課題1

- 3x3行列Aと3x1行列bを用いて定義される連立一次方程式
  Ax = b
  を解くヤコビ法によるプログラムを書け．収束は残差ベクトルの大きさによって判定せよ．すなわち，|Ax - b| < epsとなるとき，反復をやめて答えを返すようにせよ．
- 解答はkadai1(A, b, eps)という形の関数で提出すること．ただし，A = [[1, 2, 4], [4, 2, 3], [5, 2, 3]], b = [3, 2, 4] のように表現せよ．
- 解に収束しない場合も考えて，一定回数以上実行したら即打ち切って答えを返すようにせよ．

解答例１（1）

- 授業で示したプログラムの形を整えて，作れば良い．

```
function kadai1(A, b, eps){
    function res(x){
        var ssum = 0.0
        for (var i = 0; i < 3; i++){
            var r = A[i]
            var sum = b[i]
            for (var j = 0; j < 3; j++)
                sum -= r[j] * x[j]
            ssum += sum * sum
        }
        return ssum
    }
```

答えにどれだけ近いかを調べる関数res()

```
    var N = 1000
    var x = [0.0, 0.0, 0.0]
    for (var k = 0; k < N; k++){
        if (res(x) < eps) return x
        var y = []
        for (var i = 0; i < 3; i++){
            var sum = b[i]
            var r = A[i]
            for (var j = 0; j < 3; j++)
                if (i !== j) sum -= r[j] * x[j]
            y.push(sum / r[i])
        }
        x = y
    }
    return null
}
```

Jacobi法のプログラム

最大くらい返し回数はN

解答例１ (2)

- 実行例

$$\begin{pmatrix} 5 & 2 & 1 \\ 2 & 5 & 1 \\ 2 & 2 & 9 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} 3 \\ 2 \\ 1 \end{pmatrix}$$

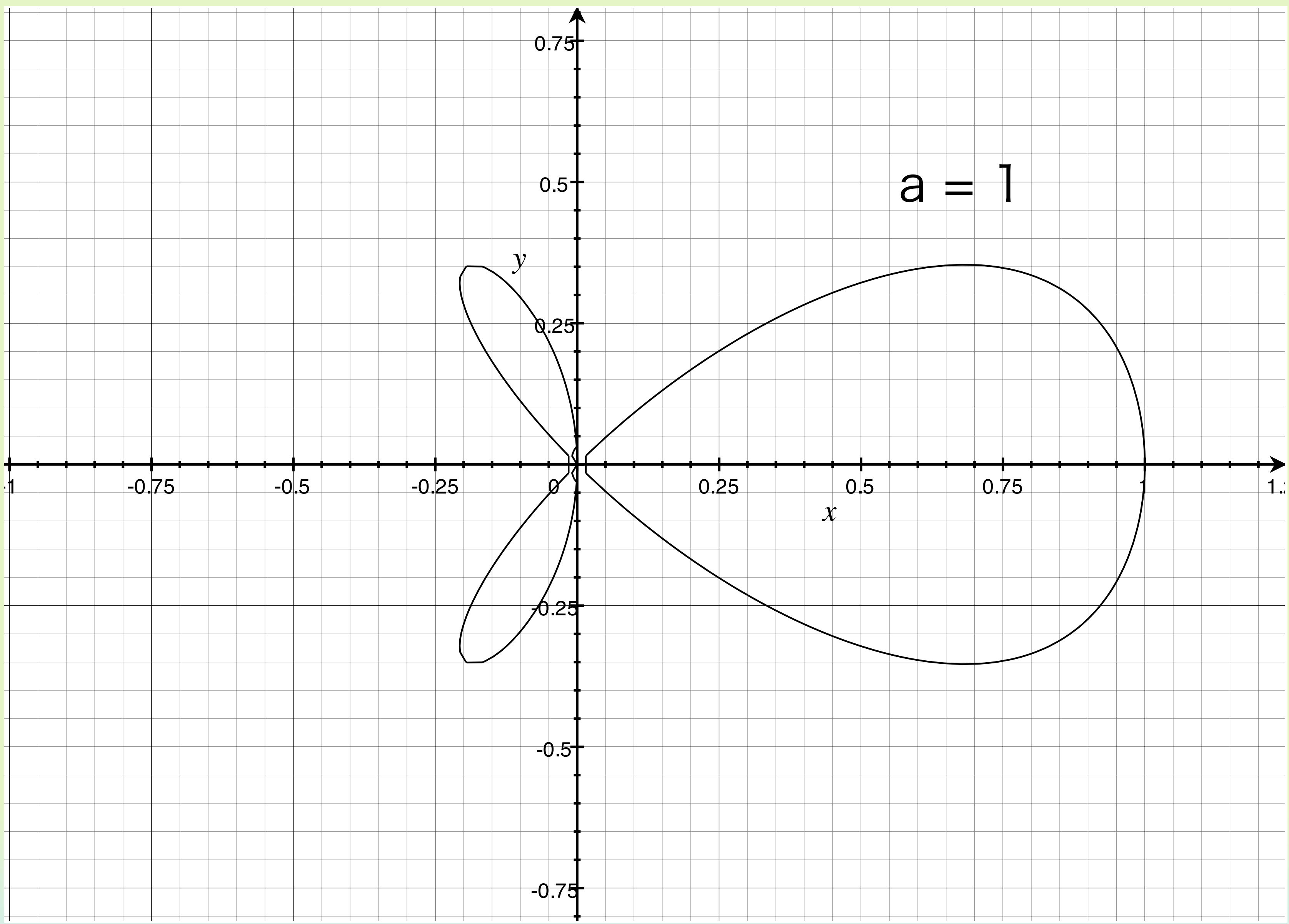
```
A = [[5, 2, 1],[2, 5, 1], [2, 2, 9]]
b = [3, 2, 1]
puts(kadai1(A, b, 1.0e-5))
```

出力：

```
0.5312126391799127,0.19788154280871262,-0.050735902394895553
```

課題２

曲線 $x^4 + y^4 = ax(x^2 - y^2)$ を $x = -a$〜a, $y = -a$〜aの範囲で描画するプログラム kadai2(a) を作れ．この曲線は以下のような形をしている．

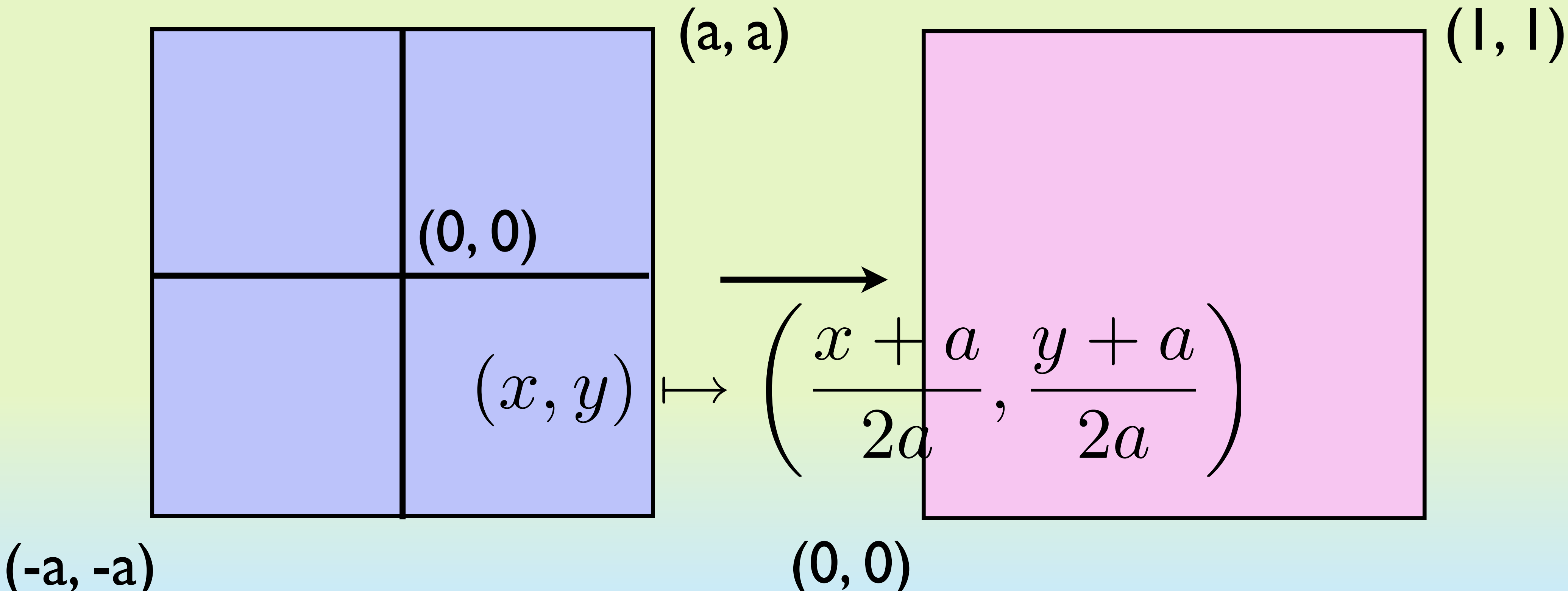


課題2へのヒント

曲線上の点で以下の式を満たす点を特異点という．特異点ではおかしなことが起こるので，そこをはずす必要がある．

$$\frac{\partial f}{\partial x} = \frac{\partial f}{\partial y} = 0$$

この曲線の場合，x = y = 0が特異点となる．特異点を避けて描画するのが賢い．さらに座標系は以下のように適当に線形変換することができる．



$$(x, y) \mapsto \left(\frac{x+a}{2a}, \frac{y+a}{2a}\right)$$

解答例２ (1)

- 基本的には授業で説明した描画方法による

```
function kadai2(a){
   var EPS = 0.002
   function sqr(x){return x * x}
   function quad(x){return sqr(x) * sqr(x)}
   function f(x, y, a) {
      return quad(x) + quad(y) -
         a * x * (sqr(x) - sqr(y))
   }
   function trans(x, a){
         return (x - 0.5) * 2 * a }
   function draw_it(pt1, pt2, a){
      var [x1, y1] = pt1
      var [x2, y2] = pt2
      if (Math.abs(x2 - x1) < EPS){
         draw_rect(pt1, pt2)
         return
      }
      var [xx1, yy1] = [trans(x1, a), trans(y1, a)]
      var [xx2, yy2] = [trans(x2, a), trans(y2, a)]
      var u1 = f(xx1, yy1, a)
      var u2 = f(xx1, yy2, a)
      var u3 = f(xx2, yy1, a)
      var u4 = f(xx2, yy2, a)
      if ((u1 > 0 && u2 > 0 && u3 > 0 && u4 > 0) ||
          (u1 < 0 && u2 < 0 && u3 < 0 && u4 < 0))
         return
      else {
          var [mx, my] = [(x1 + x2) / 2,
                          (y1 + y2) / 2]
          draw_it(pt1, [mx, my], a)
          draw_it([x1, my], [mx, y2], a)
          draw_it([mx, my], pt2, a)
          draw_it([mx, y1], [x2, my], a)
      }
   }
   var N = 26
   var delta = 1.0/N
   for (var i = 0; i < N; i++){
      for (var j = 0; j < N; j++){
        draw_it([i / N, j / N],
                [(i + 1) / N, (j + 1) / N], a)
      }
   }
   draw_line([0, 0.5], [1, 0.5])
   draw_line([0.5, 0], [0.5, 1])
}
```

- 初期分割をある程度細かくしてから描画する

解答例２ (2)

- 実際に描画させると以下のようになる.