「アルゴリズムとデータ構造入門」 第8回 資料 (*Nov. 30, 2004*, 奥乃)

http://winnie.kuis.kyoto-u.ac.jp/~okuno/Lecture/04/IntroAlgDS/

1 Supplements: Finding Fixed Points

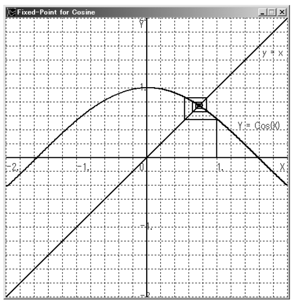

図 1. $x = \cos(x)$

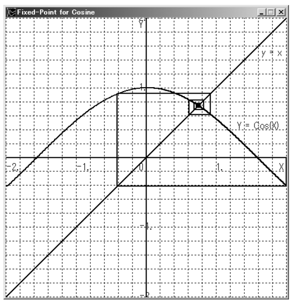

図 2. $x = \cos(x)$

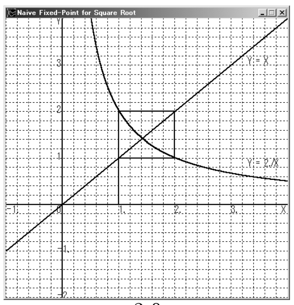

図 3. $x = \frac{2.0}{x}$ (無限ループ)

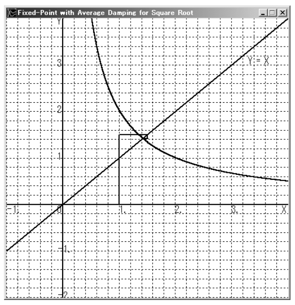

図 4. 平均緩和法による解

随意課題１ (期限: ２００５年２月１５日)

$\boxed{1}$ **不動点探索を行う** fixed-point **の途中経過を示す上記のようなグラフを書く** Scheme **プログラムを作成せよ.**

$\boxed{2}$ $x = \sin(x) + \cos(x)$ **を** fixed-point **で探索し, その探索過程を上記のようなグラフで示せ.**

$\boxed{3}$ Newton **法による平方根を計算を** fixed-point **で探索し, その探索過程を上記のようなグラフで示せ.**

2 Supplements: Procedures as Returned Values – 次の 2 式 (expression) は等価！

```
(define (fact n)
  (if (< n 1)
      1
      (* n (fact (- n 1))) ))
```

```
(define fact
  (lambda (n)
     (if (< n 1)
         1
         (* n (fact (- n 1))) )))
```

第２章のコード

2.1.1 Arithmetic Operations for Rational Numbers

```
(define (add-rat x y)
  (make-rat (+ (* (numer x) (denom y))
               (* (numer y) (denom x)) )
            (* (denom x) (denom y)) ))

(define (sub-rat x y)
  (make-rat (- (* (numer x) (denom y))
               (* (numer y) (denom x)) )
            (* (denom x) (denom y)) ))

(define (mul-rat x y)
  (make-rat (* (numer x) (numer y))
            (* (denom x) (denom y)) ))

(define (div-rat x y)
  (make-rat (* (numer x) (denom y))
            (* (denom x) (numer y)) ))
```

```
(define (equal-rat? x y)
  (= (* (numer x) (denom y))
     (* (numer y) (denom x)) ))
```

【Constructor, Selectors, Printer】

```
(define (make-rat n d) (cons n d))

(define (numer x) (car x))

(define (denom x) (cdr x))

(define (print-rat x)
  (newline)
  (display (numer x))
  (display "/")
  (display (denom x)) )
```

2.1.2 When is reduction performed?

```
(define (make-rat n d)
  (let ((g (gcd n d)))
    (cons (/ n g) (/ d g)) ))

(define (numer x) (car x))

(define (denom x) (cdr x))
```

```
(define (numer x)
  (let ((g (gcd (car x) (cdr x))))
     (/ (car x) g) ))

(define (denom x)
  (let ((g (gcd (car x) (cdr x))))
     (/ (cdr x) g) ))
```

2.1.3 When is meant by Data?

```
(define (cons x y)
  (define (dispatch m)
    (cond ((= m 0) x)
          ((= m 1) y)
          (else (error "Argument not 0 or 1 -- CONS" m)) ))
  dispatch )
(define (car z) (z 0))
(define (cdr z) (z 1))
```

```
(lambda (m) (cond ((= m 0) x)
                  ((= m 1) y)
                  (else (error "...")) ))
```

2.1.4 Interval Arithmetic (自習と宿題)

```
(define (add-interval x y)
  (make-interval (+ (lower-bound x) (lower-bound y))
                 (+ (uppper-bound x) (uppper-bound y)) ))

(define (mul-interval x y)
  (let ((p1 (* (lower-bound x) (lower-bound y)))
        (p2 (* (lower-bound x) (upper-bound y)))
        (p3 (* (upper-bound x) (lower-bound y)))
        (p2 (* (upper-bound x) (upper-bound y))) )
    (make-interval (min p1 p2 p3 p4)
                   (max p1 p2 p3 p4) )))
```

```
(define (make-interval a b) (cons a b))
(define (lower-bound x) (car x))
(define (upper-bound x) (cdr x))
```

```
(define (div-interval x y)
  (mul-interval x
                (make-interval (/ 1.0 (upper-bound y))
                               (/ 1.0 (lower-bound y)) )))

(define (sub-interval x y)
  (add-interval x
                (make-interval (- (upper-bound x))
                               (- (lower-bound x)) )))

(define (sub-interval x y)
  (make-interval (- (lower-bound x) (upper-bound x))
        (- (upper-bound x) (lower-bound x)) ))
```

3 宿題 – 〆切は従来通り: １２月６日 (月) 午後５時 事務室レポート箱

$\boxed{1}$ Exercise 2.1 〜 2.22

$\boxed{2}$ プログラムは動くことを確認すること.