コンピュータ概論 A/B

-- Programming on Mathematica --

数学科 栗野 俊一

伝言

私語は慎むように !!

- □ Mathematica のインストール
  - ○ インストールがまだ済でいない人は直に、インストールしてください
    - ▹ TA に、申し出て、学生証と交換に DVD を借りて、作業をしてください
- □ 教室に入ったら
  - ○ 直に Note-PC の電源を入れておく
    - ▹ Network に接続し、当日の資料に目を通す
    - ▹ skype に Login する
    - ▹ Windows Update をしておこう
- □ やる気のある方へ
  - ○ 今日の資料は、すでに上っています
    - ▹ どんどん、先に進んでかまいません

今後の予定

- 今後の予定
  - 概要
    - 2013/01 : 補講日や、振替で「火曜日」がない
    - コンピュータ概論は今年(2012)一杯で終わり
    - 試験 : 試験期間中ではなく講義中で行います
- スケジュール(後ろから..)
  - 2012/12/18 (コンピュータ概論講義最終日)
    - 試験日
  - 2012/12/11 (講義最終前週)
    - 模擬試験日
  - 2012/12/04 - 2012/11/27 (二週)
    - 通常講義
  - 2012/11/20 (本日)

前回(2012/11/13)の復習

- 講義
  - Mathematica とは : 数式処理言語システム
    - 「数式」処理システム : 「数式」を対象とした高級電卓 (前回)
    - 処理「言語」システム : 機能を拡張するための仕組み(今回)
  - Mathematica の基本操作
    - 様々な数式を処理する「メタ関数」がある
    - 微積・代幾の「計算」は Mathematica でできる(「証明」は .. ?)
  - 現実・理論・計算
    - 「現実の問題」を、「数学」と「計算機」によって、「人間」が解く
- 実習
  - Mathematica の基本操作
    - 起動と終了 : 他のアプリケーションと同様
    - 式の入力と評価 ([SE]=[Shift]+[Enter]) : 式を入れ、[SE]すれば結果が出る
  - Help の利用方法
    - ヘルプセンターの利用法 : [ヘルプ] >> [ドキュメントセンター]
    - バーチャルブックの利用法 :[ヘルプ] >> [バーチャルブック]

本日(2012/11/20)の予定

- □ 講義
  - ○ Mathematica によるプログラミング基礎
- □ 実習
  - ○ [演習 1] Mathematica の変数の利用法
  - ○ [演習 2] Mathematica の関数の作成方法
  - ○ [演習 3] 課題の作成

本日(2012/11/20)の課題

- 今週 (2012/11/20) の課題
  - 次のファイルを提出しなさい
    - 表題 : ペアノの方法による「有理数の差」の関数 qsub を定義しなさい
    - ファイル名 : 20121120-QQQQ.nb (QQQQ は学生番号)
    - 詳しくは、配布した nat.txt の内容を参照
- 先週 (2012/11/13) の課題
  - 次のファイルを提出しなさい
    - 表題 : Mathematica の課題
    - ファイル名 : 20121120-QQQQ.nb (QQQQ は学生番号)
    - 詳しくは、配布した sample-20121120.nb の内容を参照

システム

□「システム」とは

- ○「対象」と「操作」の対 (この二つは完全に区別される)
  - ▹「対象(Object)」: 興味の対象となる集合の要素(広い意味での「数」/「何」)
  - ▹「操作(Operation)」: 「対象」の要素を「扱う」もの(広い意味での「関数」/「どうする」)

表 1: システムの例

システム	対象	操作	事例
計算 (電卓)	数	演算 (四則/関数)	$1+1=2,\ \sin\left(\frac{\pi}{3}\right)$
代幾	ベクトル	行列 (線形変換)	$A\vec{v}=\vec{u}$
微積	関数	極限・微分・積分操作	$\int f(x)dx = F(x) +$
数学 (入門)	数学の対象 (集合)	論理 (集合演算)	「仮定」から「結論
$\TeX$	文章/文字列	マクロ	`\frac{1}{2}` $= \frac{1}{2}$
Excel の計算	セルの値	式	他のセルの値から値
Mathematica	数式 (数を含む)	Mathematica の関数	$D[Sin[x], x] = Cos[$
OS	ファイル	ファイル操作	ファイルのコピー/削
アプリケーション	データファイル (テキスト)	プログラム (エディタ)	ファイルの内容の変
計算機	データ	プログラム	入力データから出力
料理	食材 (素材の味)	調理 (調味量)	食物 (味) から食物 (
現実	世界	人間 (愛)	愛は世界を救う (ア

システム図

- □ 個々の役割
  - ○ 興味の対象は Object
  - ○ 対象を操作するために Operation(方法)を利用
  - ○ Operation の仕組みを知る事が学問する事
- □ 学習
  - ○ 数を操作するために計算方法を学ぶ
  - ○ 食事を作るためには料理法を学ぶ
  - ○ 情報を得るためには情報検索方法を学ぶ
- □ 応用
  - ○ 学んだ内容結果を実地に適用する
  - ○ 学んだ操作方法を利用して、対象を操作する

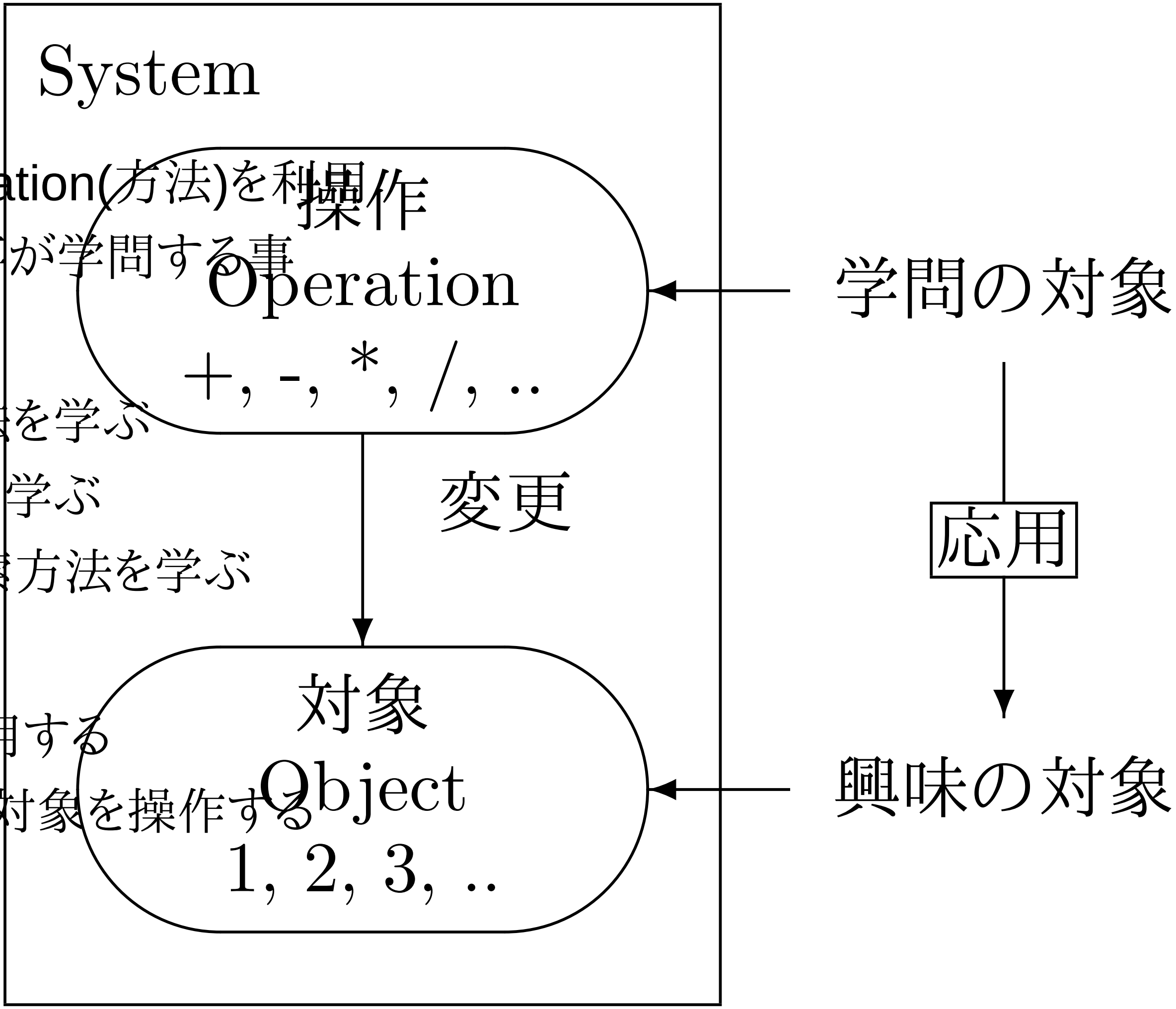


図 1: システム図

メタ概念

□ メタ〜(超〜)

○「『システムの操作』を操作する」仕組み

▷「メタ○○」は、「○○の○○」と置き換える事ができる(cf. メタ情報)

表 2: メタシステムの例

システム	対象	操作	メタ論
計算	数	演算 (四則/関数)	汎関数/関数操作 (微積/関数解析)
代幾	ベクトル	行列 (線形変換)	行列の計算 (線形空間)
数学	数学の対象	論理	帰納法, 背理法 (証明論/ヒルベルトのプロ
$\TeX$	文章/文字列	マクロ	マクロ定義
Mathematica	数式	Mathematica の関数	関数定義 (今日の課題)
計算機	データ	プログラム	プログラミング
現実 (料理)	世界 (食物)	人物 (コック)	人物 (コック) の評判

○「対象を操作する操作方法」そのものを操作する

▷「対象の操作」を「操作方法の操作」で「間接的に操作」する

▷「間接」なので、「強力」だが「理解/応用するのは難しい」

○ メタ論の例

▷ あの議論(操作)の内容(対象)は解らないが、矛盾した議論(メタ)なので主張は正くないだろう(背理法)

▷ あの料理人(操作)の料理(対象)は食べた事はないが、三つ星レストランのシェフ(メタ)なので、美味しいだろう(評判)

▷ あの政治家(操作)の政策(対象)は知らないが、顔が嫌い(メタ)だから票は入れない(感情)

▷ 平面幾何学の定理は、点と線を入れ替えても成立する(双対原理)

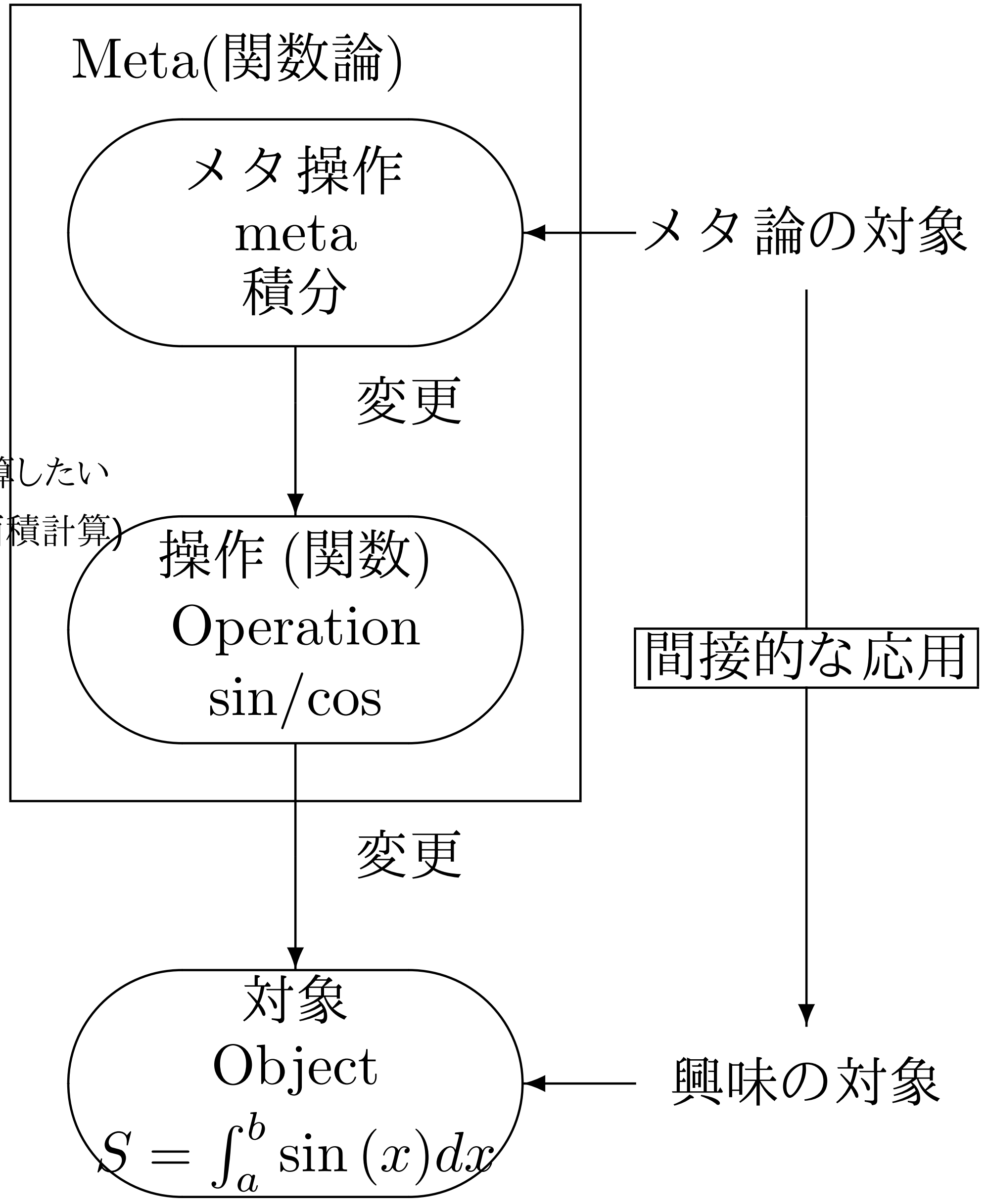
メタシステム図

□ メタシステム

- ○ 特定なシステム上に構成される
- ○ システムの操作がメタシステムの対象
  - ▹ 操作の操作により間接的に対象を操作

□ 関数論の例

- ○ 対象:数 / 操作:関数 / メタ:微分積分
  - ▹ 関数 f と x 軸、直線 x=a, x=b で囲まれた面積が計算したい
  - ▹ 細分化して総和を計算してもよいが.. (cf. Excel の面積計算)
  - ▹ 定積分で計算可能
  - ▹ 原始関数を求める必要がある(メタ論の利用)



メタ関数と Mathematica プログラミング

- メタ関数とは
  - 関数を引数、あるいは結果とする関数 (汎函数とも言う)
    - 普通の関数 (数を引数にして結果を数とする) の例 : Sin[x]
    - メタ関数の例 : D[Sin[x]] → Cos[x] (引数も結果も関数)
- Mathematica プログラミング
  - メタ関数を定義する事
    - メタ関数には広義の意味で普通の関数を含めてしまう
- Mathematica は数式「処理言語」システム
  - 新しい関数が定義できる
    - 「定義のための表現形式」が含まれている (言語システム)
  - 「言語」システムの定義には、「構文規則」と「意味規則」が必要
    - 構文規則 : どのような「(表現)形:文字列」が適切か
    - 意味規則 : その「(表現)形」はどんな「意味」に対応するか
- Mathematica の関数定義とは
  - 関数を表すシンボルに、その関数の意味を表す式を「割り当てる」事
    - 「add[x_,y_] := x+y」とすれば、「add[2,3]が5になる」ように定義した事になる

構文(表現)と意味

- 構文規則
  - 「文字の並び」を「何の表現か?」と「判断するための規則」
    - 「123 + a」は、「123」、「a」、「+」に分解される
    - 「123」は、「1」、「2」、「3」という「数字列」なので「数値」
    - 「a」は、「a」という「英文字から始まる文字列」なので「変数」
    - 「+」は、「足し算」を表す「演算子」
    - 「123 + a」全体は、「式」になる
  - 「どんな文字から構成されるか」という基準で区別される
    - 「何か」を表す、「表現方法」の規則
    - 規則に従って、「文字列」をみると、幾つかの「構文要素」に分解される
- 意味規則
  - 構文規則に対応した「意味」を決める規則
    - 「123」という「数字列」は、123 という「整数値」
    - 「a」という「シンボル」は、a という「変数」
    - 「123 + a」は、「二つの数を加える」という「文字式」
- 言語
  - 構文規則 (いわゆる文法)と、意味規則(「辞書」ならびに、「意味」) から成る
    - 「1 + 2」は「構文的」には、「「1」と「2」の「足し算」」を表す「式」となる
    - 「1 + 2」は「意味的」には、「3」(1, 2 は数値を表し、+ は足し算を表すから)

Mathematica Notebook

- Mathematica Notebook (*.nb)
  - Mathematica の計算結果を保存する形式
  - 結果の保存方法
    - [ファイル]→[別名で保存] : 好きな名前で保存できる
    - [ファイル]→[保存] (Ctrl-S) : 今の名前で内容を更新する(古い内容は失われる)
  - 結果の利用方法
    - [ファイル]→[開く] (Ctrl-O) : 以前に保存した内容を読み込む
- 電卓としての利用
  - 「式」を入れて [Shift]+[Enter] (以下、[SE]) すると計算
    - 入力した式は In[番号] の形で表示される
    - 計算した結果は Out[番号] の形で表示される
    - 番号は、[SE] の順番につけられる
  - [SE] の効果
    - 「式」の評価が行われる
    - 「番号」が増える
    - In[番号]/Out[番号]が、それぞれ「定義」される

変数とその値

- □ シンボル (構文的な要素)
  - ○ シンボルとは
    - ▹「英字」から始まり「英数字」からなる文字列の事 (a, abc, a1, a12z)
    - ▹ (注意) 「英字」には、「ギリシャ文字」も含まれる (π, α)
    - ▹ cf. 「数字」から始まる文字列はシンボルではない (2a は 2 × a となる)
  - ○ Mathematica は、大文字で始まるシンボルの幾つかを事前に定義している
    - ▹ (注意) 自分が利用する場合は、小文字で始まるシンボルを利用する事
- □ 変数 (意味的な要素)
  - ○ 変数とは
    - ▹ 名前として「シンボル」をもち、「値を持つ事ができるもの」
    - ▹「変数名」は、「変数」の名前で、それは「シンボル」である
  - ○ 変数の(即)値とは
    - ▹ (現時点で)変数に対応している「式(の評価結果)」の事
    - ▹「環境」は、「変数名」と「変数の値」の対応表をもっている
  - ○ 変数の値の参照
    - ▹「シンボル」の形を示す物を入れると、「その値」が評価され、その結果が表示される
    - ▹「値」が設定されていない場合は、その「変数自身」が「変数の値」となる
    - ▹ ?変数名で、「変数の値」である「式」を見る事ができる

変数への代入と定義

- 代入と定義
  - 共に変数に値を「割り当て」る
  - 代入 :「変数 = 式」
    - 「式」は「即時評価」される。その評価結果が「変数の値」となる
  - 定義 :「変数 := 式」
    - 「式」は「遅延評価」される。その「式」そのものが「変数の値」となる
  - 変数の値は何度でも変更できる
- 環境
  - 変数名とその値の対応表をもっている
    - 代入も定義も、その対応表を書換えている
    - 書き換える値が違うだけ
- 「式」の評価(変数の場合)
  - 式の中に変数名が含まれていた場合に、それを変数の値に書き換える

関数の利用

- Mathematica の関数
  - Mathematica の関数とは
    - 「シンボル[引数列]」の形で表現され、値を持つ事ができる「もの」
    - cf. Sin[Pi] / f[3] / g[x,y^4]
  - 関数の評価
    - 「シンボル[引数列]」の形を、その値に置き換える
- 引数のパターンマッチ
  - 「_」(アンダースコア)を利用して「式」の「抽象パターン」が表現できる
  - 例1
    - next[0] := 1
    - next[1] := 2
    - next[_] := 0
  - 例2
    - fib[1]=1
    - fib[2]=1
    - fib[x_]:=fib[x-1]+fib[x-2]

自然数

- □ ペアノの公理
  - ○ 自然数を定義する公理
    - ▹ 0 は自然数である
    - ▹ n が自然数なら n+1 も自然数である
    - ▹ 上記の二つ以外に自然数はない
- □ Mathematica でペアノの公理の形の「自然数」を考える
  - ○ 自然数 (これを「ペアノ形式の表現」と呼ぶ事にする)
    - ▹ 0 は 0 で表現
    - ▹ n + 1 は s[n] で表現
  - ○ 自然数の和
    - ▹ padd[0,x_] := x
    - ▹ padd[s[x_],y_] := s[padd[x,y]]
  - ○ これを繰り返すと、「分数(有理数)」まで表現できる
    - ▹ 「実数」を表現するには、「収束概念」が必要になる

自然数による数の表現 (nat.txt)

- □ 自然数
  - ○ 0 と +1 (サクセッサー) のみで作る
    - ▹ 3 = 0+1+1+1 = s[s[s[0]]]
- □ 整数
  - ○ 自然数の二つ組 (m,n) で整数 z を表現する
    - ▹ z = m - n => pp[m,n]
    - ▹ 同じ整数に対する異なる(m,n) 組があるので、同値類(〜)を作る
    - ▹ 例 pp[4,2] 〜 pp[3,1] 〜 pp[2,0]
- □ 有理数
  - ○ 自然数と整数の組 (z,n) で整数 q を表現する
    - ▹ q = z/n => qq[z,n]
    - ▹ 同じ有理数に対する異なる(z,n) 組があるので、同値類(〜)を作る
    - ▹ 例 qq[18,12] 〜 pp[9,4] 〜 pp[6,2]

「ペアノ形式」と普通の表現の関係

表 3:「S 表現」と普通の表現の関係

概念	普通の表現 (例)	ペアノ形式 (例)	コメント
0	0	0	0 は自然数
自然数 (0 以外)	1,2,3,..	s[0], s[s[0]], s[s[s[0]]], ..	x が自然数なら s[x] も
n の次	n + 1	s[n]	s はサクセッサ (後継)
和	1 + 2	padd[s[0],s[s[0]]]	
差	1 - 2	psub[s[0],s[s[0]]]	引く数の方が大きい場
積	1 * 2	pmul[s[0],s[s[0]]]	
商	1 / 2	pdiv[s[0],s[s[0]]]	整数割り算 (小数点以
最大公約数	gcd(1,2)	pgcd[s[0],s[s[0]]]	
整数	1, -1	pp[s[0],0], s[0,s[0]]	自然数の対を同値類で
整数の正規化		zbar	ペアのどちらか一方を
整数の四則	+,-,*,/	zadd,zsub,zmul,zdiv	
有理数	1/2	qq[pp[s[0],0],s[s[0]]]	分子は整数で、分母が
有理数の正規化		qbar	約分する
有理数の四則	+,-,*,/	qadd,qsub,qmul,qdiv	

- 可換性(well defined)
  - 通常の自然数とペアノ形式の自然数は ntop, pton で同型になっている