データ可視化 I

担当 : 坂本 尚久

計算科学演習A1

2016年4月28日

※本資料は昨年の陰山先生の資料に坂本が加筆したものです。

内容

1. 可視化とは
   - １次元データの可視化
   - ２次元データの可視化
   - ３次元データの可視化
2. gnuplot入門
   - X Windowシステムの設定
   - 演習１～４
3. 課題

可視化

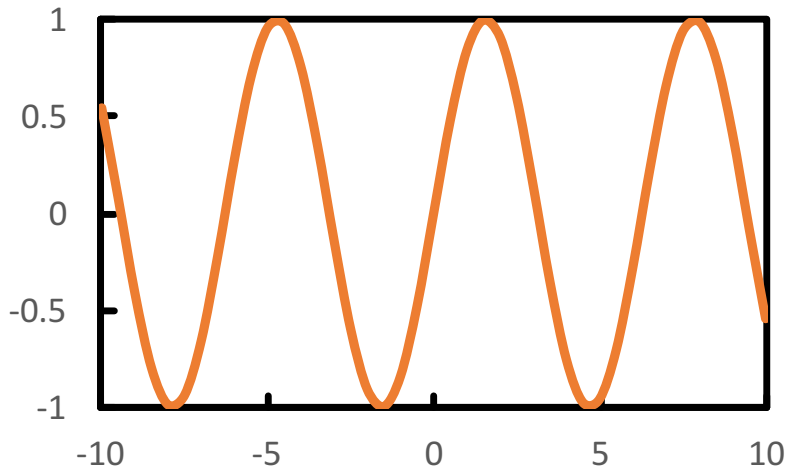
- 情報可視化
  - Information Visualization
- データ可視化・科学的可視化
  - Data Visualization
  - Scientific Visualization
- 視覚的分析
  - Visual Analytics

可視化

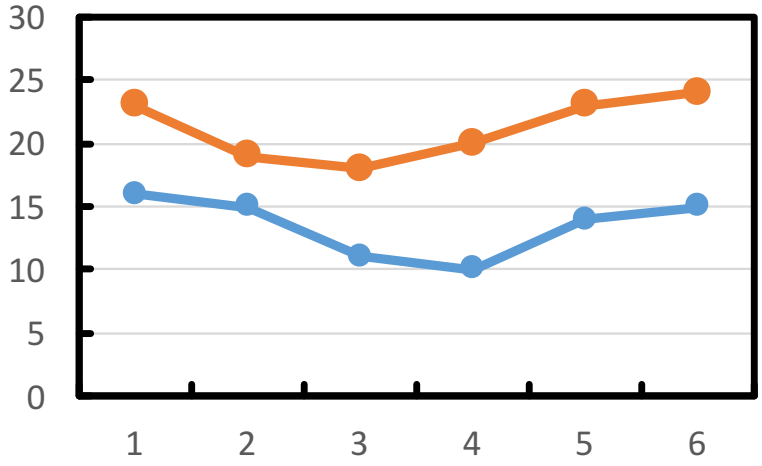
- 情報可視化
  - Information Visualization
- データ可視化・科学的可視化
  - Data Visualization
  - Scientific Visualization
- 視覚的分析
  - Visual Analytics

１次元データの可視化

- xの関数 $f(x)$
  - $f(x) = \sin x$



- 計測データ
  - 23, 19, 18, 20, 23, 24
  - 16, 15, 11, 10, 14, 15



１次元データの可視化

- $f(x) = \sin x$ に対する $f^{\times 10}(x)$ はどのような関数であろうか？

  – ただし、

$$y = f(x)$$

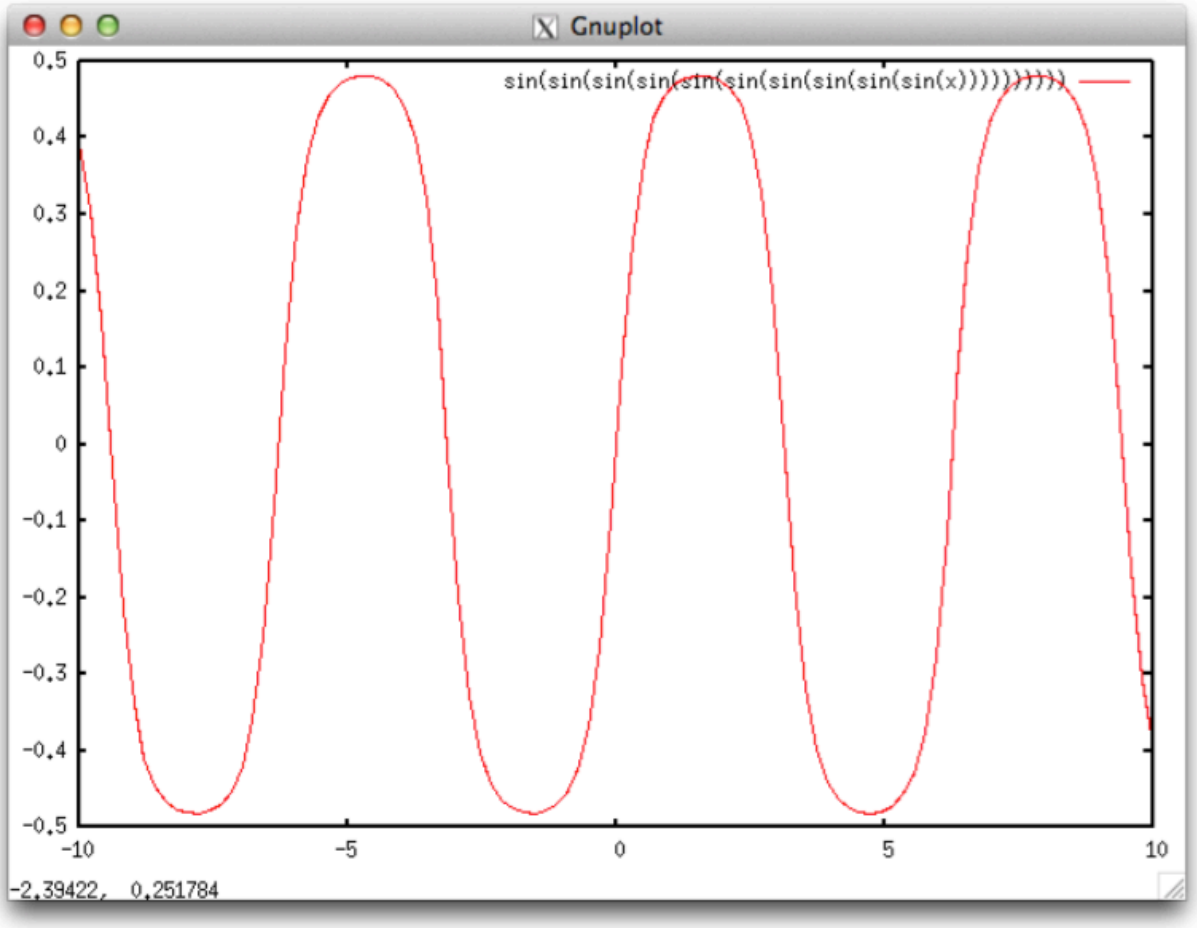
  に対して、f の値域が定義域に含まれるとき、

$$f^{\times 2} := f((f(x)), \quad f^{\times 3} := f(f(f(x))), \quad \cdots$$

  等と定義する。

1次元データの可視化

$$y = \sin(\sin(\sin(\sin(\sin(\sin(\sin(\sin(\sin(\sin(x)))))))))))$$



クイズ

- x を0以上の実数として、x の x乗、つまり

$$f(x) = x^x \quad (x \geq 0)$$

はどんな関数であろうか？

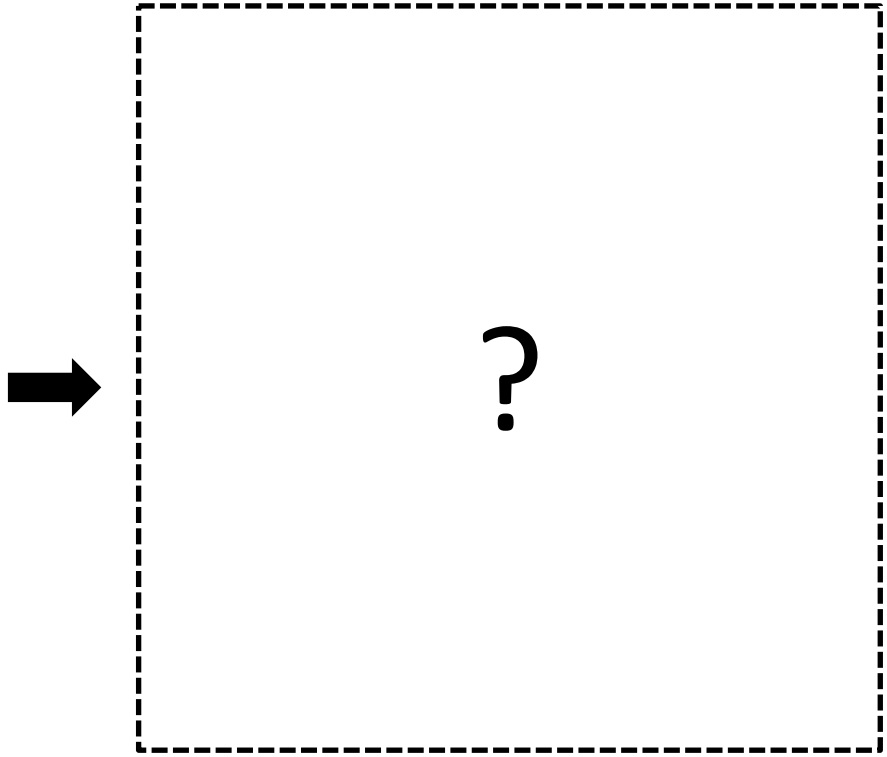
– 最大値/最小値をとる x は？
– $x = 0$ の時の値 $f(0) = 0^0$ は何だろう？

→ 後で演習

２次元データの可視化

- xとyの関数 $f(x,y)$
- 数値データ

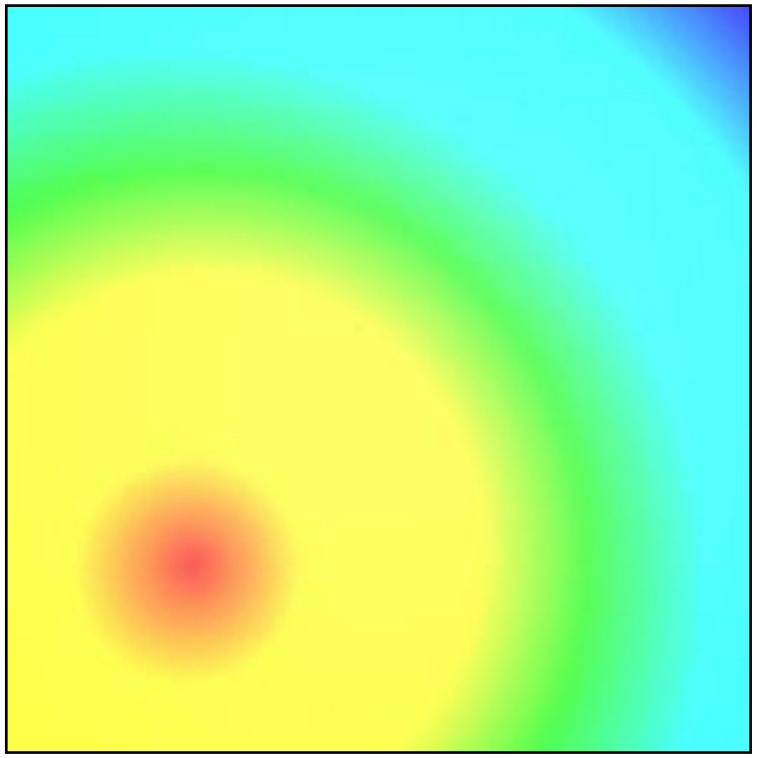
5	7	5	2	0
12	15	12	9	2
17	20	17	12	5
20	25	20	15	7
17	20	17	12	5



２次元データの可視化

- xとyの関数 $f(x,y)$
- 数値データ

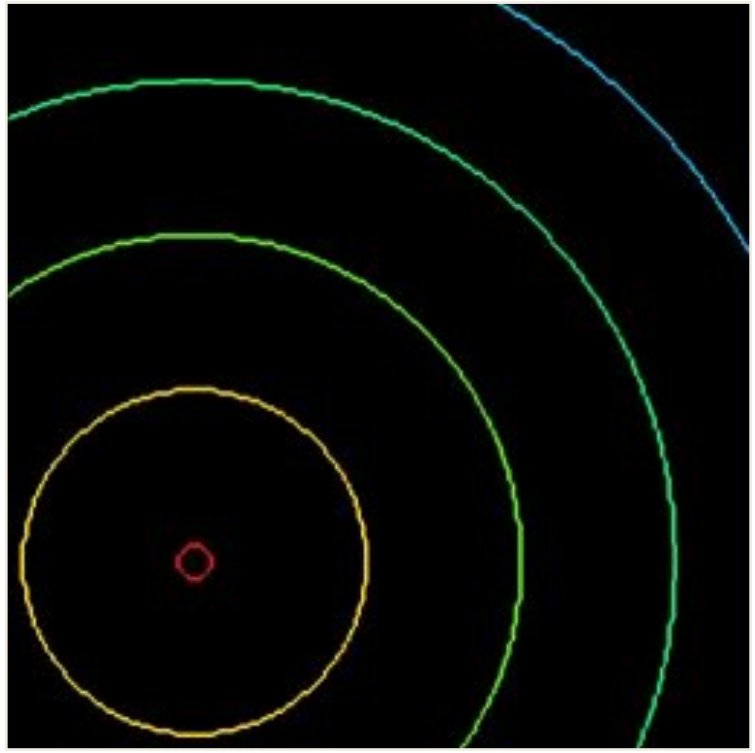
5	7	5	2	0
12	15	12	9	2
17	20	17	12	5
20	25	20	15	7
17	20	17	12	5



２次元データの可視化

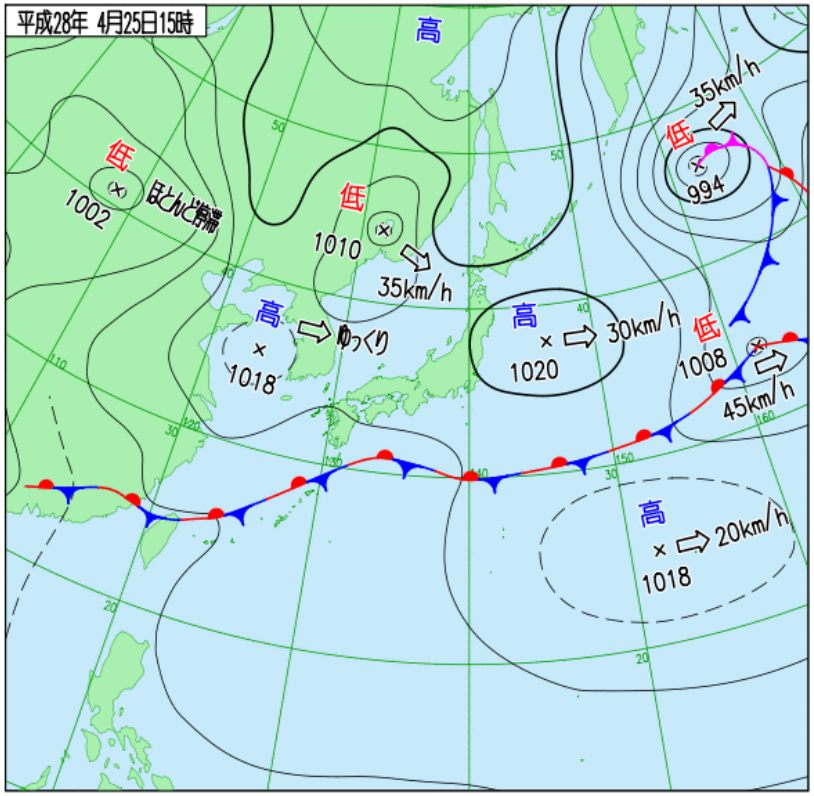
- xとyの関数 $f(x,y)$
- 数値データ

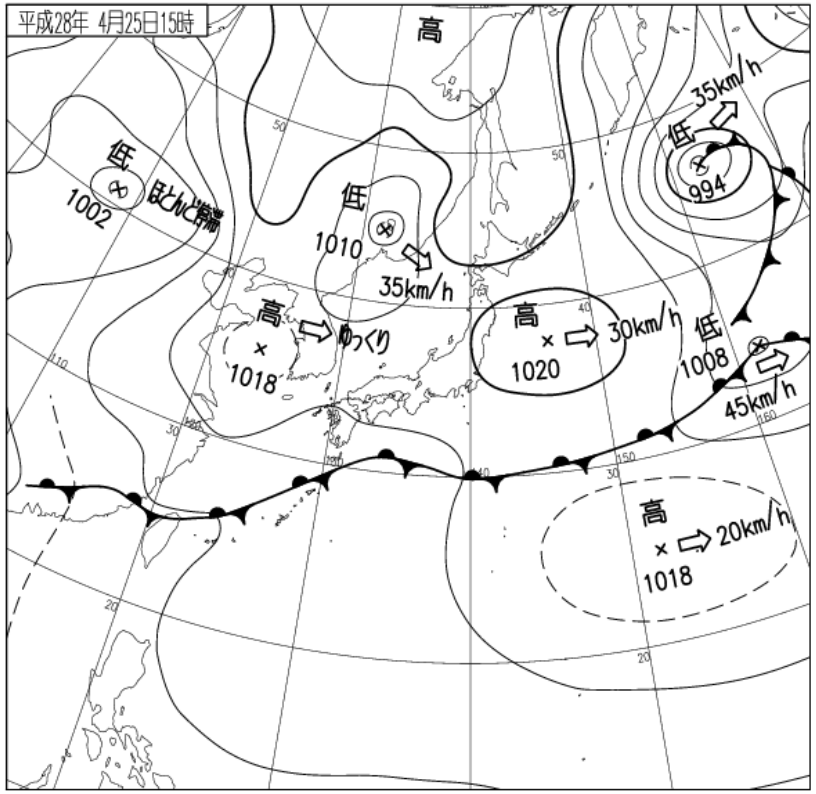
5	7	5	2	0
12	15	12	9	2
17	20	17	12	5
20	25	20	15	7
17	20	17	12	5



2次元データの可視化

- 例）天気図（気圧配置図）
  - 地表面での大気の圧力pの分布$p(x,y)$の等高線



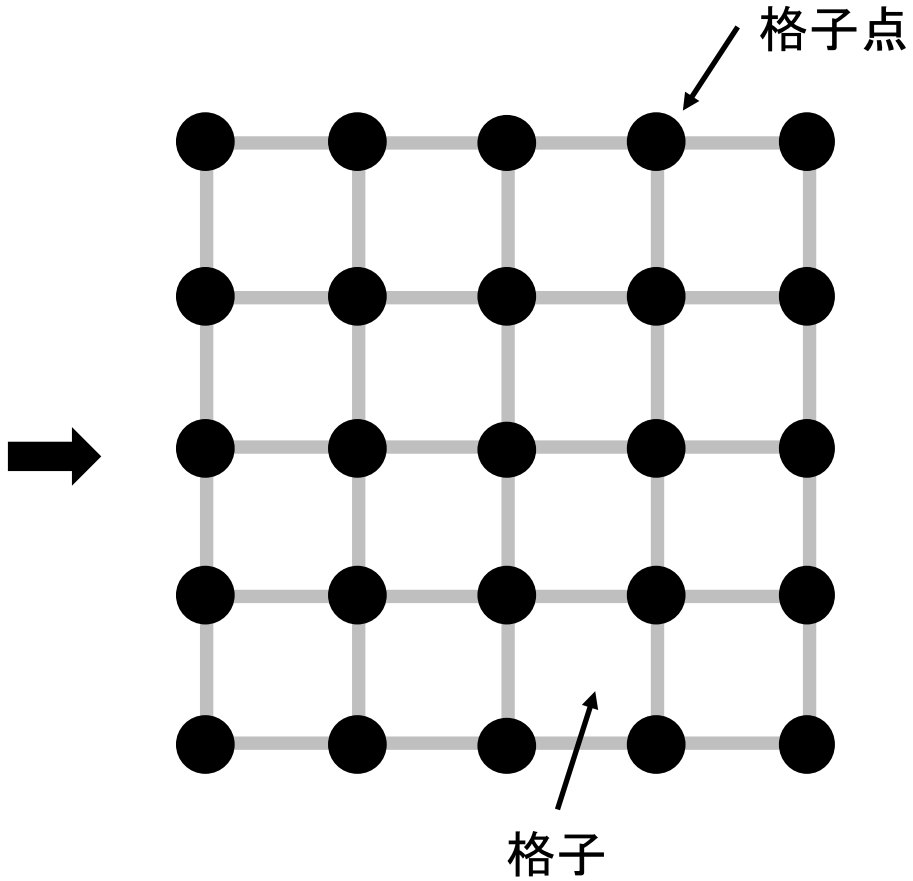


http://www.jma.go.jp/jp/g3/

等高線の描画アルゴリズム

- 等高線
  - 指定された値 S を持つ点の集合

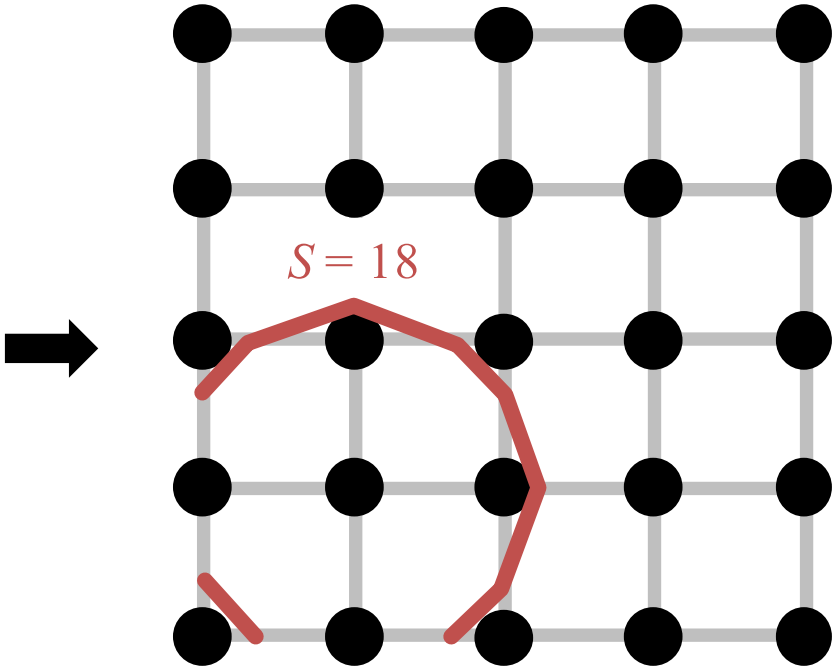
5	7	5	2	0
12	15	12	9	2
17	20	17	12	5
20	25	20	15	7
17	20	17	12	5



等高線の描画アルゴリズム

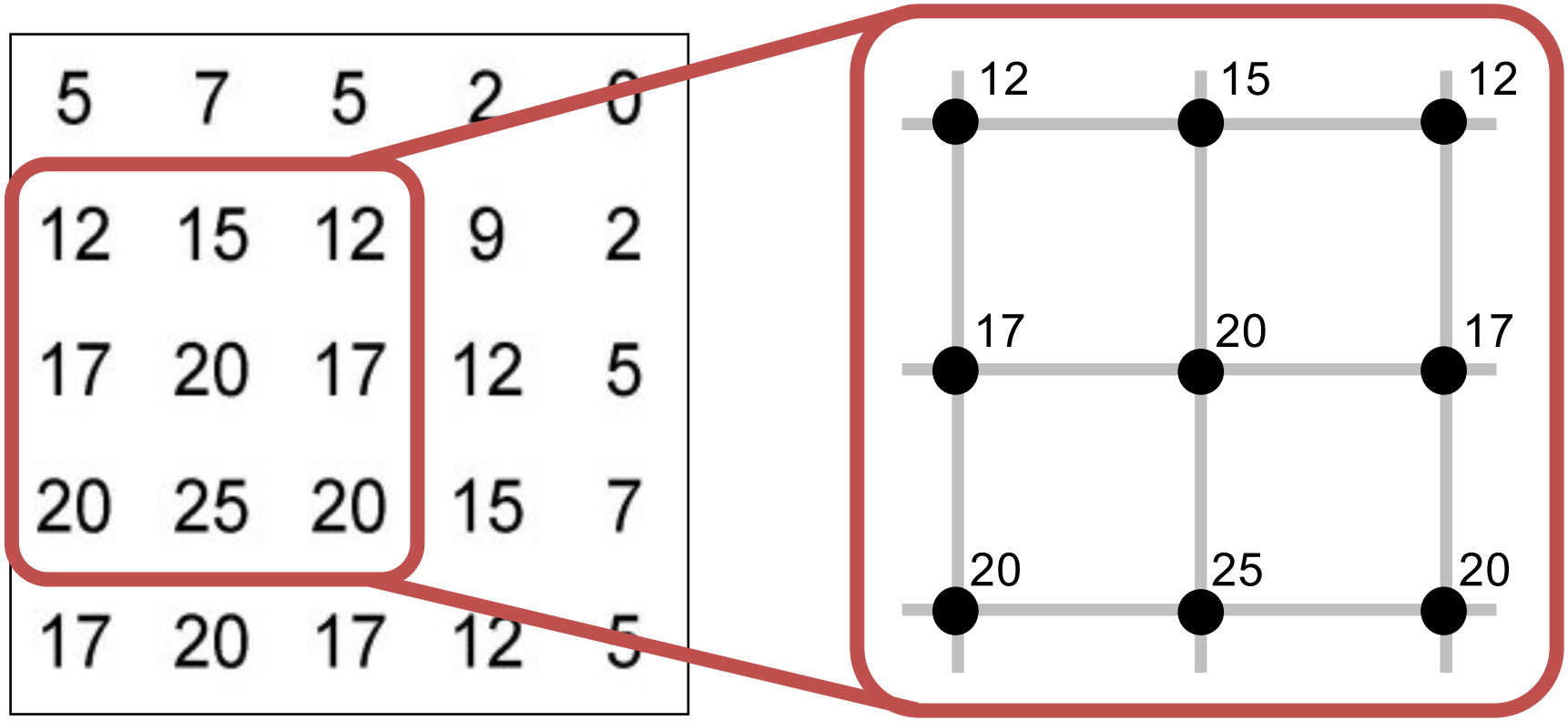
- 等高線
  - 例）$S = 18$の点の集合

5	7	5	2	0
12	15	12	9	2
17	20	17	12	5
20	25	20	15	7
17	20	17	12	5



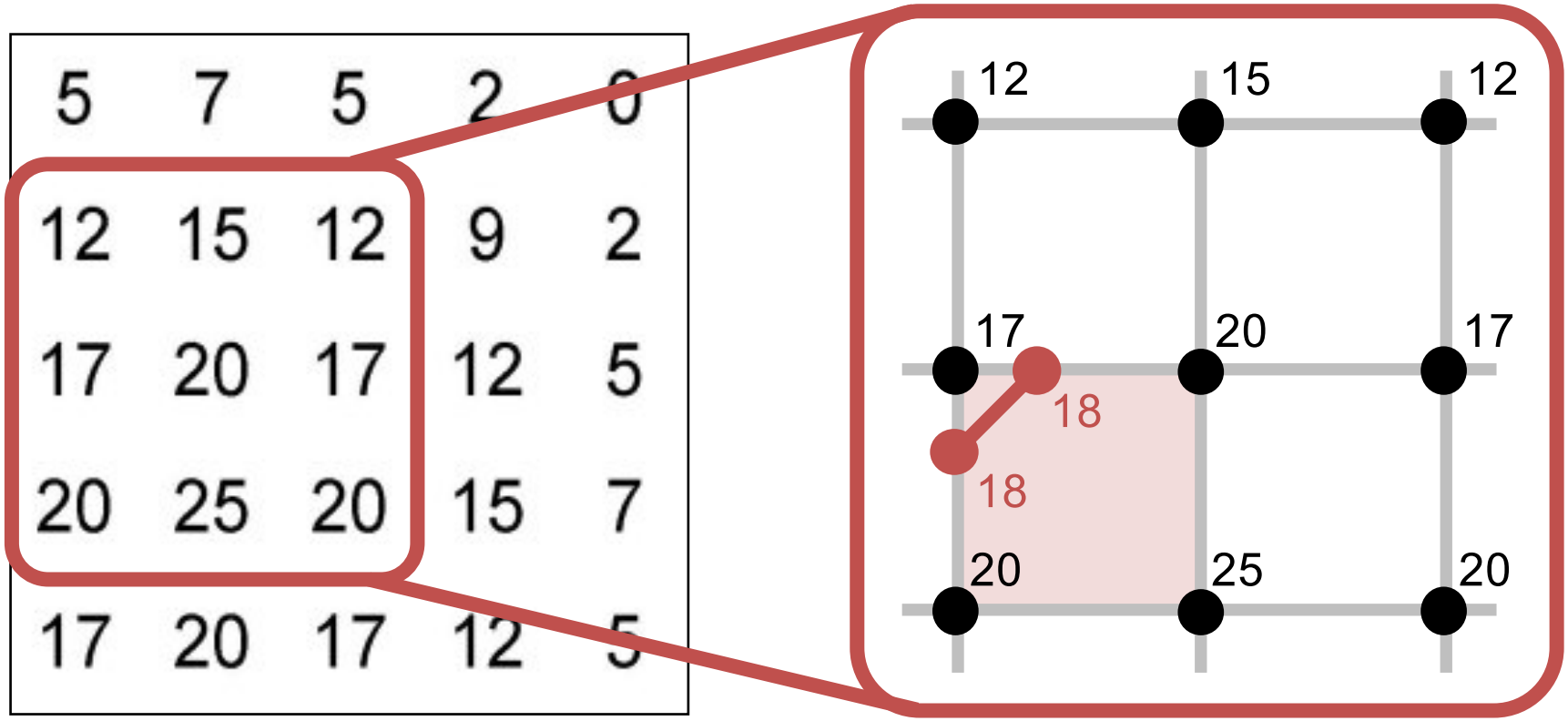
等高線の描画アルゴリズム

- 等高線
  - 例）$S = 18$の点の集合



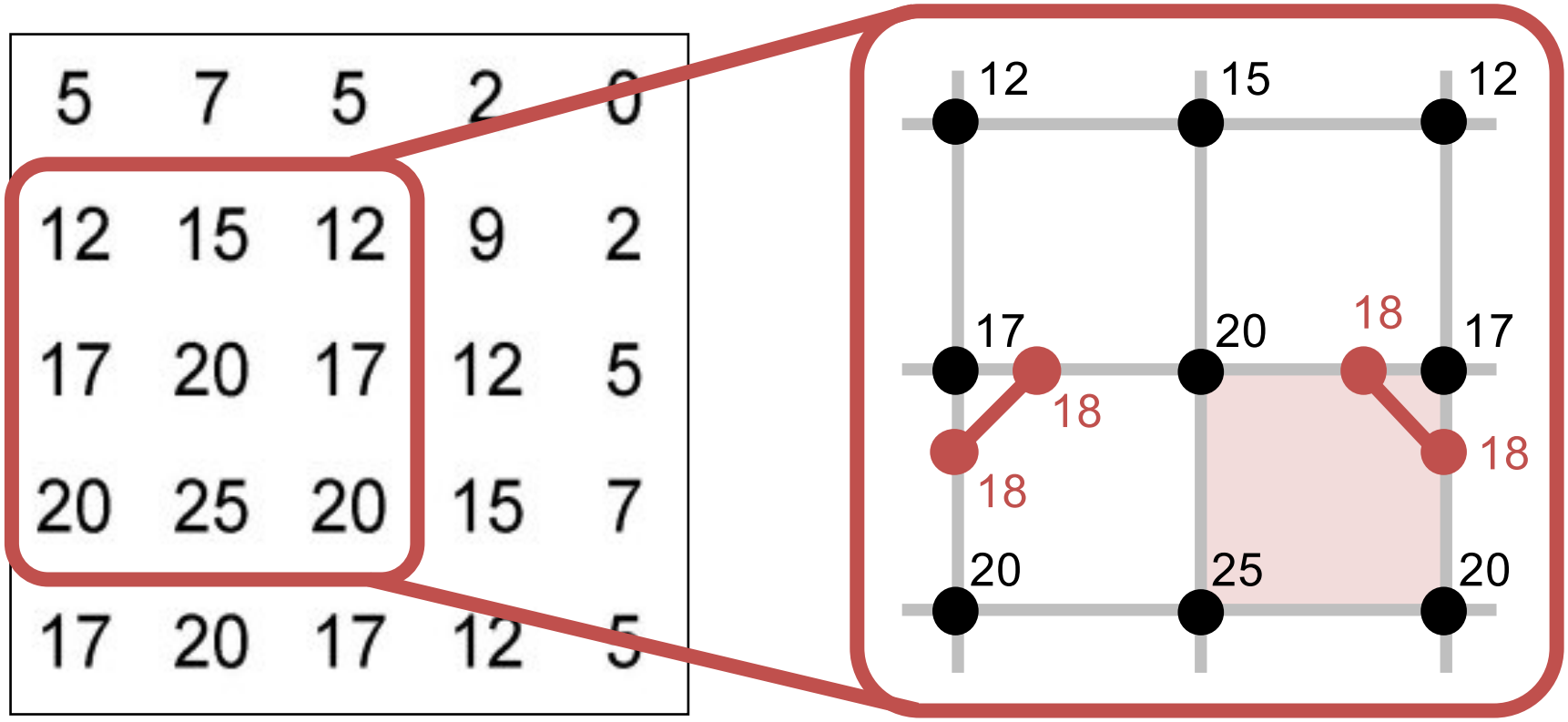
等高線の描画アルゴリズム

- 等高線
  - 例）$S = 18$の点の集合



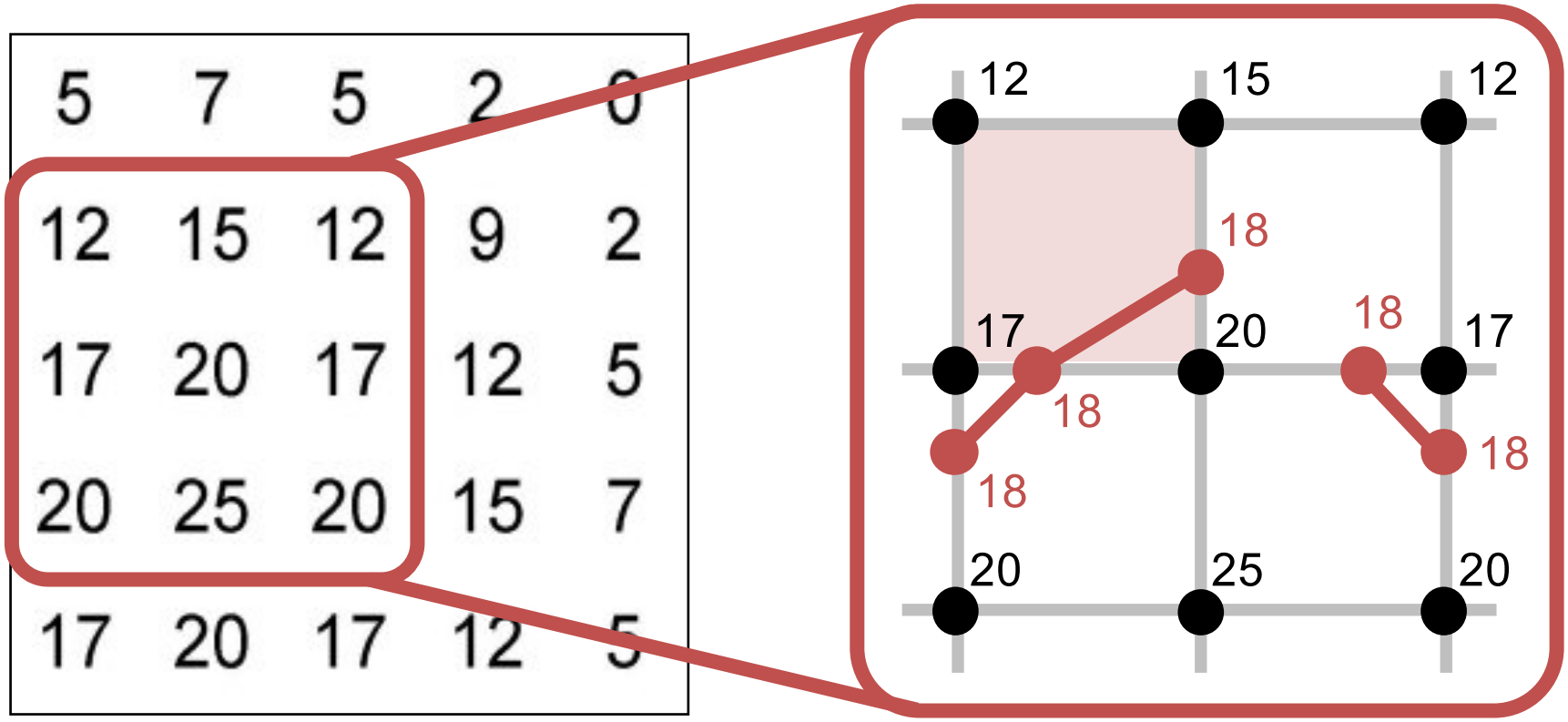
等高線の描画アルゴリズム

- 等高線
  - 例）$S = 18$の点の集合



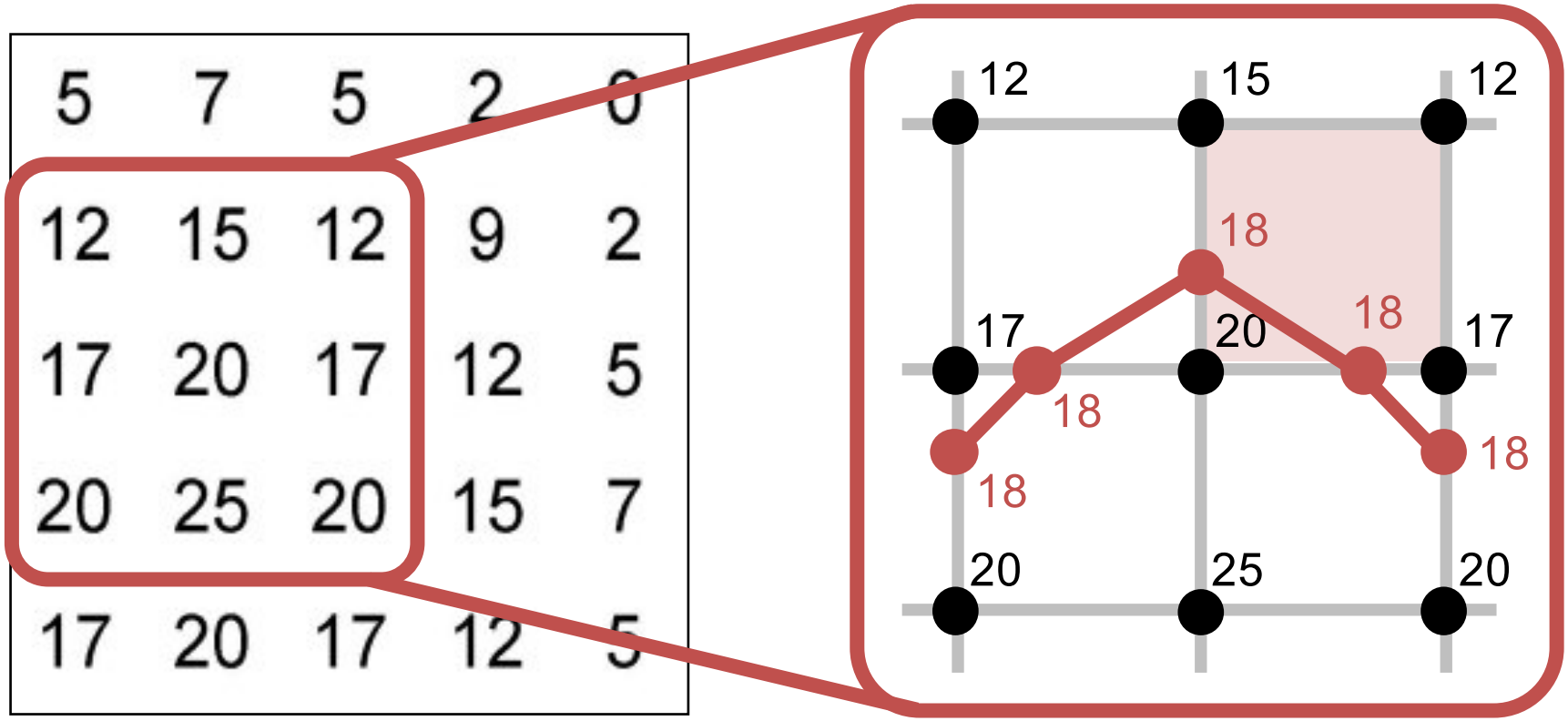
等高線の描画アルゴリズム

- 等高線
  - 例）$S = 18$の点の集合



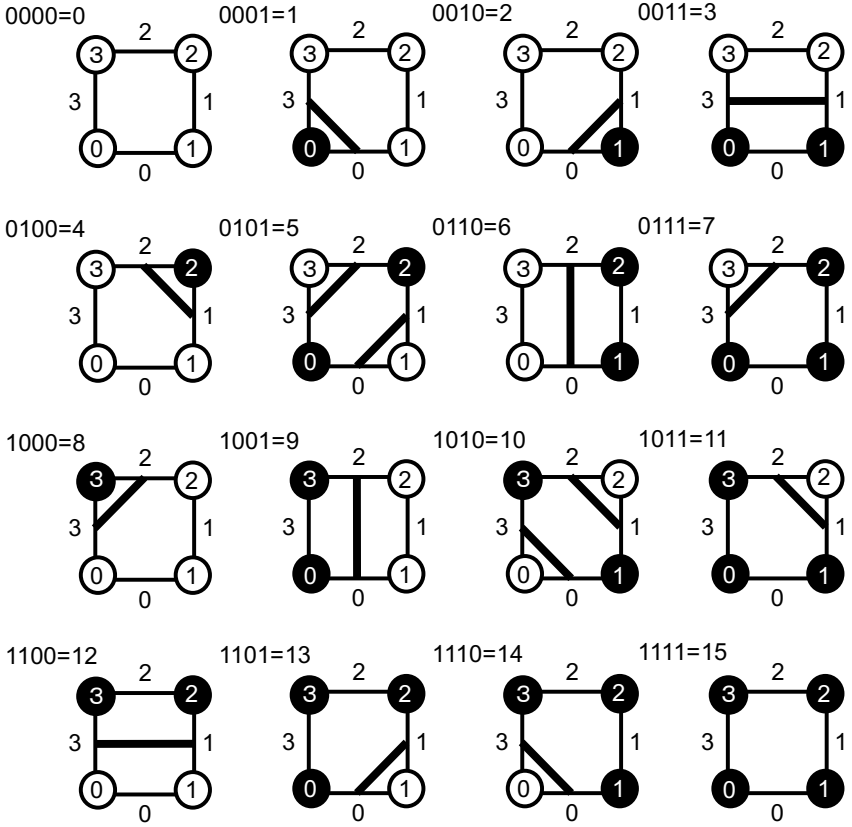
等高線の描画アルゴリズム

- 等高線
  - 例）$S = 18$の点の集合



等高線の描画アルゴリズム

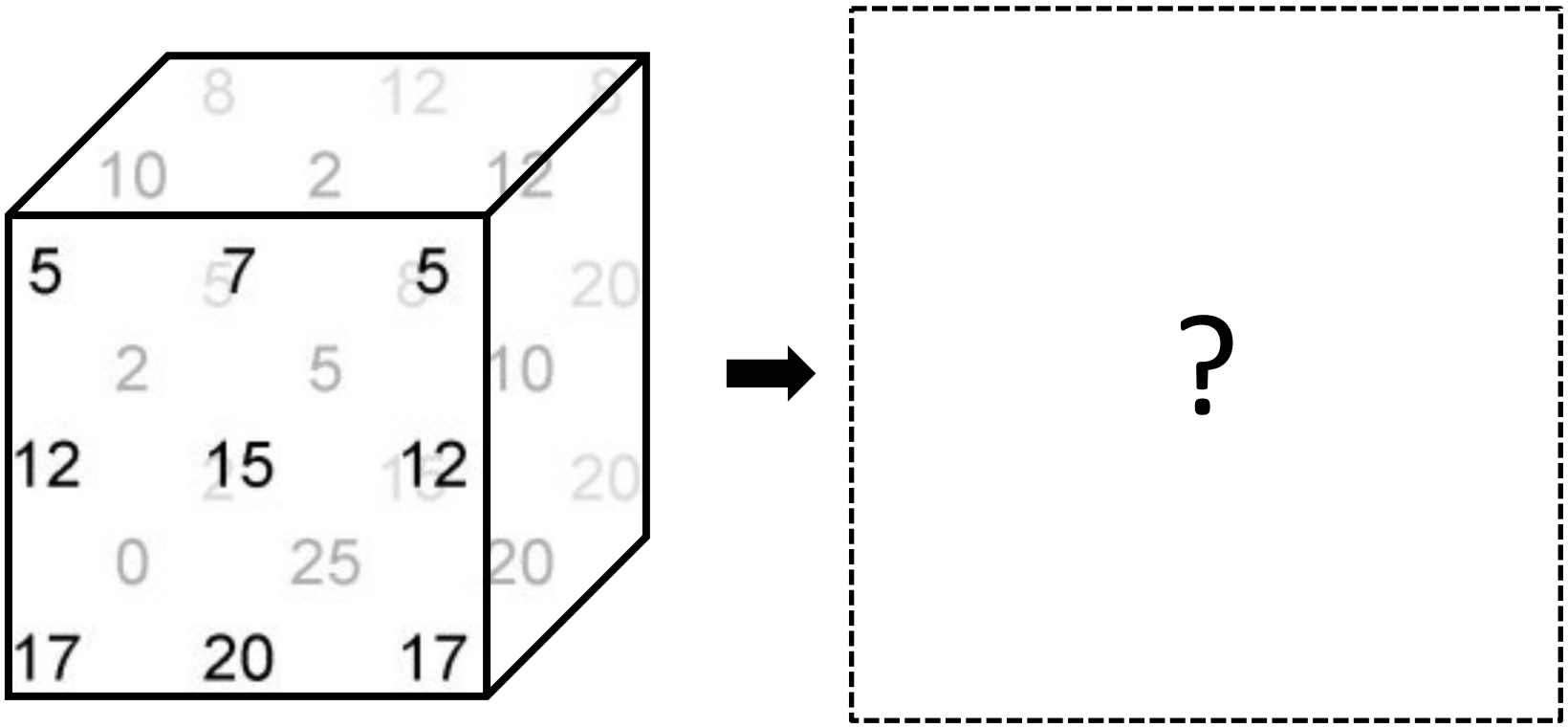
- Marching Squares
  - 稜線との交差パターン（2^4=16 通り）



0000 = 0	-1	-1	-1	-1
0001 = 1	0	3	-1	-1
0010 = 2	0	1	-1	-1
0011 = 3	3	1	-1	-1
0100 = 4	1	2	-1	-1
0101 = 5	1	0	3	2
0110 = 6	2	0	-1	-1
0111 = 7	3	2	-1	-1
1000 = 8	2	3	-1	-1
1001 = 9	0	2	-1	-1
1010 = 10	1	2	0	3
1011 = 11	1	2	-1	-1
1100 = 12	1	3	-1	-1
1101 = 13	0	1	-1	-1
1110 = 14	0	3	-1	-1
1111 = 15	-1	-1	-1	-1

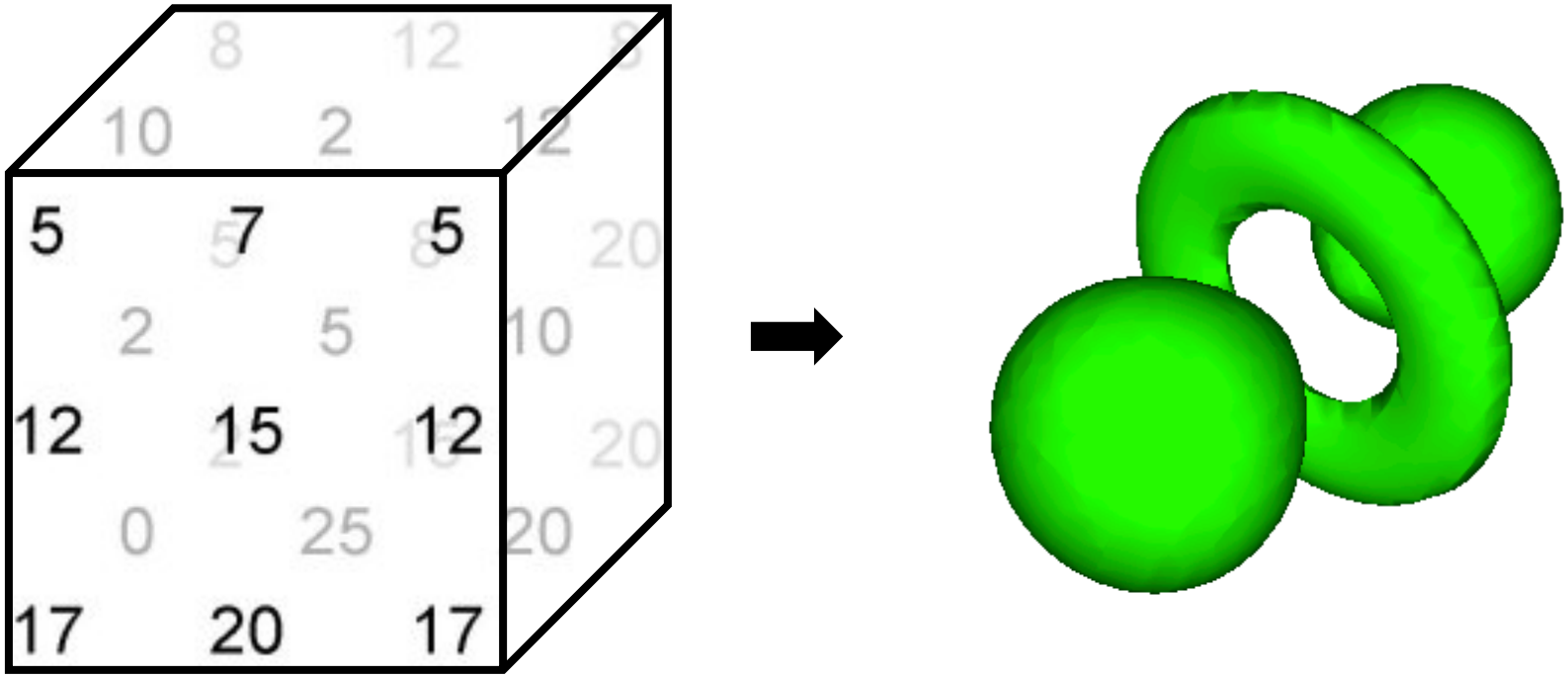
３次元データの可視化

- x,yとzの関数 $f(x,y,z)$
- 数値データ



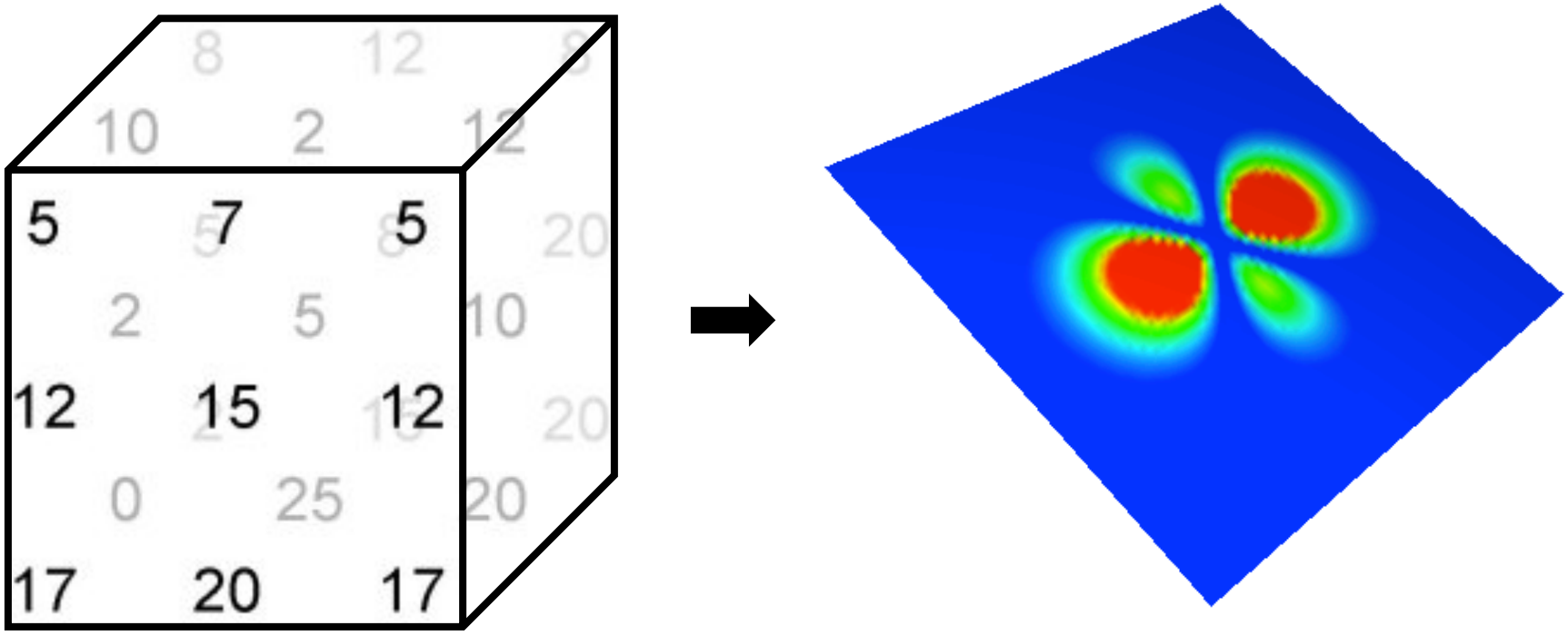
3次元データの可視化

- 等値面
  - Marching Cubes



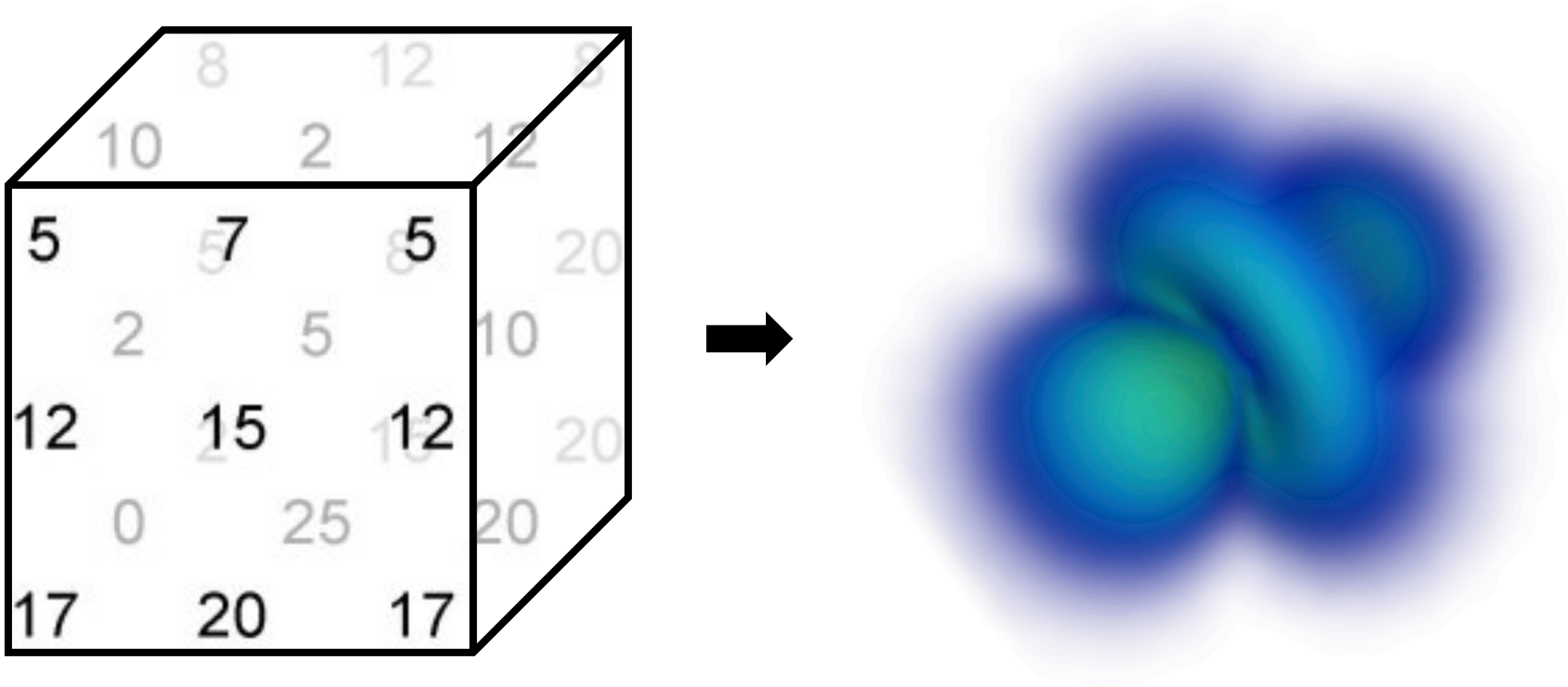
3次元データの可視化

- 断面
  - Slice Plane



３次元データの可視化

- ボリュームレンダリング
  - Direct Volume Rendering



可視化ソフトウェア

- 様々な可視化アルゴリズムを実装した便利なソフトウェアが多数開発されている。
  - 市販可視化ソフト
    - IDL, AVS/Express, Tecplot, …
  - 無料可視化ソフト
    - ParaView, VisIt, Amira, Vapor, …
  - 数式処理ソフトの可視化機能
    - Mathematica, MATLAB, …
  - 基本ライブラリ
    - VTK, Visualization Library, …

gnuplot入門

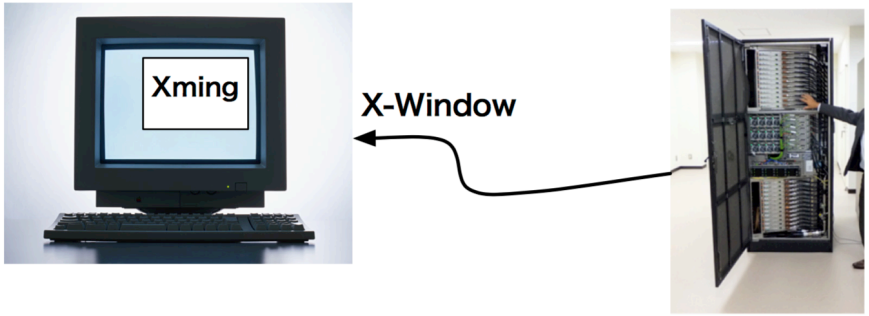
- この演習では、gnuplotを利用する。
  - http://www.gnuplot.info
  - 「ニュープロット」しばしば「グニュープロット」
  - GNUプロジェクトとは無関係

gnuplotとは

- gnuplot FAQより
  http://www.gnuplot.info/faq/faq.html
  - a command-driven interactive plotting program
  - both 2- and 3-dimensional plots
  - designed primarily for the visual display of scientific data
  - gnuplot is copyrighted, but freely distributable
  - you don’t have to pay for it
  - gnuplot is neither written nor maintained by the FSF

演習室の環境設定

- π-computer（のログインノード）にインストールされているgnuplotを使う。
- グラフは（Unixの）X-Windowシステム（X11）
- 端末の（マイクロソフトの）Winodwsシステムで、X11のクライアントを立ち上げる。
- デフォルトでは外部のX11アプリケーションを拒否する設定なので、それを変更する必要がある。



演習室での設定手順

- 各端末で
  1. すべてのプログラム→Xming→Xming
     ※特に何も起きない。
  2. Tera termを立ち上げる
     1. 「キャンセル」
     2. 「設定」
     3. 「SSH転送」
     4. リモートの（X）アプリケーションを…にチェックが入っていなければチェック
     5. ファイル→「新しい接続」→ログイン

参考：Unix系システムからの設定手順

1. X11が使えるようにする。（普通は何もする必要はない）
2. Macでは、OS X 10.7 Lion以降、X11が標準ではなくなったため、XQuartz.appをインストールする必要がある。
3. ターミナルから以下のコマンドを実行する。

```
$ ssh -X my id@pi.ircpi.kobeu.ac.jp
```

gnuplotの立ち上げ

- 上記の手順でX11アプリケーションの「貼り付け」を許可した上で
- （π-computer上で）gnuplotと打つ。

```
$ gnuplot
```

確認

- 以下のコマンドプロンプトが出ればgnuplotの立ち上げ成功。

```
gnuplot>
```

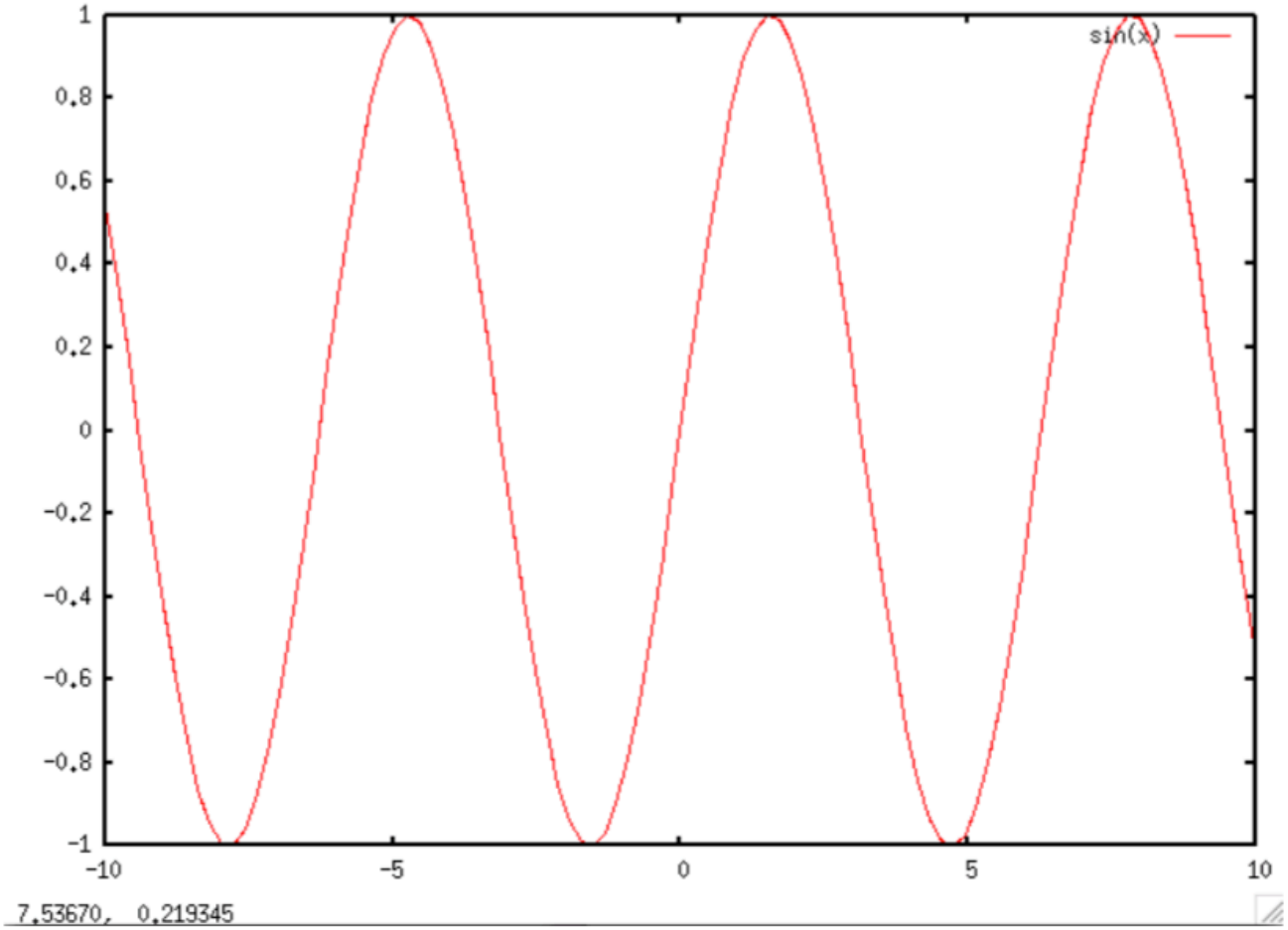
- ここで

```
gnuplot> plot sin(x)
```

と入れてみよう。

確認

- 以下のようなグラフが表示されれば成功。



gnuplotのヘルプと終了方法

- ヘルプ

```
gnuplot> help
```

- 終了

```
gnuplot> quit
```

gnuplotの単項演算子

- gnuplot 4.4のマニュアルより引用

単項演算子		
記号	例	説明
-	-a	マイナス符号
+	+a	プラス符号 (何もしない)
~	~a	* 1 の補数 (ビット反転)
!	!a	* 論理的否定
!	a!	* 階乗
$	$3	* ‘using‘ 内での引数/列指定

gnuplotの二項演算子

- gnuplot 4.4のマニュアルより引用

二項演算子		
記号	例	説明
**	a**b	累乗
*	a*b	積
/	a/b	商
%	a%b	* 余り
+	a+b	和
-	a-b	差
==	a==b	等しい
!=	a!=b	等しくない
<	a<b	より小さい
<=	a<=b	以下
>	a>b	より大きい
>=	a>=b	以上

gnuplotの二項演算子

- gnuplot 4.4のマニュアルより引用

&	a&b	* ビット積 (AND)
^	a^b	* ビット排他的論理和 (XOR)
|	a|b	* ビット和 (OR)
&&	a&&b	* 論理的 AND
||	a||b	* 論理的 OR
=	a = b	代入
,	(a,b)	累次評価
.	A.B	文字列の連結
eq	A eq B	文字列が等しい
ne	A ne B	文字列が等しくない

gnuplotの組み込み関数

- gnuplot 4.4のマニュアルより引用

数学ライブラリ関数		
関数	引数	戻り値
abs(x)	任意	x の絶対値, $\|x\|$; 同じ型
abs(x)	複素数	x の長さ, $\sqrt{\mathrm{real}(x)^2 + \mathrm{imag}(x)^2}$
acos(x)	任意	$\cos^{-1} x$ (アークコサイン)
acosh(x)	任意	ラジアンでの $\cosh^{-1} x$ (逆双曲余弦)
arg(x)	複素数	x の偏角
asin(x)	任意	$\sin^{-1} x$ (アークサイン)
asinh(x)	任意	ラジアンでの $\sinh^{-1} x$ (逆双曲正弦)
atan(x)	任意	$\tan^{-1} x$ (アークタンジェント)
atan2(y,x)	整数または実数	$\tan^{-1}(y/x)$ (アークタンジェント)
atanh(x)	任意	ラジアンでの $\tanh^{-1} x$ (逆双曲正接)
EllipticK(k)	実数 k $\in$ (-1:1)	$K(k)$ 第 1 種完全楕円積分
EllipticE(k)	実数 k $\in$ [-1:1]	$E(k)$ 第 2 種完全楕円積分
EllipticPi(n,k)	実数 n<1, 実数 k $\in$ (-1:1)	$\Pi(n,k)$ 第 3 種完全楕円積分

gnuplotの組み込み関数

- gnuplot 4.4のマニュアルより引用

besj0(x)	整数または実数	j_0 ベッセル関数 (0 次ベッセル関数)
besj1(x)	整数または実数	j_1 ベッセル関数 (1 次ベッセル関数)
besy0(x)	整数または実数	y_0 ベッセル関数 (0 次ノイマン関数)
besy1(x)	整数または実数	y_1 ベッセル関数 (1 次ノイマン関数)
ceil(x)	任意	$\lceil x \rceil$, x (の実部) 以上の最小の整数
cos(x)	任意	x のコサイン $\cos x$
cosh(x)	任意	$\cosh x$, x のハイパボリックコサイン
erf(x)	任意	erf(real(x)), x の 実部の誤差関数
erfc(x)	任意	erfc(real(x)), 1.0 - (x の実部の誤差関数)
exp(x)	任意	e^x, x の指数関数
floor(x)	任意	$\lfloor x \rfloor$, x (の実部) 以下の最大の整数
gamma(x)	任意	gamma(real(x)), x の実部のガンマ関数
ibeta(p,q,x)	任意	ibeta(real(p, q, x)), p,q,x の実部の不完全ベータ関数
inverf(x)	任意	x の実部の逆誤差関数
igamma(a,x)	任意	igamma(real(a, x)), a,x の実部の不完全ガンマ関数
imag(x)	複素数	x の虚数部分 (実数)
invnorm(x)	任意	x の実部の逆正規分布関数

gnuplotの組み込み関数

- gnuplot 4.4のマニュアルより引用

int(x)	実数	x の整数部分 (0 に向かって丸め)
lambertw(x)	実数	Lambert W 関数
lgamma(x)	任意	lgamma(real(x)), x の実部のガンマ対数関数
log(x)	任意	$\log_e x$, x の自然対数 (底 e)
log10(x)	任意	$\log_{10} x$, x の対数 (底 10)
norm(x)	任意	x の実部の正規分布 (ガウス分布) 関数
rand(x)	任意	rand(real(x)), 疑似乱数生成器
real(x)	任意	x の実部
sgn(x)	任意	$x > 0$ なら 1, $x < 0$ なら -1, $x = 0$ なら 0. x の虚部は無視
sin(x)	任意	$\sin x$, x のサイン
sinh(x)	任意	$\sinh x$, x のハイパボリックサイン
sqrt(x)	任意	$\sqrt{x}$, x の平方根
tan(x)	任意	$\tan x$, x のタンジェント
tanh(x)	任意	$\tanh x$, x のハイパボリックタンジェント

演習1

- 次のグラフを描け。

$$f(x) = x^x$$

演習1

- 解答

```
gnuplot> plot x**x title "x¥^x"
```

複数のグラフ

- カンマで区切る。

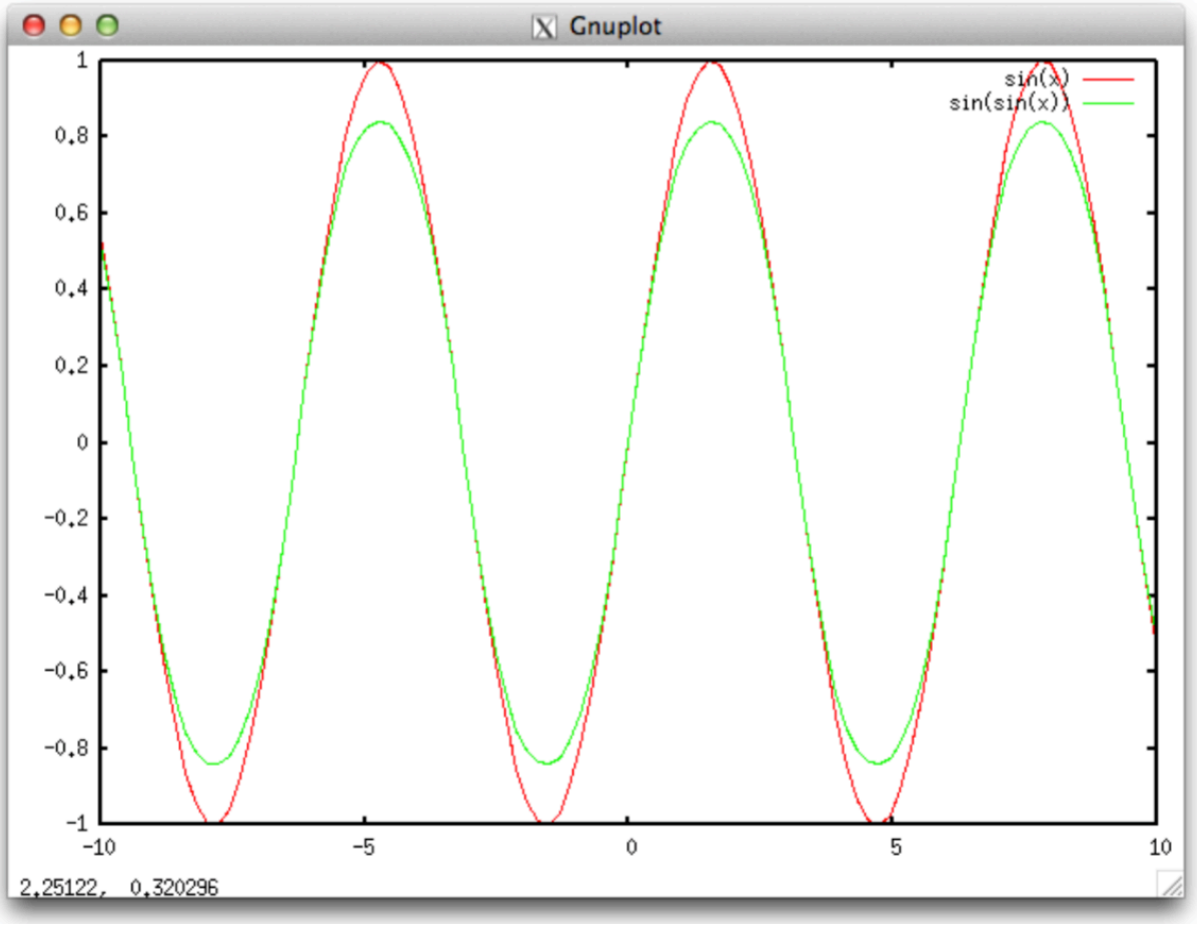
```
gnuplot> plot sin(x), sin(sin(x))
```

- グラフを区別する。

```
gnuplot> plot sin(x) title "sin(x)", sin(sin(x)) title
"sin(sin(x))"
```

複数のグラフ

- 結果


Gnuplot
sin(x)
sin(sin(x))
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
-10
-5
0
5
10
2.25122, 0.320296

様々なパラメータ

- setコマンド

```
gnuplot>  set title "y=x^x"
gnuplot>  set xlabel "x (no units)"
gnuplot>  set ylabel "y (no units)"
gnuplot>  plot x**x
```

様々なパラメータ

- 定義域と値域

```
gnuplot>  set xrange [0:5]
gnuplot>  replot
```

様々なパラメータ

- グリッド表示

```
gnuplot>  set grid
gnuplot>  replot
```

様々なパラメータ

- 関数の定義

```
gnuplot>  s2(x) = sin(sin(x))
gnuplot>  s4(x) = s2(s2(x))
gnuplot>  s10(x) = s4(s4(s2(x)))
gnuplot>  plot s10(x)
```

データのファイルからの読み込み

- gnuplot には、ファイルに書き込まれた離散データを読み込み、それをグ ラフにする機能がある。

グレゴリー・ライプニッツ級数

$$\pi = 4\left(\frac{1}{1} - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \frac{1}{9} - \frac{1}{11} + \cdots\right)$$

- 第 n 項までの級数がどの程度 π に近いかみるプログラム
  - leibniz.f95

```
1    4.0000000000000000
2    2.6666666666666670
3    3.4666666666666668
4    2.8952380952380956
5    3.3396825396825403
.              .
.              .
```

準備

- 作業ディレクトリの作成

```
$ cd                    ホームディレクトリに移動
$ mkdir vis01           作業ディレクトリの作成（名前は何でもOK）
```

- サンプルコードをコピー

```
$ cd vis01              作業ディレクトリに移動
$ cp /tmp/160428/*  .   サンプルコード（2つ）をコピー
```

演習2

- データ作成
  1. leibniz.f95をgfortranコンパイラでコンパイルし、実行せよ。

```
$ gfortran  leibniz.f95
./a.out                          100行の長い出力
./a.out  | head                  あるいは more/less/tail コマンド
/a.out  > test.data
```

  2. ファイルtest.dataの中身を確認せよ。

     ※エディタで開くよりもmore / less / head / tailコマンドで見る方が早い。

```
$ less  test.data
```

space: scroll forward, **b**: scroll backward, **q**: quit

演習2

- データファイルの中身

```
# sample data generated by leibniz.f95
#       term    sum
#
              1    4.000000000000000
              2    2.666666666666670
              3    3.466666666666668
              4    2.895238095238095 6
              5    3.339682539682540 3
              6    2.976046176046176 5
              7    3.283738483738484 4
```

演習３

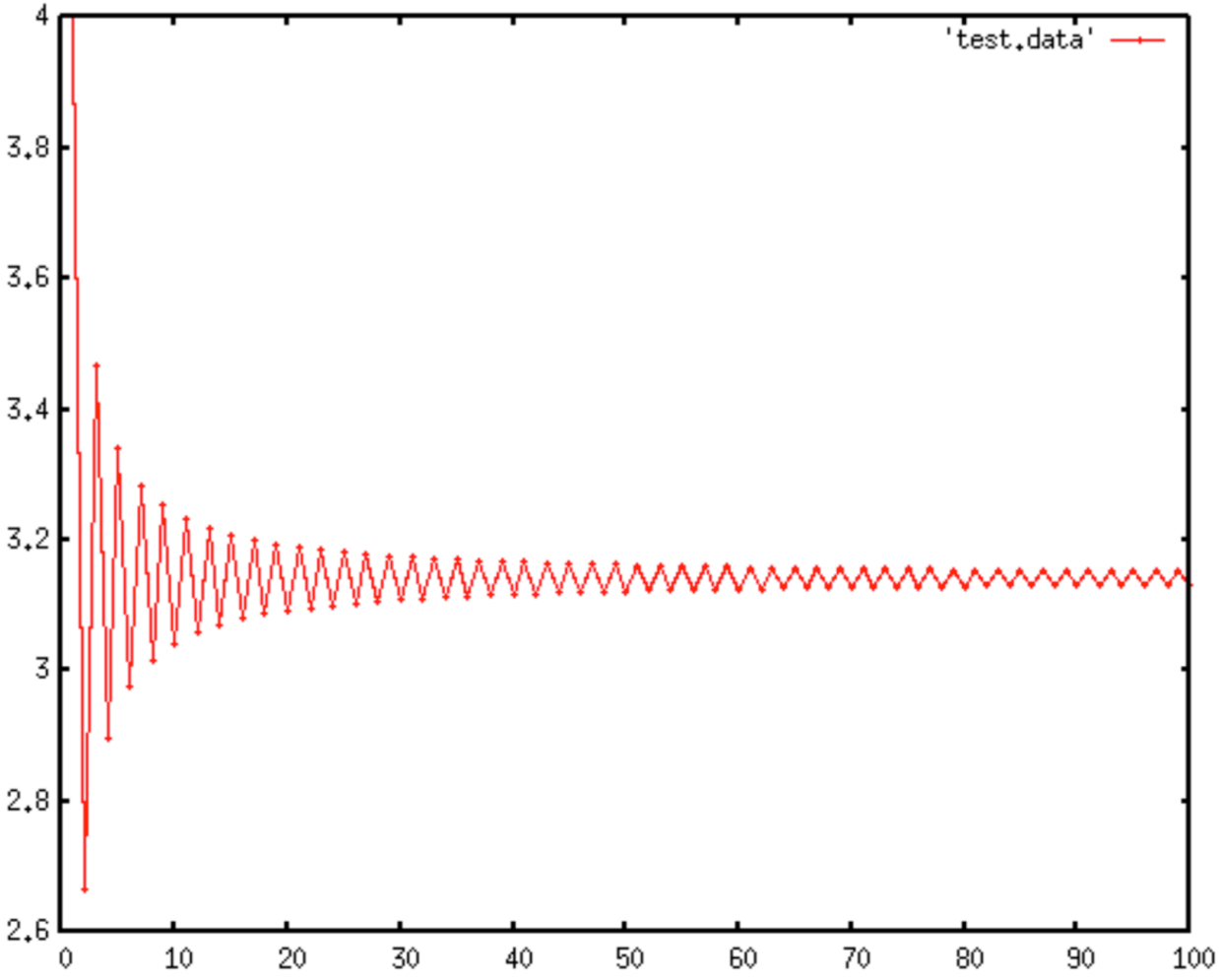
- １次元グラフ
  - gnuplotを立ち上げ、コマンドプロンプトに次のコマンドを入力せよ。

```
gnuplot>  plot  'test.data'  w lp
```

    - w lpは、with linespointsの略記法。
    - linespointsは、線（line）と点（point）を表示することを意味する。

演習3

- 出力例



gnuplotの入力ファイル

- #はコメント開始
- １行にx,y値のペア
- デフォルトでは第１列がplotのx座標、第２列がy座標（変更可能）

演習４

- オプションの変更
  1. ラベルの文字を消す。

```
gnuplot> unset key
gnuplot> replot
```

  2. 縦軸の表示範囲を調整する。

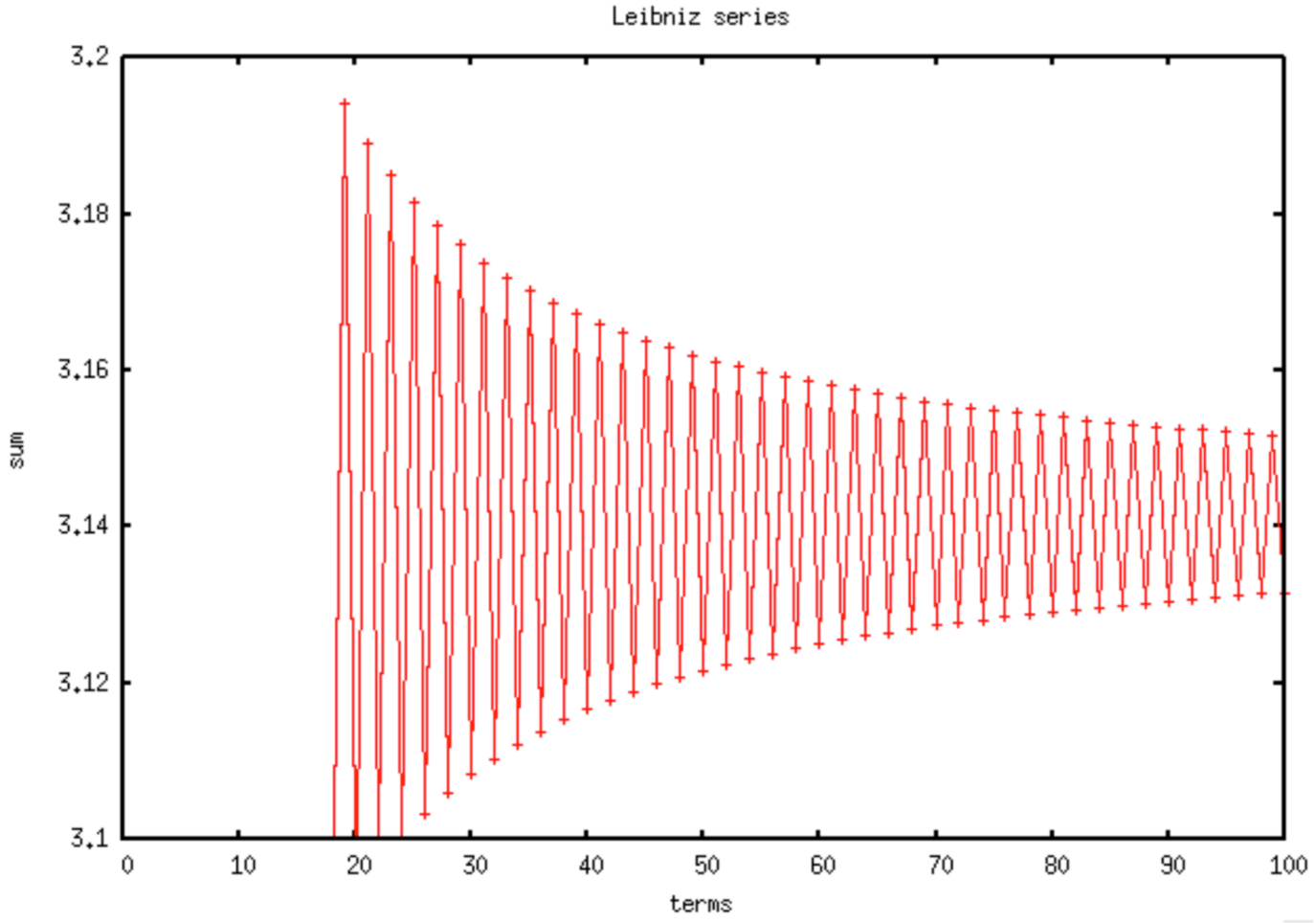
```
gnuplot> set yrange [3.1:3.2]
gnuplot> replot
```

  3. 図全体のタイトルと、x軸、y軸の説明を入れる。

```
gnuplot> set title "Leibniz series"
gnuplot> set xlabel "terms"
gnuplot> set ylabel "sum"
gnuplot> replot
```

演習4

- 出力例


Leibniz series
3.2
3.18
3.16
3.14
3.12
3.1
sum
0
10
20
30
40
50
60
70
80
90
100
terms

gnuplotスクリプト

- gnuplot ではコマンドプロンプトに手で入力する内容をファイルから読み込ませることが出来る。
  - ファイル名 : leibnitz.gp（拡張子は任意）

```
#
# leibniz.gp
#
set yrange [3.1:3.2]
set xlabel "terms”
set ylabel "sum”
plot "test.data"  w lp
pause -1
```

最後の pause -1 は(一瞬だけ)表示してすぐに終了してしまうのを防ぐため。

スクリプトの実行

- gnuplotがまだ立ち上がっていたらquitコマンドで終了し、シェルから以下のコマンドを実行せよ。

```
$ gnuplot  leibnitz.gp
```

課題

- test.data のデータに重ねて、$y = \pi$ の直線も描くような gnuplot スクリプトファイルを作り、そのファイル名を leibniz2.gp とせよ。
  - ヒント：gnuplot では pi という変数に π が入っている。定数グラフは plot pi で描ける。
- leibnitz.gpとleibnitz2.gpの差分をメールで提出せよ。

```
$ diff leibnitz.gp leibnitz2.gp | mail -s your_account_name
kobeuniv.compra1@gmail.com
```

締切：次回授業日の前日23:59まで

時間が余った人は…

- leibniz.f95を改訂して200項まで和をとるプログラムにせよ。
- そのプログラムを実行し、どの程度πに近づくかgnuplotで見よ。