

情報とコンピュータ

静岡大学工学部

安藤和敏

2004.11.01

3章 数値計算と関数の学習

- 数値計算を試みよう
- 単純な計算
- 関数
- ループの作成と関数の学習
- 最適値の探索
- 情報を配列に格納する
- 総和, 極小, 極大を求める
- プログラミングのパターン

数値計算をしてみよう(1)

20歳の青年が60歳までに1億円貯めるには、毎月いくら貯金しなければならないか？

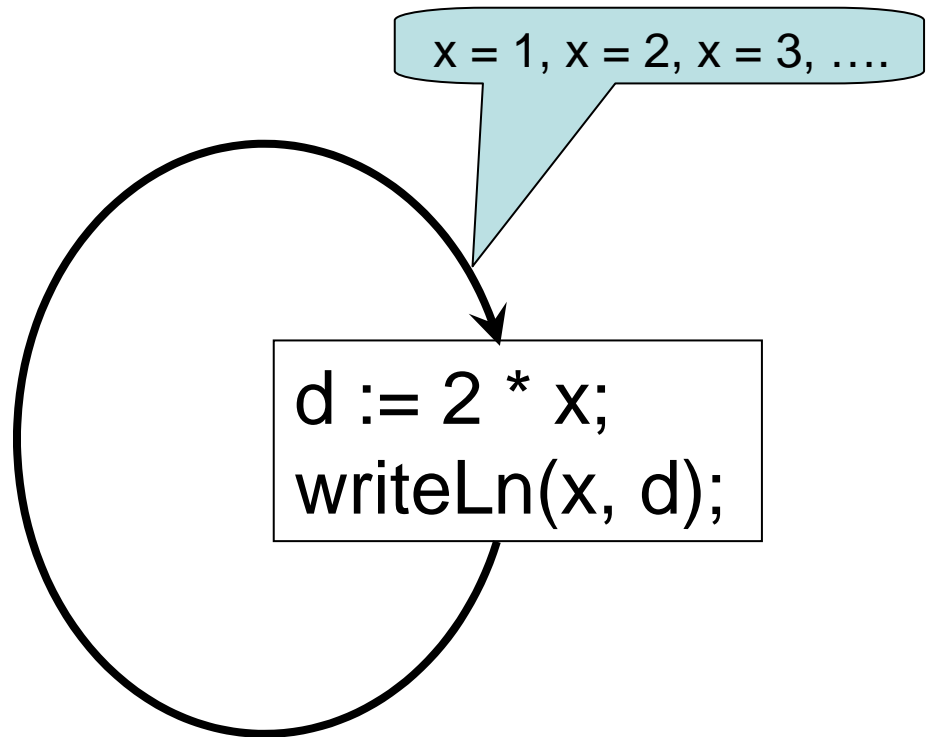
数値計算をしてみよう(2)

1000m²の錫(すず)の板で円柱をつくる
とき、その円柱の体積が最大になるの
は正確にどの寸法のとときか？

最適化: ある状況におけるパラメータの最適値
を求める作業.

2倍関数 ($d(x) = 2x$) の計算

```
x := 1;  
d := 2 * x;  
writeln(x, d);  
x := 2;  
d := 2 * x;  
writeln(x, d);  
x := 3;  
d := 2 * x;  
writeln(x, d);  
...
```



2倍関数の計算(ループの例)

```
var
  x, d: real;
begin
  x := 1;
  while true do
    begin
      d := 2 * x;
      writeln(x, d);
      x := x + 1;
    end;
  end.
```

d	6.0
x	4.0

while 構文

Pascal の while文 は例えば, 以下のような書式を取る.

```
while 条件式 do  
  複合文;
```

条件式とは, `answer = 'yes'` や `x <= 20` などのような式で, 真 (true) か偽 (false) のどちらかの値をとる.

複合文とは, `begin`と`end`で挟まれた文の並びである. ということは前に教え.

2倍関数の計算のプログラム

```
program Double(input, output);  
var  
  d, x: real;  
begin  
  x := 1.0;  
  while x <= 10.0 do  
  begin  
    d := 2.0*x;  
    writeln(x:6:2,d:6:2);  
    x := x + 1.0;  
  end;  
end.
```

d	20.0
x	11.0

円柱の体積の計算

を円周率, r を円柱の底面の半径, h を円柱の高さとすると, この円柱の体積 V は

$$V = \pi r^2 h$$

一方で, この円柱の表面積 A は,

$$A = 2\pi r^2 + 2\pi rh = 1000$$

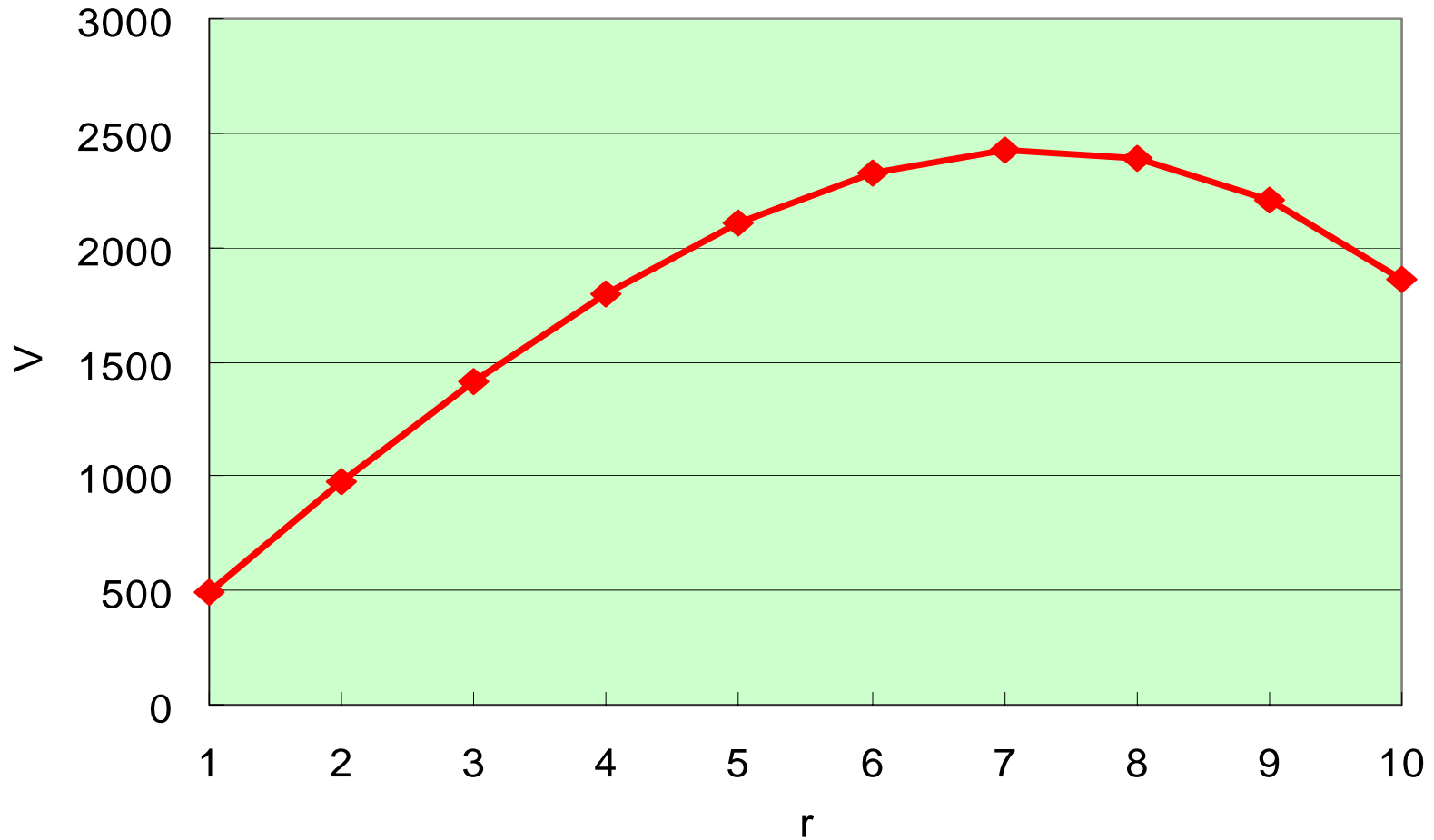
だから,
$$\pi rh = 500 - \pi r^2$$

$$\therefore V = 500r - \pi r^3$$

円柱の体積の計算のプログラム

```
program CyllinderVolumes(input, output);
var
  r, V: real;
begin
  r := 1.0;
  while r <= 10.0 do
    begin
      V := 500*r - 3.14159*r*r*r;
      writeLn(r:8:2,V:8:2);
      r := r + 1.0;
    end;
  end.
```

Vのグラフ



金利計算の公式

金利は少数で表されているとする。(12% = 0.12)

今月末の預金残高
= 前月の預金残高 + (前月の預金算残高*金利)
+ 今月の積立額

savings を預金残高, monthint を金利とすると,
Pascal の計算は,

```
savings := savings + (savings*monthint)
+ payment;
```

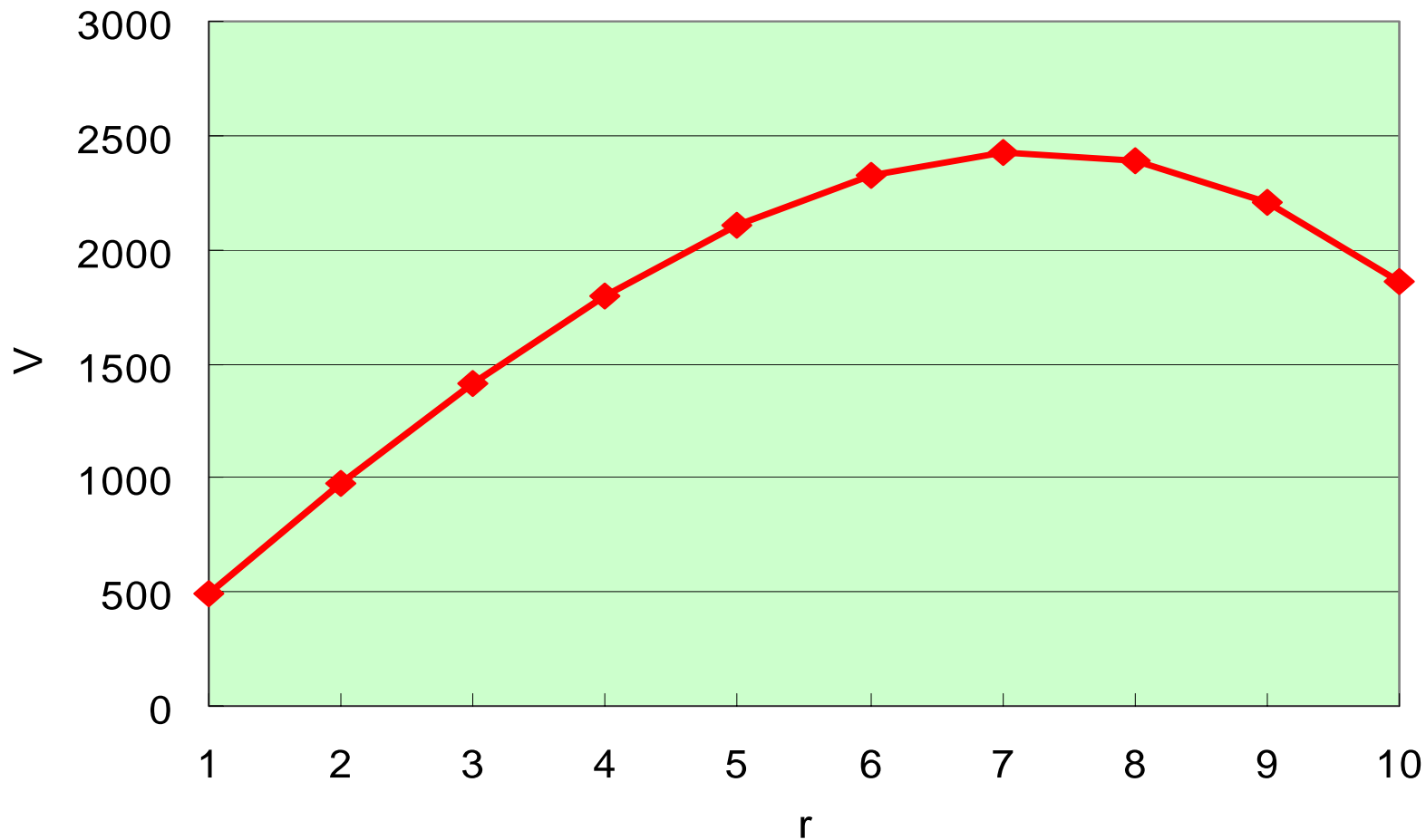
預金残高計算のアルゴリズム

1. 指定された毎月の積立額 (payment) を読み込む .
2. $\text{savings} = 0$, $\text{monthint} = 0.01$ (年率12%) と設定する . . .
3. 480ヶ月の各月の残高を
 $\text{savings} := \text{savings} + (\text{savings} * \text{monthint}) + \text{payment};$
で計算する .
4. 40年後の残高を出力する .

40年後の預金計算のプログラム

```
program Savings40Years(input, output);
var
  payment, savings, monthint, month: real;
begin
  writeln('毎月いくらずつ積み立てますか？(単位:万円)');
  readln(payment);
  savings := 0; monthint := 0.001; month := 1;
  while month <= 480 do
    begin
      savings := savings + (savings*monthint) + payment;
      month := month + 1;
    end;
  writeln('40年後の預金残高は , ', savings:10:2, '万円です . ');
end.
```

最適値の探索(円柱の体積)



円柱の体積

$$r = 6 \text{ のとき, } V = 500 \times 6 - 3.14159 \times 6^3 = 2321.42$$

$$r = 6.01 \text{ のとき, } V = 500 \times 6.01 - 3.14159 \times 6.01^3 \\ = 2323.02$$

$$r = 6.02 \text{ のとき, } V = 500 \times 6.02 - 3.14159 \times 6.02^3 \\ = 2324.61$$

⋮

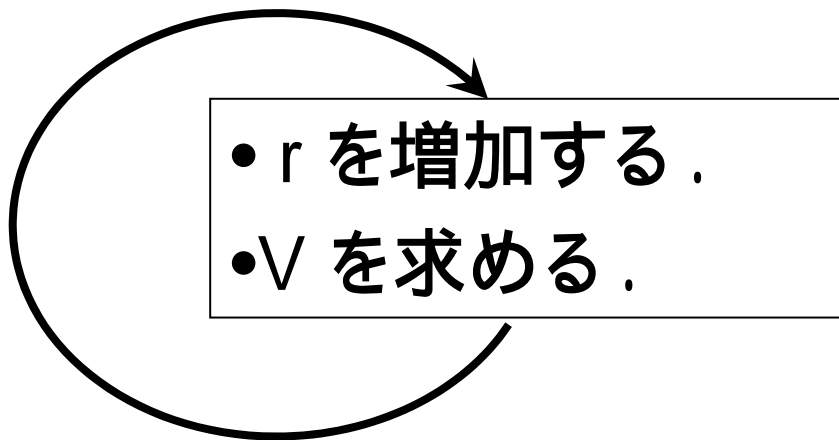
r を繰り返し増やしていき, V が減少したところで停止すればよい.

最適値の探索の方法

- r を適当な初期値に設定する .
- r をどのくらいいつつ増加するかを決定する .
- V を求める .
- r を増加する .
- V を求める .
- r を増加する .
- ...
- V が小さくなった時点で停止する .
- そのときの V が最適値である .

最適値の探索の方法

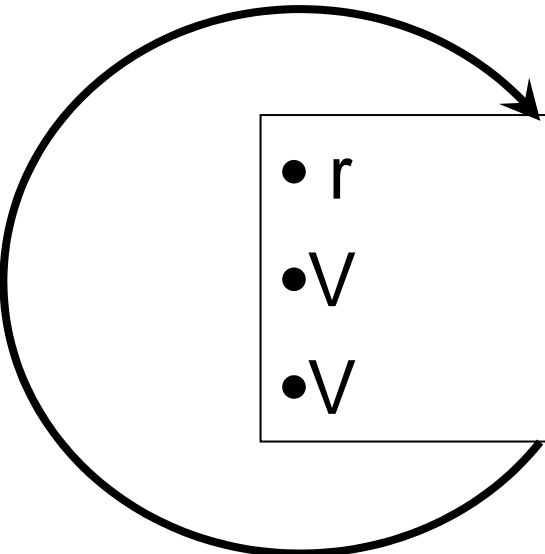
- r を適当な初期値に設定する.
- r をどのくらいずつ増加するかを決定する.
- V を求める.



- V が小さくなった時点で停止する.
- その直前の V が最適値である.

最適値の探索の方法

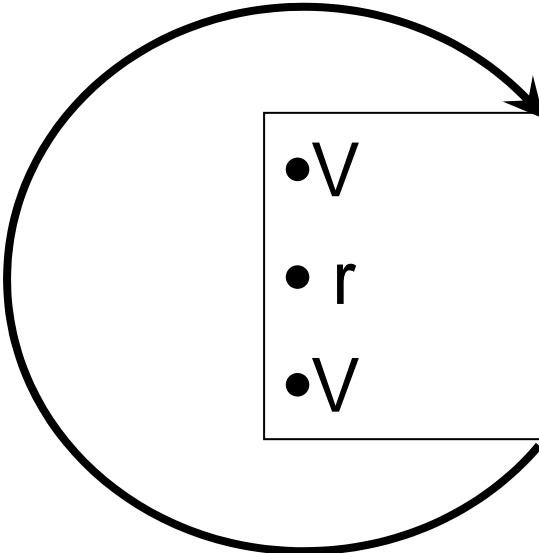
- r を適当な初期値に設定する.
- r をどのくらいずつ増加するかを決定する.
- V を求める.

- 
- r を増加する.
 - V を求める.
 - V が直前の V より小さければ停止する.

- 直前の V が最適値である.

最適値の探索のアルゴリズム(1)

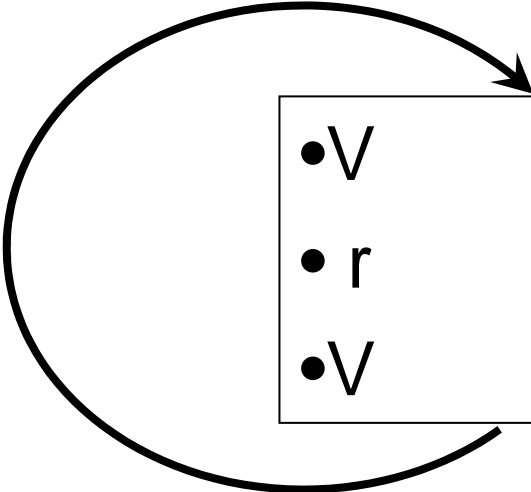
- r を適当な初期値に設定する.
- r をどのくらいずつ増加するかを決定する.
- V を求める.

- 
- V が直前の V より小さければ停止する.
 - r を増加する.
 - V を求める.

- 直前の V が最適値である.

最適値の探索のアルゴリズム(2)

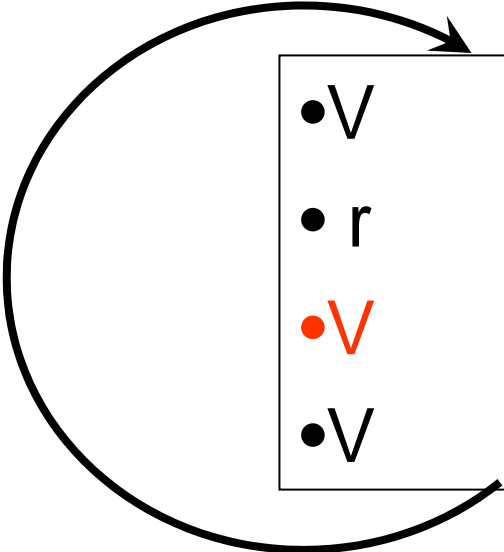
- r を適当な初期値に設定する.
- r をどのくらいずつ増加するかを決定する.
- V を求める.
- 直前の V を 0 とする.

- 
- V が直前の V より小さければ停止する.
 - r を増加する.
 - V を求める.

- 直前の V が最適値である.

最適値の探索のアルゴリズム (3)

- r を適当な初期値に設定する.
- r をどのくらいずつ増加するかを決定する.
- V を求める.
- 直前の V を 0 とする.

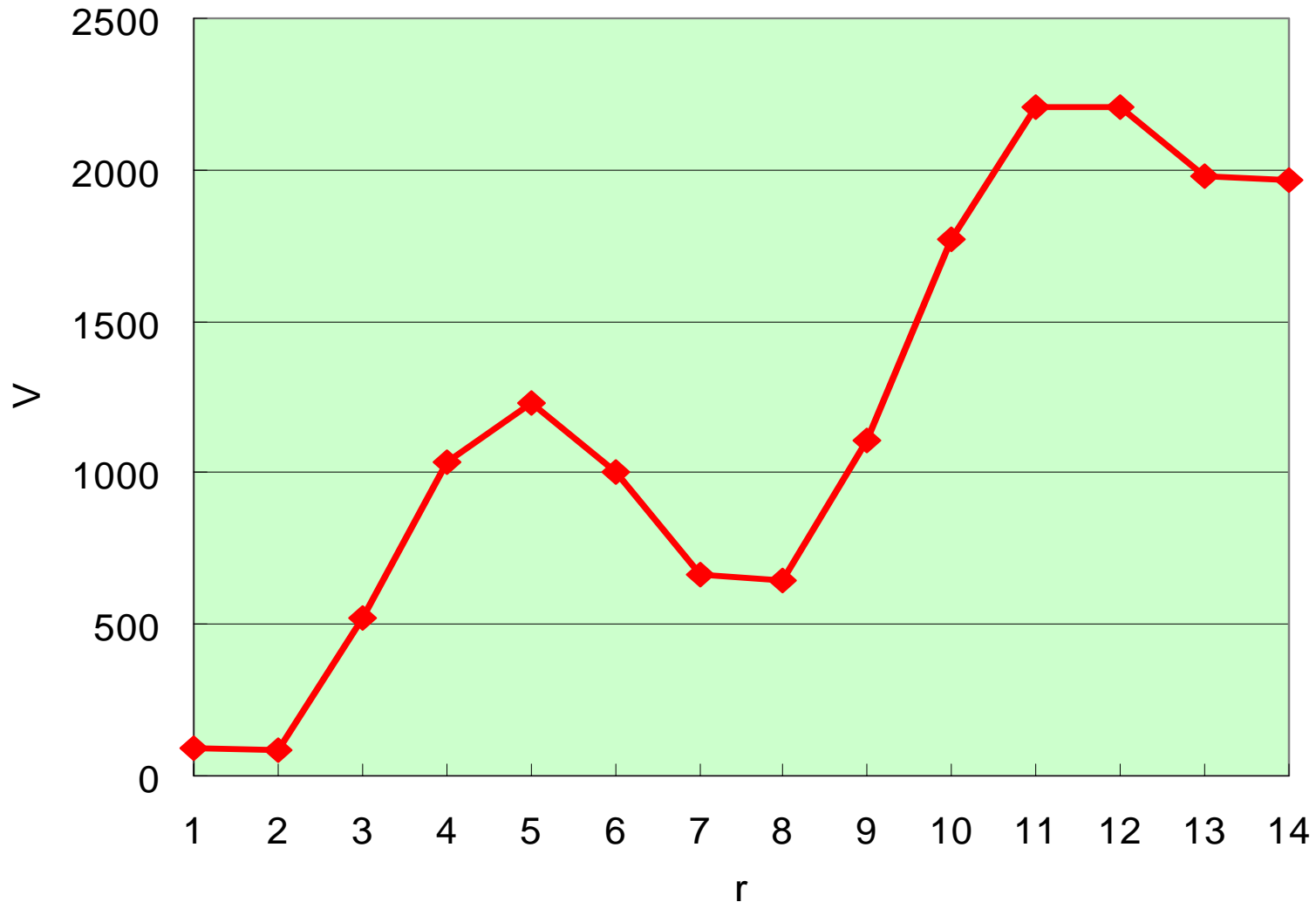
- 
- V が直前の V よりければ停止する.
 - r を増加する.
 - V の値を「直前の V 」に代入する.
 - V を求める.

- 直前の V が最適値である.

円柱体積の最適値探索のプログラム

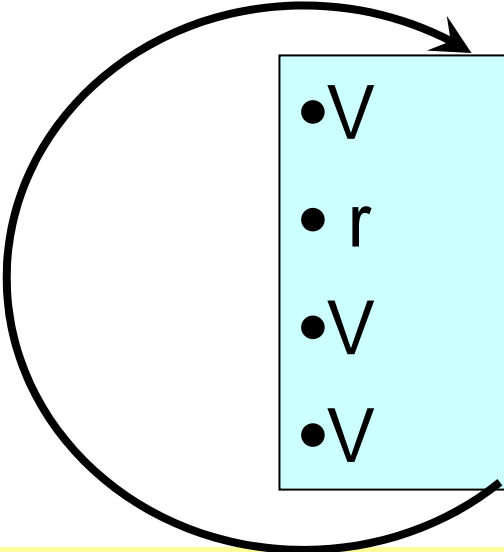
```
program FindBest(input, output);
var
  r, V, previousV, increase: real;
begin
  writeLn('r の初期値はいくつですか?');
  readLn(r);
  writeLn('毎回 r をいくつずつ増やしますか?');
  readLn(increase);
  previousV := 0;
  V := 500 * r - 3.14159 * r * r * r;
  while V >= previousV do
    begin
      r := r + increase;
      previousV := V;
      V := 500 * r - 3.14159 * r * r * r;
      writeLn(r:10:2,V:10:2);
    end;
  writeLn('V の最大値は , ', previousV:10:2);
end.
```

Vが以下のような関数だったら？



最適値の探索のアルゴリズム

- r を適当な初期値に設定する.
- r をどのくらいずつ増加するかを決定する.
- V を求める.
- 直前の V を 0 とする.

- 
- V が直前の V よりければ停止する.
 - r を増加する.
 - V の値を「直前の V 」に代入する.
 - V を求める.

- 直前の V が最適値である.

最適値の探索のアルゴリズム

- (計算のための初期条件の設定).



項目が見つからない間は繰り返す

• 次の項目を得るための繰り返し作業

- 見つかった項目の出力.

1億貯めるための毎月の積立額を求めるアルゴリズム

- 最初の毎月の積立額を決定する。
- 各繰返して、積立額をどれだけ増加するかを決定。
- 最初の預金残高を0に設定する。

預金残高が1億未満の間は繰り返す

- 毎月の積立額を増加。
- Savings40years を使って、40年後の預金残高を計算。
- 毎月の預金残高と40年後の残高を出力。

プログラム MillionDollarAnswer

```
program MillionDollarAnswer(input, output);  
var  
  payment, increase, savings, monthint, month: real;  
begin  
  writeln('最初の積立額は何万円ですか?');  
  readln(payment);  
  writeln('毎回, 積立額は何万円ずつ増やしますか?');  
  readln(increase);  
  savings := 0;
```

プログラム MillionDollarAnswer (続き)

```
while savings < 10000 do
  begin
    payment := payment + increase;
    savings := 0;
    monthint := 0.01;
    month := 1;
    while month <= 480 do
      begin
        savings := savings + (savings*monthint) + payment;
        month := month +1;
      end;
    write('積立額 ', payment:10:2, '万円, ');
    writeLn('預金残高 ', savings:10:2, '万円');
  end;
writeLn(payment:10:2, '万円ずつ積み立てれば, 1億貯まります. ');
end.
```