrow C \equiv \overline{A \wedge \bar{B}} \vee C \equiv (\bar{A} \vee B) \vee C$$

   b. ↑ だけを使い、$\wedge, \vee, \Rightarrow$ NOT（否定）などを用いない。

$$(A \uparrow A) \uparrow ((B \uparrow C) \uparrow (B \uparrow C))$$

6. $(A, B, C)$ の入力が $(0,0,0), (0,1,0), (1,0,1)$ のときだけ $1$ でそれ以外、$0$ を出力する回路を作成せよ。

   a. AND, OR, NOT ゲートだけを用いよ。
   CNF を書いておきます。

$$(\bar{A} \wedge \bar{B} \wedge \bar{C}) \vee (\bar{A} \wedge B \wedge \bar{C}) \vee (A \wedge \bar{B} \wedge C)$$

   b. NAND ゲートだけを用いよ。
   ヒント：$\bar{X} \equiv X \uparrow X$ でしたから、問題4を用いることで、書き換えることができます。使う、NAND ゲートの数を少なくするにはどうしたらよいでしょうか。

# 6.2.2 練習問題 3.4.3

1. 次数が3以下の多項式 $f(x)$ で $f(0) = 4, f(1) = 3, f(2) = -6, f(3) = 1$　を満たすものは何か。

$$f(x) = 4 \cdot \frac{(x-1)(x-2)(x-3)}{(0-1)(0-2)(0-3)} + 3 \cdot \frac{x(x-2)(x-3)}{1 \cdot (1-2)(1-3)} - 6 \cdot \frac{x(x-1)(x-3)}{2 \cdot (2-1)(2-3)} + \frac{x(x-1)(x-2)}{3 \cdot (3-1)(3-2)}$$

```
# R Program
library(polynom)
x<-c(0,1,2,3)
y<-c(4,3,-6,1)
poly.calc(x,y)
```

```
## 4 + 11*x - 16*x^2 + 4*x^3
```

2. 次数が3以下の多項式 $r(x)$、$f(x)$ で $r(-1) = 1, r(0) = r(1) = r(2) = 0$、 および $f(-1) = 2, f(0) = -1, f(1) = 3,$ $f(2) = -6$　となるものを求めよ。

$$r(x) = \frac{x(x-1)(x-2)}{(-1)(-1-1)(-1-2)} = -\frac{1}{6}x(x-1)(x-2)$$

$$f(x) = 2 \cdot \frac{x(x-1)(x-2)}{(-1) \cdot (-1-1)(-1-2)} - \frac{(x+1)(x-1)(x-2)}{(0+1) \cdot (-1) \cdot (0-2)} + 3 \cdot \frac{(x+1)x(x-2)}{(1+1) \cdot 1 \cdot (1-2)} - 6 \cdot \frac{(x+1)x(x-1)}{(2+1) \cdot 2 \cdot (2-1)}$$

```
# R Program for r(x)
library(polynom)
x<-c(-1,0,1,2)
y<-c(1,0,0,0)
poly.calc(x,y)
```

```
## -0.3333333*x + 0.5*x^2 - 0.1666667*x^3
```

```
# R Program for f(x)
library(polynom)
x<-c(-1,0,1,2)
y<-c(2,-1,3,-6)
poly.calc(x,y)
```

```
## -1 + 3.833333*x + 3.5*x^2 - 3.333333*x^3
```

3. 多項式 $f(x)$ は $f(1) = 2,\ f(2) = 7,\ f(3) = 1,\ f(4) = 8$ を満たすものとする。

a. 次数が3以下である $f(x)$ を求めよ。

$$f(x) = 2 \cdot \frac{(x-2)(x-3)(x-4)}{(1-2)(1-3)(1-4)} + 7 \cdot \frac{(x-1)(x-3)(x-4)}{(2-1)(2-3)(2-4)} + \frac{(x-1)(x-2)(x-4)}{(3-1)(3-2)(3-4)} + 8 \cdot \frac{(x-1)(x-2)(x-3)}{(4-1)(4-2)(4-3)}$$

```
# R Program for f(x)
library(polynom)
x<-c(1,2,3,4)
y<-c(2,7,1,8)
poly.calc(x,y)
```

```
## -38 + 65.5*x - 29.5*x^2 + 4*x^3
```

b. 次数が丁度 4 である $f(x)$ を求めよ。
前の問題 a で求めた $f(x)$ を $a(x)$ と置くと

$$f(x) = a(x) + x(x-1)(x-2)(x-3)(x-4)$$

ひとつあげればよいのでこれで良いです。$a(x)$ の次数は、3以下で、第二項目次数が丁度４ですから、全体として次数は4になります。

$$f(x) = a(x) + c \cdot x(x-1)(x-2)(x-3)(x-4), c \neq 0$$

であるものとすることもできます。

4. 多項式 $r(x)$ は $r(-5) = r(0) = r(5) = r(10) = 0$ and $r(15) = 1$ を満たすものとする。次数が4 のものと5 のものを一つずつ求めよ。
次数４:

$$r(x) = \frac{(x+5)x(x-5)(x-10)}{(15+5) \cdot 15 \cdot (15-5)(15-10)}$$

次数５:

$$r(x) = \frac{(x+5)x(x-5)(x-10)}{(15+5) \cdot 15 \cdot (15-5)(15-10)} + (x+5)x(x-5)(x-10)(x-15)$$

```
# R Program for r(x)
library(polynom)
x<-c(-5,0,5,10, 15)
y<-c(0,0,0,0,1)
poly.calc(x,y)
```

```
## 0.01666667*x - 0.001666667*x^2 - 0.0006666667*x^3 + 6.666667e-05*x^4
```

分数で書くとどうなりますか。

5. 多項式 $f(x)$ は $f(1) = -2,\ f(2) = 2,\ f(3) = 14,\ f(4) = 40$ を満たすものとする。

a. 次数が 3 以下の多項式 $g(x)$ で $g(1) = 1, g(2) = g(3) = g(4) = 0$ となるものを求めよ。

$$g(x) = \frac{(x-2)(x-3)(x-4)}{(1-2)(1-3)(1-4)} = -\frac{1}{6}(x-2)(x-3)(x-4)$$

```
# R Program for g(x)
library(polynom)
x<-c(1,2,3,4)
y<-c(1,0,0,0)
poly.calc(x,y)
```

```
## 4 - 4.333333*x + 1.5*x^2 - 0.1666667*x^3
```

b. 次数が 3 以下の $f(x)$ を求めよ。

$$f(x) = -2 \cdot \frac{(x-2)(x-3)(x-4)}{(1-2)(1-3)(1-4)} + 2 \cdot \frac{(x-1)(x-3)(x-4)}{(2-1)(2-3)(2-4)} + 14 \cdot \frac{(x-1)(x-2)(x-4)}{(3-1)(3-2)(3-4)} + 40 \cdot \frac{(x-1)(x-2)(x-3)}{(4-1)(4-2)(4-3)}$$

```
# R Program for f(x)
library(polynom)
x<-c(1,2,3,4)
y<-c(-2,2,14,40)
poly.calc(x,y)
```

```
## -4 + 3*x - 2*x^2 + x^3
```

6. 多項式、$f(x)$ は $f(1)=a_1, f(2)=a_2, f(3)=a_3, f(4)=a_4$ を満たすものとする。多項式 $g(x)$ で $g(1)=a_1$, $g(2)=a_2$, $g(3)=a_3$, $g(4)=a_4$, $g(5)=a_5$ を満たすものを $f(x)$ を利用して求めたい。

a. 多項式 $h(x)$ で $g(x)=f(x)+h(x)(x-1)(x-2)(x-3)(x-4)$ となるものが、あることを説明せよ。

条件から、$f(1)=g(1), f(2)=g(2), f(3)=g(3), f(4)=g(4)$。したがって、因数定理によて、
$g(x)-f(x)=h(x)(x-1)(x-2)(x-3)(x-4)$ と書ける。

b. もし $h(x)$ が $h(5)=(a_5-f(5))/(5-1)(5-2)(5-3)(5-4)=(a_5-f(5))/24$ を満たせば (a) の $g(x)$ は条件を満たすことを示せ。

$$g(5)=f(5)+h(5)(5-1)(5-2)(5-3)(5-4)=f(5)+24\cdot h(5)=a_5.$$

# 6.3 ロジック・ゲート

## 6.3.1 参考サイト

- Half Adder and Full Adder Circuits: https://www.electronicshub.org/half-adder-and-full-adder-circuits/ (https://www.electronicshub.org/half-adder-and-full-adder-circuits/)

## 6.3.2 The Logic Lab

- Half Adder: http://www.neuroproductions.be/logic-lab/index.php?id=105072 (http://www.neuroproductions.be/logic-lab/index.php?id=105072)
- Full Adder: http://www.neuroproductions.be/logic-lab/index.php?id=105073 (http://www.neuroproductions.be/logic-lab/index.php?id=105073)

## 6.3.3 Logicly

下のリンクから、Download し、Logicly から開いてください。

- Gates: https://icu-hsuzuki.github.io/science/class/icuhs/logicly/gates.logicly (https://icu-hsuzuki.github.io/science/class/icuhs/logicly/gates.logicly)
- Full Adder: https://icu-hsuzuki.github.io/science/class/icuhs/logicly/full-adder.logicly (https://icu-hsuzuki.github.io/science/class/icuhs/logicly/full-adder.logicly)
- 23 NAND Adder: https://icu-hsuzuki.github.io/science/class/icuhs/logicly/23nand-adder.logicly (https://icu-hsuzuki.github.io/science/class/icuhs/logicly/23nand-adder.logicly)

# 6.4 参考サイト

## 6.4.1 Open Courseware

- ICU OCW 数学の方法2014: http://ocw.icu.ac.jp/ge/gen024_2014w/ (http://ocw.icu.ac.jp/ge/gen024_2014w/)
- KYOTO U OCW 農業機械学実験I 2011: http://ocw.kyoto-u.ac.jp/ja/faculty-of-agriculture-jp/5989000 (http://ocw.kyoto-u.ac.jp/ja/faculty-of-agriculture-jp/5989000)

## 6.4.2 NAND GATE

- Wikipedia（E): https://en.wikipedia.org/wiki/NAND_gate (https://en.wikipedia.org/wiki/NAND_gate)
- Wikipedia (J): https://ja.wikipedia.org/wiki/NANDゲート (https://ja.wikipedia.org/wiki/NANDゲート)