計算幾何入門

~円~

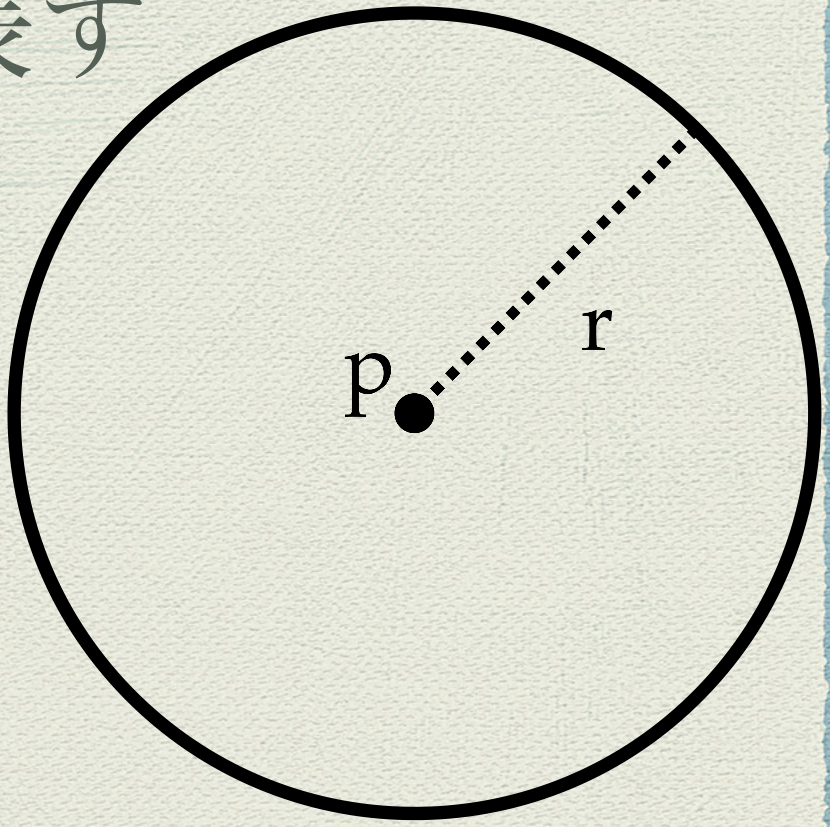
アルゴリズム研究室 D1
井上 祐馬

内容

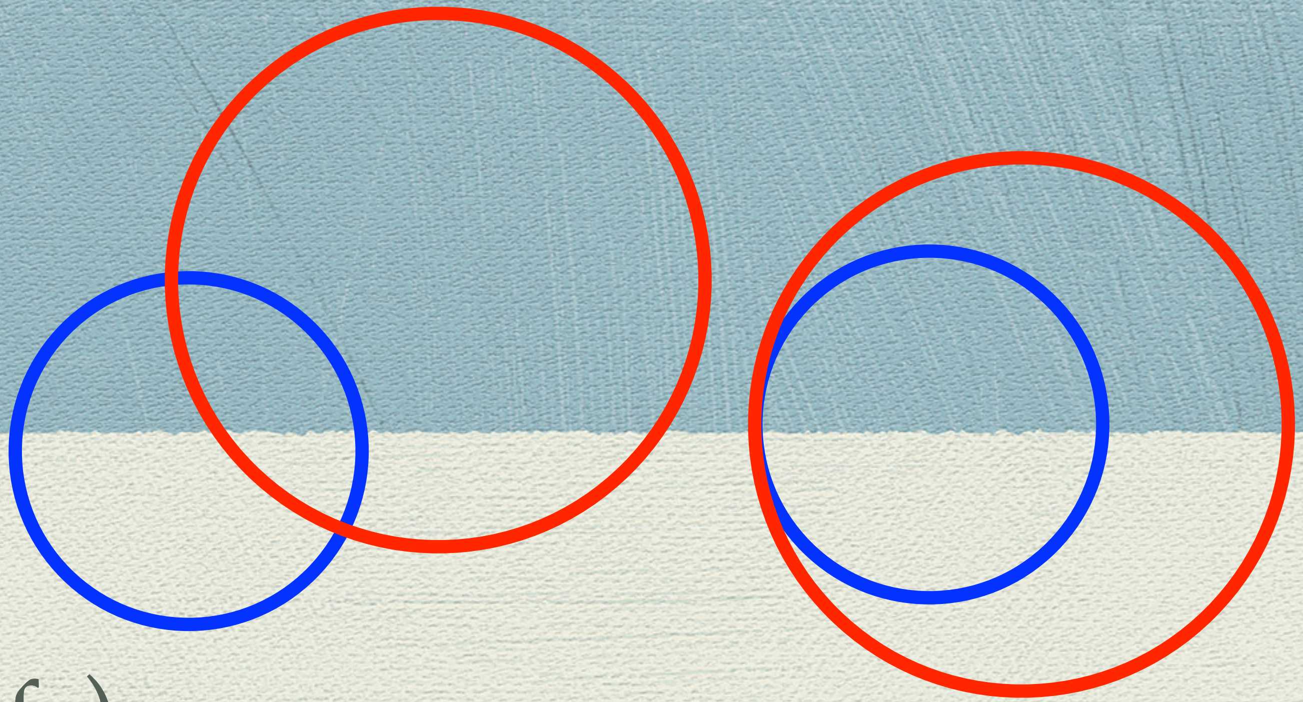
- 円の表現
    - 2円の交点
    - 直線との交点
    - 2円の共通部分の面積
    - ある1点を通る円の接線
    - 2円の共通接線

円の表現

- 円は中心の点pと半径rの組で表す
  - typedef pair<P, double> C;
- 基本事項
  - 直径 = 2r, 円周 = 2πr, 面積 = πr^2
  - 2円の面積共有判定: $dis(p_a, p_b) ≦ r_a + r_b$



2円の関係



```
int cpr(C a, C b){
  double d = abs(a.fs - b.fs);
  if(a.sc+b.sc + EPS < d)return -1;           //0 cp (outside)
  if(b.sc+d + EPS < a.sc) return 1;           //0 cp (B in A)
  if(a.sc+d + EPS < b.sc)return 2;            //0 cp (A in B)
  if(abs(a.sc+b.sc - d) < EPS)return -3;  //1 cp (outside)
  if(abs(a.sc+d - a.sc) < EPS)return 3;   //1 cp (B in A)
  if(abs(a.sc+d - b.sc) < EPS)return 4;   //1 cp (A in B)
  return 0;                                    //2 cp
}
```

角度と回転

- 3辺a,b,cの長さからaの対角の角度(rad)を求める(余弦定理)

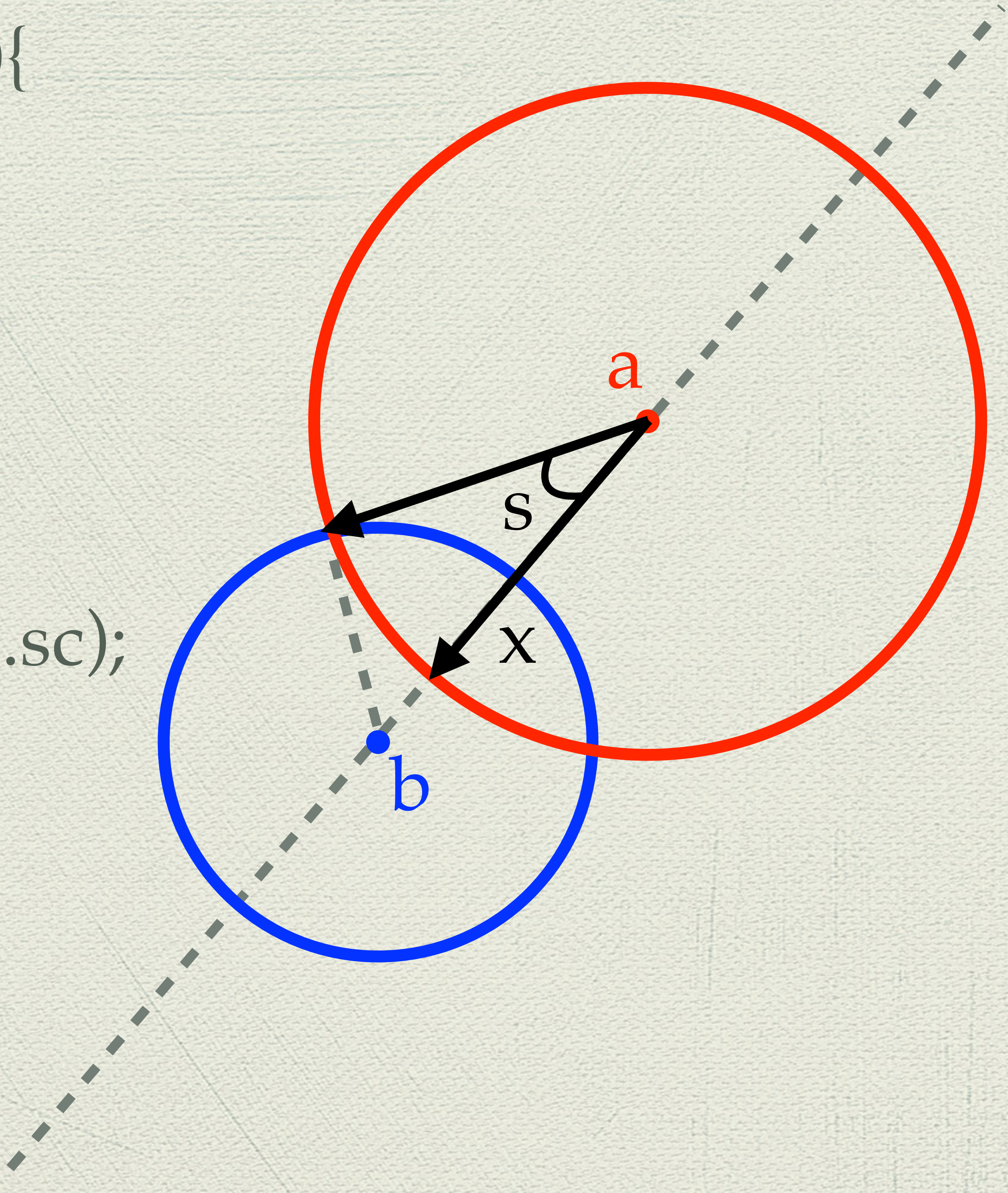
```
double arg(double a, double b, double c){
  return acos( (b*b+c*c - a*a)/(2*b*c) );
}
```

- 原点を中心にベクトルvを半時計回りにs (rad)回転させる
  (回転行列)

```
P rotate(P v, double s){
  return P(v.real()*cos(s) - v.imag()*sin(s),
                v.real()*sin(s) + v.imag()*cos(s) );
}
```

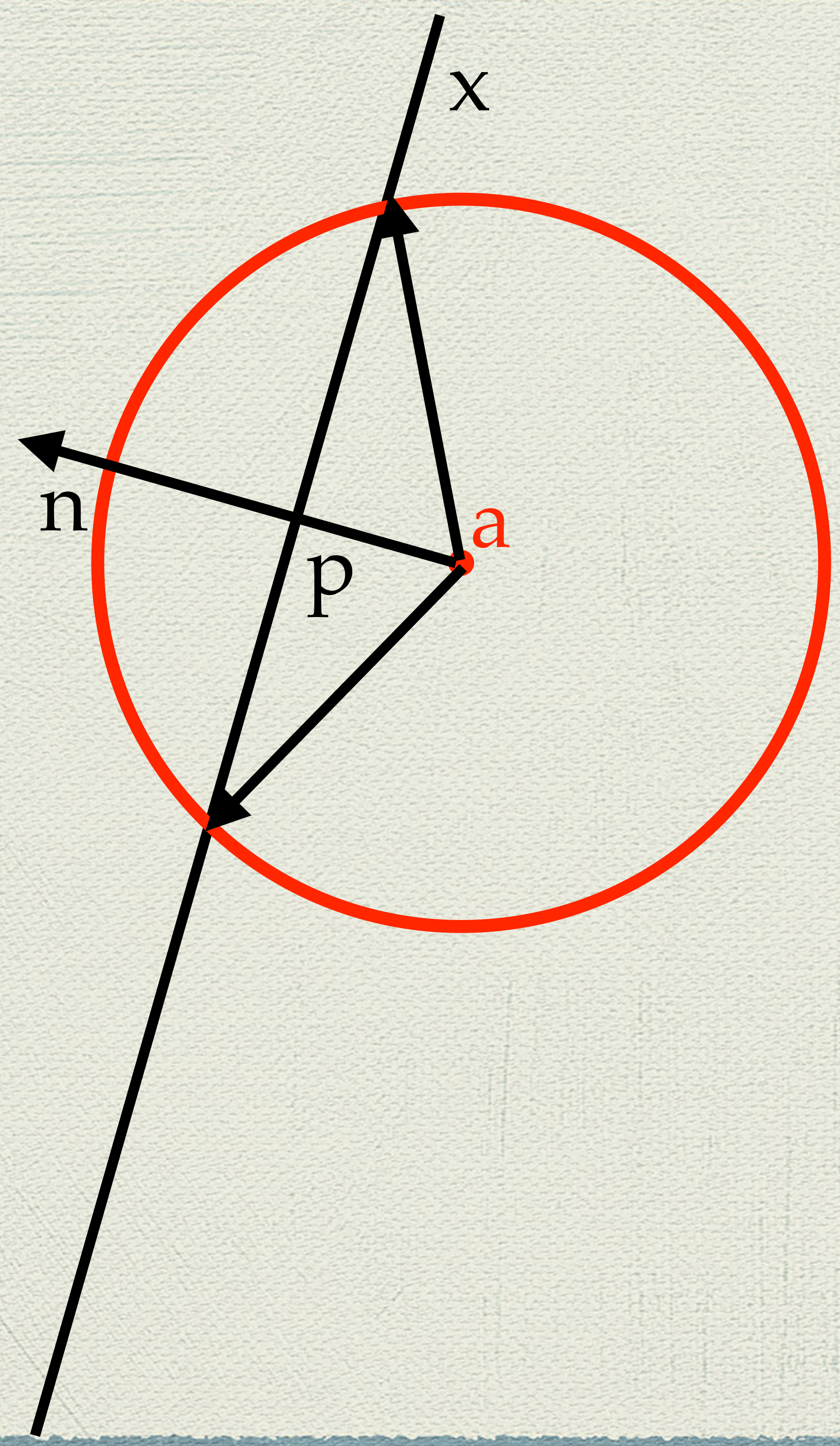
2円の交点

```
vector<P> CircleCrossPoint(C a, C b){
 vector<P> res;
 P x = a.sc * unit(b.fs - a.fs);
 if(cpr(a,b) >= 3){
  res.push_back(a.fs + x);
 }else if(cpr(a,b) == 0){
  double s = arg(b.sc, abs(b.fs-a.fs), a.sc);
  res.push_back(a.fs + rotate(x, s));
  res.push_back(a.fs + rotate(x,-s));
 }
 return res;
}
```



円と直線の交点

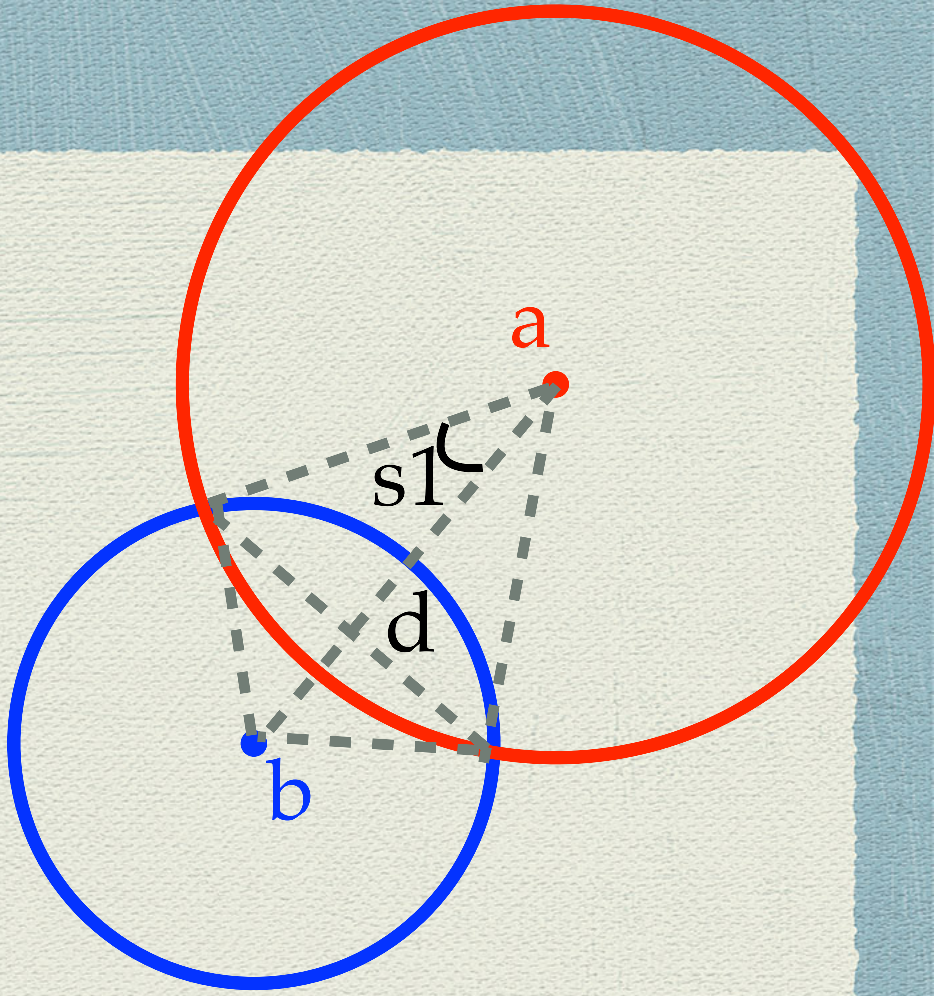
```
vector<P> cpCircleLine(C a, L x){
 vector<P> res;
 P n = normal(x.fs-x.sc).fs;
 P p = line_cp(x, L(a.fs,a.fs+n));
 if(abs(abs(a.fs-p) - d)<EPS){
  res.push_back(p);
 }else if(abs(a.fs-p)+EPS<d){
  D len = sqrt(a.sc*a.sc - norm(a.fs-p));
  P cp = len * unit(x.fs - x.sc);
  res.push_back(p + cp);
  res.push_back(p - cp);
 }
 return res;
}
```



2円の共通面積

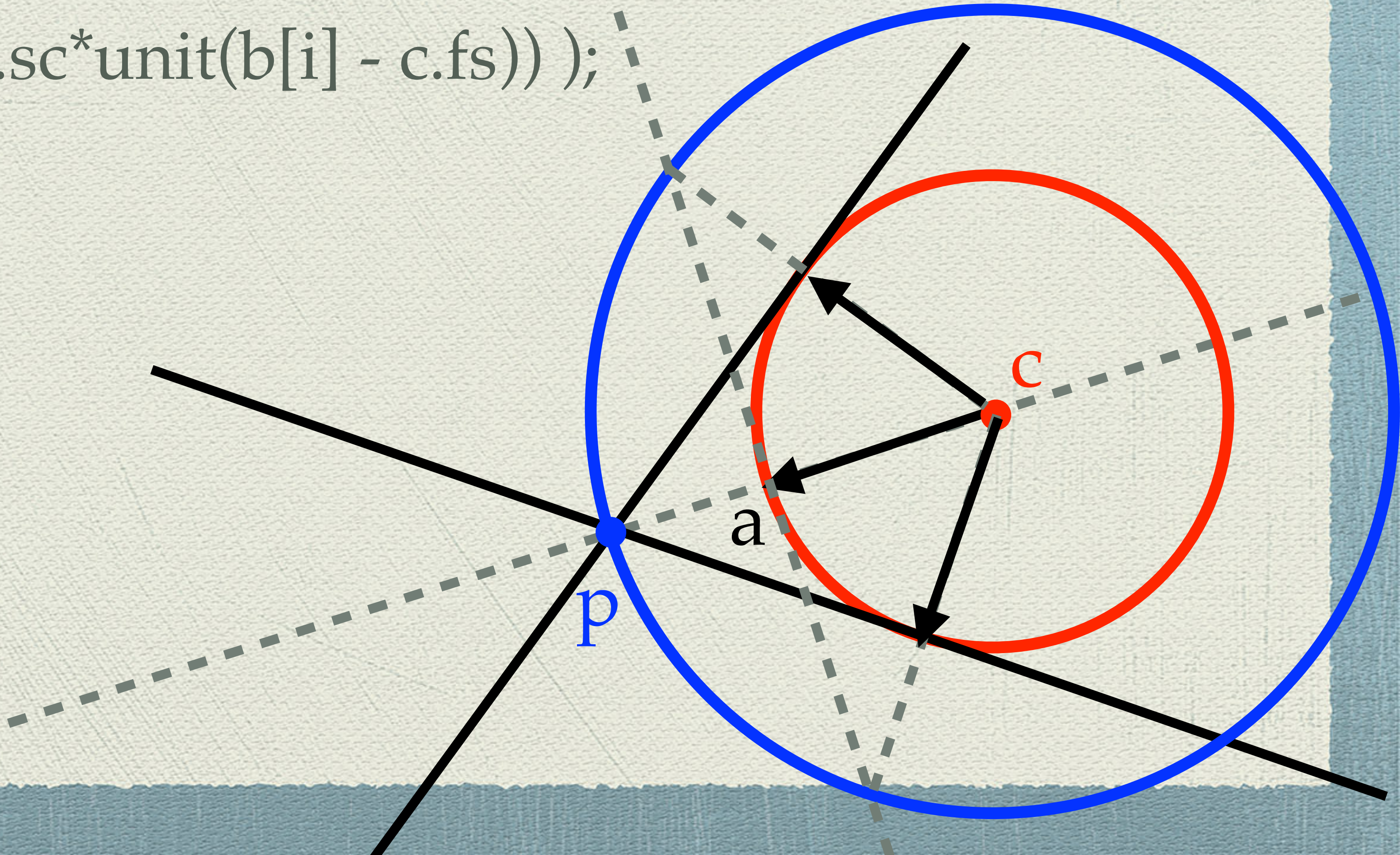
```
vector<P> CircleCrossPoint(C a, C b){
  double d = abs(a.fs - b.fs);
  if(a.sc + b.sc < d+EPS)return 0;
  if(a.sc < b.sc)swap(a,b);
  if(b.sc + d < a.sc + EPS)return area(b);

  double s1 = arg(b.sc,a.sc,d), s2 = arg(a.sc,b.sc,d);
  double tri = (a.sc*a.sc*sin(s1*2) + b.sc*b.sc*sin(s2*2) )/2;
  return a.sc*a.sc*s1 + b.sc*b.sc*s2 - tri;
}
```



ある1点を通る接線

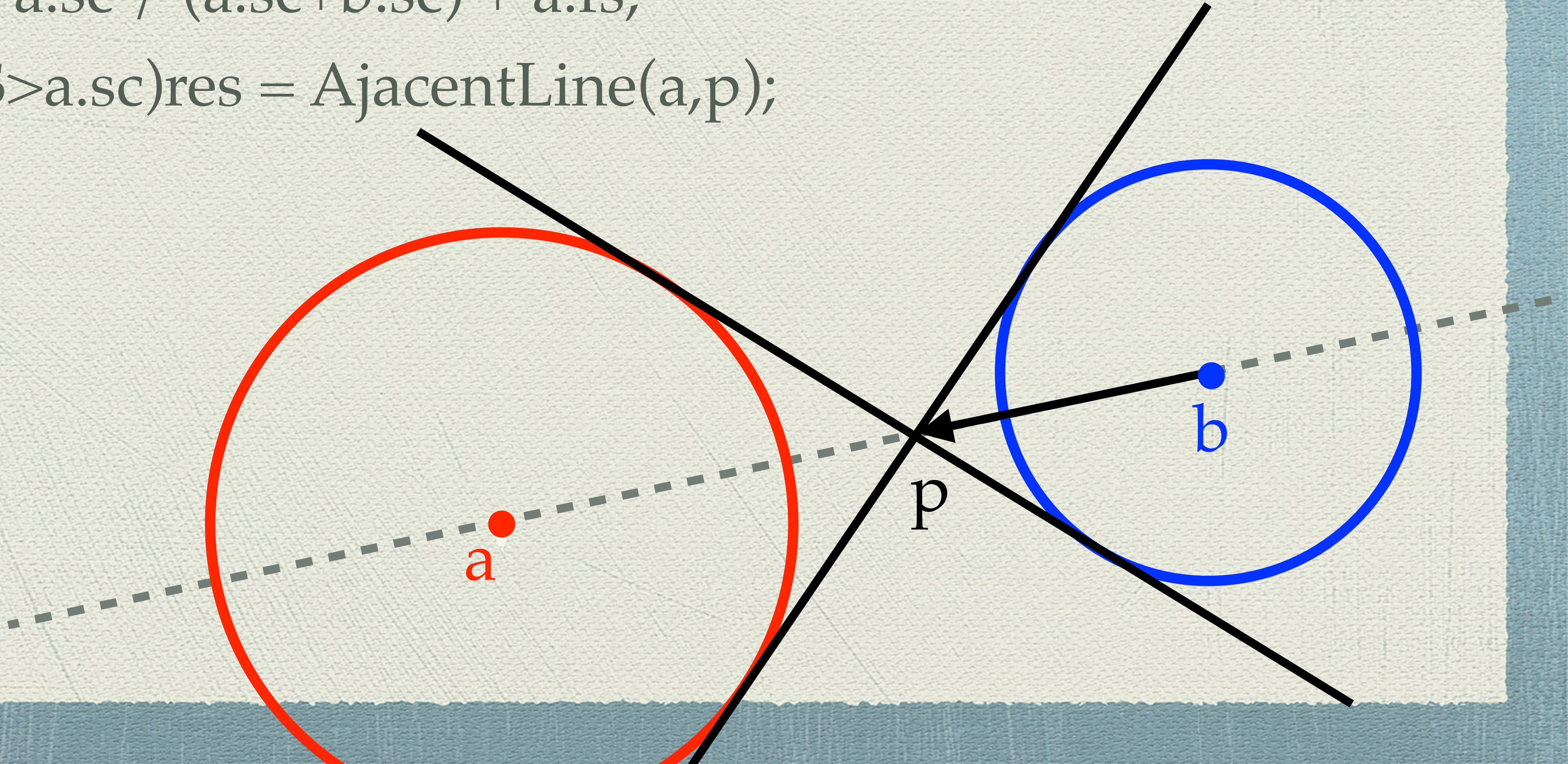
```
vector<L> TangentLine(C c, P p){
  vector<L> res;
  P a = c.fs + c.sc * unit(p - c.fs);
  vector<P> b = cpCircleLine( C(c.fs, abs(c.fs-p), L(a,a+normal(c.fs-p).fs) );
  for(int i=0;i<b.size();i++){
    res.push_back( L(p, c.fs + c.sc*unit(b[i] - c.fs)) );
  }
  return res;
}
```



2円の共通接線

```
vector<L> CommonTangentLine(C a, C b){
 vector<L> res;
 if(a.sc+EPS < b.sc)swap(a,b);
 if(a == b)return res;  //ちゃんと誤差処理する

 P p = (b.fs-a.fs)*a.sc / (a.sc+b.sc) + a.fs;
 if(abs(a-p)+EPS>a.sc)res = AjacentLine(a,p);
 ......
```



2円の共通接線

```
vector<L> CommonTangentLine(C a, C b){
 ......
 if(abs(a.sc-b.sc) < EPS){
  pair<P,P> n = normal(unit(b.fs-a.fs)*a.sc);
  res.push_back(L(a.fs+n.fs, b.fs+n.fs));
  res.push_back(L(a.fs+n.sc, b.fs+n.sc));
 }else{
  P p = (b.fs-a.fs) * a.sc/(a.sc-b.sc) + a.fs;
  if(abs(a-p)+EPS>a.sc){
   vector<L> tmp = AdjacentLine(a,p);
   for(int i=0;i<tmp.size();i++)res.push_back(tmp[i]);
  }
 }
 return res;
}
```