# Homework Collection (with Solutions)
# Computational Geometry and Modeling
# G22.3033.007, Spring 2005, Professor Yap

April 6, 2005

## 1 Homework List

SOLUTION PREPARED BY Instructor and T.A.s

This document will be a cumulative list of all the homeworks assigned in this class. It is in reverse chronological order.

---

Hw 4 INSTRUCTIONS

- Out: Apr 6; Due Apr 13
- We explore some further aspects of Delaunay tetrahedralizations.

1. Question 1. Let $S$ be a finite set of spheres (or weighted points). Define the Voronoi diagram as usual.

   (a) Give an explicit (numerical) example where a sphere $\widehat{p} \in S$ is redundant. Recall that Edelsbrunner defines a point to be "redundant" to mean its Voronoi region is empty.

   (b) Let $\widehat{p} = (p, P^2)$. If $p$ is a vertex of the convex hull of the centers in $S$, then $\widehat{p}$ is not redundant.

2. Question 2. Please read the Notes on Tetrahedralization, below.

   (a) Do PROBLEM 1.

   (b) Do PROBLEM 2.

   (c) Do PROBLEM 3.

   (d) Do PROBLEM 5 (not 4).

---

Hw 3 INSTRUCTIONS

- Out: Mar 2; Due Mar 9
- You will need to read Lecture VII (Sturm Theory) of my book, given here as classNotesIII.ps.

1. Question 1. Show how to convert the subresultant PRS algorithm into a Sturm sequence algorithm.

   What is its complexity? Assume the input polynomials have degree $\leq d$ and integer coefficients at most $L$ bits.

2. Question 2. Suppose in the previous algorithm, you only want to compute the signs of the leading coefficients of the Sturm sequence. What is its complexity?

3. Question 3. Edelsbrunner used the power diagram to prove an acyclicity property of Delaunay Triangulations. We will reconstruct these properties in this exercise.

   Let $C, C'$ be two circles whose centers are $A$ and $A'$. Assume that $A$ lies on the positive $x$-axis, and $A'$ on the negative $x$-axis. Let $pow(p, C) = \|p - A\| - r^2$ denote the **power** of a point $p$ to a circle $C$ centered at $A$ with radius $r$.
   (a) If $L$ is a vertical line, show that every point on $L$ has the same value of $pow(p, C) - pow(p, C')$.
   (b) There is a unique point $p_0$ on the $x$-axis such that $pow(p_0, C) = pow(p_0, C')$. Moreover, if $p$ lies to the left of $p_0$, $pow(p, C) > pow(p, C')$ and if $p$ lies to the right of $p_0$, $pow(p, C) < pow(p, C')$.
   (c) Prove the acyclicity theorem of Edelsbrunner.

---

Hw 2 INSTRUCTIONS

- Out: Wed Feb 16; Due: Wed Feb 23.
- Lecture 11 is the basis of our lecture on algebraic techniques. Please pick up the latest version of this lecture (out on Feb 16).

1. Question 1. In Lecture 11 (Class Notes, p.4), we noted that $\mathbb{Z}[\sqrt{-5}]$ is the ring of integers of $\mathbb{Q}[\sqrt{-5}]$ but it is not a UFD. Let us verify this.
   (i) By definition, $\mathbb{Q}[\sqrt{-5}]$ is only a ring. Show that it is really a field (i.e., every non-zero element has an inverse).

   **SOLUTION:** The inverse of any non-zero element $\alpha = a + b\sqrt{-5}$ is $\frac{a-b\sqrt{-5}}{a^2+5b^2}$. ■

   (ii) The "integers" in $\mathbb{Q}[\sqrt{-5}]$ is the set $\mathbb{Z}[\sqrt{-5}]$. The **conjugate** of a number $\alpha = a+b\sqrt{-5} \in \mathbb{Q}[\sqrt{-5}]$ is defined to be $\overline{\alpha} = a - b\sqrt{-5}$. The **norm** of $\alpha = a + b\sqrt{-5}$ is defined to be $N(\alpha) = \alpha\overline{\alpha} = a^2 + 5b^2$. Prove that $N(\alpha\beta) = N(\alpha)N(\beta)$.

   **SOLUTION:** $\alpha = a + b\sqrt{-5}, \beta = a' + b'\sqrt{-5}$. Then $\alpha\beta = (aa' - 5bb') + (ab' + a'b)\sqrt{-5}$. So $N(\alpha\beta) = (aa' - 5bb')^2 + 5(ab' + a'b)^2$ Also $N()N() = (a^2 + 5b^2)(a'^2 + 5b'^2)$. We verify that both sides are equal. ■

   (iii) Determine the units in $\mathbb{Z}[\sqrt{-5}]$. HINT: show that $\alpha$ is a unit iff $N(\alpha) = 1$.

   **SOLUTION:** If $\alpha\alpha' = 1$ then $N(\alpha)N(\alpha') = 1$, so $N(\alpha) = \pm 1$. But $N(\alpha)$ cannot be negative. CONVERSELY, if $N(\alpha) = 1$ then $\alpha = a + b\sqrt{-5}$ where $a = -1, b = 0$. Thus the only units are $\pm 1$. ■

   (iv) The numbers $2, 3, 1 + \sqrt{-5}, 1 - \sqrt{-5}$ are irreducible and not associates of each other.

   **SOLUTION:** $N(2) = 4, N(3) = 9, N(1 + \sqrt{-5}) = 6, N(1 - \sqrt{-5}) = 6$. Clearly the only possible associates are those of $N() = 6$. But since the units are $\pm 1$, these are not associates. WHY are they irreducible? If 2 is reducible, then there is some element of norm 2, but this is impossible. Similarly, there are no elements of norm 3. Thus All these values are irreducible. ■

   (v) Since $2 \cdot 3 = (1 + \sqrt{-5}) \cdot (1 - \sqrt{-5})$, conclude that $\mathbb{Z}[\sqrt{-5}]$ is not a UFD.

   **SOLUTION:** From these 2 expressions of 6, we conclude it is not UFD. ■

2. Question 2: Exercise 7.1 of Lecture 11 (p.13). Computing inverses.

3. Question 3: Exercise 7.2 of Lecture 11 (p.13). Computing polynomial GCD.

4. Question 4: Exercise 7.4 of Lecture 11 (p.13). Complexity of Subresultant PRS.

   **SOLUTION:** The number of operations required are $\mathcal{O}(d^2)$, where $d$ is the maximum among the degrees of the two polynomials. Suppose that the degree reduces by one at every step, then the operations at the $i^th$ step is $\mathcal{O}(d - i)$; summing over all the steps, which are atmost $d$, we get the mentioned operation count. Now we bound the size of the coefficients appearing in Subresultant PRS. Since each coefficient is a determinant of a matrix of size $\mathcal{O}(d-i)$, $i = 0, \ldots, d$, whose each entry has bit size L, the bit size of the determinant is bounded by $\mathcal{O}(dL + d\lg d)$. Thus the total bit operations required for computing Subresultant PRS is $\mathcal{O}(d^2 M(dL + d\lg d))$, where $M(n)$ denotes the bit-complexity of multiplying two $n$ bit numbers. ■

---

Hw 1 INSTRUCTIONS

- Due: The written part is due on Wed Feb 9, in class. The programming parts are due in 2 weeks, on Wed Feb 16.
- Please get started on the programming immediately. An appendix below will give you more guide on the programming part.

---

1. Question 1. Give the complexity of computing $[x/y]_n$ where $x, y$ are bigfloats. You must first determine a suitable algorithm, of course.

   Unlike the lecture notes (supplement to Lecture II), we no longer assume $y$ is a bounded bigfloat. These notes are found in this directory (classNotesII.ps).

   **SOLUTION:** The algorithm to compute $[x/y]_n$ when both the operands are unbounded bigfloats is:

   (a) Compute $[x]_{n+2}$.
   (b) Compute $[1/y]_{n+2}$.
   (c) Return $[x]_{n+2}[1/y]_{n+2}$.

   The complexity of the first step is $\mathcal{O}(n)$. The second step for unbounded bigfloats takes $\mathcal{O}(M(n) + \lg|e_y|)$, and the resulting inverse has an exponent bounded by $|e_y|+1$. The final step has the complexity $\mathcal{O}(M(n) + \lg(|e_x|(|e_y|+1)))$, which is the dominating factor and hence is the over all complexity. ■

2. Question 2. Give the complete analysis for the approximate Newton iteration for computing $\sqrt{c}$ where $c$ is a bounded bigfloat.

   HINT: Imitate our development for $1/c$ where you first do the analysis for exact Newton iteration, and then adapt the argument for the approximate iteration.

   **SOLUTION:**

   (a) Error-free Analysis for square-root. Let $f(x) = x^2 - c$, we are interested in computing $\sqrt{c}$. The Newton iterator for $f$ is $N_f(x) = \frac{x^2+c}{2x}$. Let $\varepsilon_i = \frac{x_i - \sqrt{c}}{\sqrt{c}}$, i.e., the relative error in the $i^{th}$ iterator. Then

   $$\begin{aligned}\varepsilon_{i+1} &= \frac{x_{i+1} - \sqrt{c}}{\sqrt{c}} \\ &= \frac{(x_i - \sqrt{c})^2}{2x_i\sqrt{c}} \\ &= \frac{\sqrt{c}\varepsilon_i^2}{2x_i}.\end{aligned}$$

   Observe that for $x > 0$, $N_f(x) \geq \sqrt{c}$ since $N_f(x) - \sqrt{c} = \frac{(x-\sqrt{c})^2}{2x}$ which is positive. Thus we may choose $x > 0$ and we are ensured that all $x_i > \sqrt{c}$, consequently we have $\varepsilon_{i+1} \leq \varepsilon_i^2/2$. Assuming $|\varepsilon_0| \leq 1$ we have $|\varepsilon_i| \leq 2^{-2^i+1}$.

(b) Analysis for sqaure-root considering error in computations. Let $\widetilde{x_i}$ denote the iterates with error and $\delta_i := \frac{\widetilde{x_i} - x^*}{x^*}$, where $x^* := \sqrt{c}$. Suppose we do the Newton iteration as follows:

i. $y_i \leftarrow [c/\widetilde{x_i}]_{2^{i+1}}$.

ii. $z_i \leftarrow [\widetilde{x_i} + y_i]_{2^{i+1}}$.

iii. $\widetilde{x_{i+1}} \leftarrow z_i/2$.

Note that the last step is error-free, since division by 2 is simply modifying the exponent. Also, it is clear that this step takes time $\mathcal{O}(M(2^i))$; not sure since there is addition. We next show that the iterates have quadratic convergence. From the first step we have

$$\begin{aligned}
y_i &= \frac{c}{\widetilde{x_i}}(1 \pm 2^{-2^{i+1}}) \\
&= \frac{x^*}{1+\delta_i}(1 \pm 2^{-2^{i+1}}) \\
&= x^*[1 - \delta_i + \sum_{j\geq 2}(-\delta_i^j) \pm 2^{-2^{i+1}} \sum_{j \geq 0}(-\delta_i)^j],
\end{aligned}$$

since $\widetilde{x_i} = x^*(1+\delta_i)$ and by the definition of $x^*$. The second step yields

$$\begin{aligned}
z_i &= (\widetilde{x_i} + y_i)(1 \pm 2^{-2^{i+1}}) \\
&= (x^*(1+\delta_i) + x^*[1 - \delta_i + \sum_{j\geq 2}(-\delta_i^j) \pm 2^{-2^{i+1}} \sum_{j\geq 0}(-\delta_i)^j])(1 \pm 2^{-2^{i+1}}) \\
&= x^*(2 + \sum_{j\geq 2}(-\delta_i^j) \pm 2^{-2^{i+1}} \sum_{j\geq 0}(-\delta_i)^j))(1 \pm 2^{-2^{i+1}})
\end{aligned}$$

Since $\widetilde{x_{i+1}} = z_i/2$ we have

$$\delta_{i+1} = (0.5 \sum_{j\geq 2}(-\delta_i^j) \pm 2^{-2^{i+1}-1} \sum_{j\geq 0}(-\delta_i)^j))(1 \pm 2^{-2^{i+1}}).$$

If $|\delta_i| \leq 2^{-2^i}$ then we have

$$\begin{aligned}
|\delta_{i+1}| &\leq (0.5 \sum_{j\geq 2}|\delta_i|^j + 2^{-2^{i+1}-1} \sum_{j\geq 0}|\delta_i|^j))(1 + 2^{-2^{i+1}}) \\
&= 0.5\delta_i^2(1 - \frac{|\delta_i|}{1+|\delta_i|}) + 2^{-2^{i+1}-1}(1 - \frac{|\delta_i|}{1+|\delta_i|}) + 2^{-2^{i+1}}\left[0.5|\delta_i|^2 \frac{1}{1+|\delta_i|} + 2^{-2^{i+1}-1}\frac{1}{1+|\delta_i|})\right] \\
&\leq 2^{-2^{i+1}} + 2^{-2^{i+2}} \frac{1}{1+|\delta_i|}.
\end{aligned}$$

Since each iteration step costs $M(2^i)$ and we require at most $\lg n$ steps, by the regularity condition we get the complexity of computing $[\sqrt{c}]_n$ as $\mathcal{O}(M(n))$.

■

3. Question 3. Answer question 1.1 (page 5) of Lecture I ClassNotes. In this question, you are to design experiments to evaluate the predicate in equation (3),

OnLine(Intersect(L,L'),L)

where L, L' are lines. Here are more specific instructions.

a. We suggest that you write a loop to automatically generate 50 lines and try all pairs intersections taken from these 50 lines (so you get 2500 tests). Experiment with different lines.

b. You are to write your code as simply as possible using C++. But I suggest with MINIMAL of the object-oriented fuss, not because I am against O-O, but because it can obscure our emphasis on the algorithms and can make it hard to read your code. For instance, it is surely nice to have a class for points, and a class for lines. The Point class will have members X and Y, but THERE is no reason to make X and Y private so that we need special methods to read/write X and Y. I want to be able to naturally say:

```
Point P;
P.X = 1.1;        // instead of P.X() = 1.1; or P.setX(1.1);
P.Y = P.X;        // instead of P.setY(P.X()); or P.Y()=P.X();
```

The basic rule is to make your code as COMPACT and TRANSPARENT as possible – but use your judgement.
c. Your program has to choose a Core Accuracy Level and include the Core Library. This amounts to the 2 extra lines in the beginning of your code:

```
#define CORE_LEVEL I          // Other levels are II, III, IV.
#include "CORE/CORE.h"
```

d. We want you to compile and run your program in Level I (which is standard C++ semantics), and again in Level III (which is our guaranteed precision semantics). Compare the outputs from these two levels, and draw your conclusions.
e. Make sure that all your numbers (e.g., P.X, P.Y) are declared to be type `double`. This is important because in Level I, `double` is what you expect, but in Level III, it is really a special number type we call `Expr` (expression).

4. Question 4. Implement the Newton-based algorithm to compute $[x/y]_n$ where $x, y$ are bigfloats. You are to use the Core Library, but MAINLY to use its BigFloat class. Of course, in this environment, you can also easily check the output of your computation with our guaranteed precision outputs. We will give you all the hints needed to do this problem.

   You must know that the BigFloat in Core Library is not normalized, and it has an ERROR component. In other words, a Core BigFloat is a triple $(exp, man, err)$ of integers which represents the interval

$$[B^{exp} \times (man - err), B^{exp} \times (man + err)]$$

   and $B = 2^{14}$ (so the base is $2^{14}$). If $err = 0$, we say the BigFloat is **exact** (or error-free). You only want to deal with exact BigFloats, so you may need to set the error component to zero after an operation like division! IT IS IMPORTANT TO UNDERSTAND WHY THIS IS ESSENTIAL. Most of these are explained in the Core Library Tutorial. But in a few days, I will put out a set of notes for this class to make it easier for you.

# 2 General Instructions on Programming

- We recommend that you use the Cygwin Environment for your computing platform, and use gnu's g++ for your C++ compiler. Our Core Library is optimized for this environment. Cygwin is a Unix-emulator on the Windows environment; basic information are found in a link in our class webpage. It takes less than 30 minutes to get this environment up and running on your PC with a fast network connection. Using the Setup Program of cygwin, be sure to download gcc, "make" and gmp. Here, "gmp" is Gnu's multiprecision package that Core Library is based on.

- In the class webpage, we have links to Cygwin information. To start, you should immediately take these steps.
  (1) go to www.cygwin.com to download the Cygwin Setup program.
  (2) Run the Setup Program to get your initial cygwin environment – I suggest that you accept ALL the defaults including the download of just the MINIMAL cygwin environment. Choose the option of "direct installation" from the web.

(3) Rerun Cygwin Setup to pick up any additional software: you will need at least gcc, make, gmp and some editor like vim or emacs.
(4) Download Core Library from http://cs.nyu.edu/exact. (5) Compile the Core Library following the README instructions. (6) Read the Core Library Tutorial.

- HINT: If you are indifferent to editors, then I highly recommend you get "gvim" directly from the internet, and put a link from cygwin to gvim. Gvim is an enhanced GUI version of "vim" with many nice features.

- For the programming part, Vikram (sharma@cs.nyu.edu) will be helping me in trying to get you started. So if you send emails, cc to both of us.

# 3 Notes on Tetrahedralization

Let $S \subseteq \mathbb{R}^3$ be a set of $n \geq 4$ points, not all coplanar.

Let $K$ be any simplicial complex with vertices in $S$. Let $f_j = f_j(K)$ be the number of $j$-dimensional facets of $K$, $j = 0, 1, 2, 3$. Thus $f_0(K)$ is the number of vertices and $f_3(K)$ is the number of tetrahedrons. Also write

$$f_j = f_j^B + f_j^I, \qquad (j = 0, \ldots, 3) \tag{1}$$

where $f_j^B, f_j^I$ is the number of $j$-facets in the boundary and interior (resp.) of the convex hull of $S$. To avoid subscripts, we also use the notations

$$v = f_0, \quad e = f_1, \quad f = f_2, \quad t = f_3$$

for the number of vertices, edges, faces and tetrahedrons. Similarly, write $v^I = f_0^I, e^B = f_1^B$, etc. Note that

$$v = n, \qquad t^B = 0. \tag{2}$$

Thus, of the 12 values $f_j, f_j^B, f_j^I$, only 6 are independent because of Equations (1) and (2). We focus on the following 6 independent values:

$$\begin{array}{lll} v^B = f_0^B, & e^B = f_1^B, & e^I = f_1^I, \\ f^B = f_2^B, & f^I = f_2^I, & t = f^3. \end{array} \tag{3}$$

Furthermore, three of the values in (3) depend only on the set $S$:

$$v^B, \quad e^B, \quad f^B.$$

Thus different choices of $K$ can only influence

$$e^I, \quad f^I, \quad t. \tag{4}$$

To illustrate, consider two tetrahedralizations of the convex hexahedron on 5 vertices *abcde*, illustrated in Figure 1:

The hexahedron *abcde* is also known as a triangular dipyramid. Let $K_0, K_1$ denote the two tetrahedralizations. Then we have

$$\begin{array}{lll} e^I(K_0) = 0, & f^I(K_0) = 1, & t(K_0) = 2 \\ e^I(K_1) = 1, & f^I(K_1) = 3, & t(K_1) = 3 \end{array}$$

It is also useful to recall the basic insight from Edelsbrunner: $K_0$ and $K_1$ are just the projections of the upper and lower facial complex of a simplex in $\mathbb{R}^4$.

Clearly, $e^I = 0$ can be achieved by a set $S$ in convex position. How large can $e^I$ be? Towards this end, we may define $e^I(n) = \max\{e^I(K) : v(K) = n\}$, the maximum as $K$ range over all tetrahedralizations on $n$ vertices. Similarly, define $f^I(n)$ and $t(n)$. Consider the following example:

Let $S_n$ be the following set of $n$ points: $n$ is even and $n/2$ points are equally spaced on the unit circle in the $(x, y)$-plane. The remaining $n/2$ points are also equally spaced on the $z$-axis in the interval $-1 \leq z \leq 1$.

PROBLEM 1: Show that $t(n) = \Omega(n^2)$. HINT: Give a tetrahedralization of $S_n$ with $t(K) = \Omega(n^2)$.

PROBLEM 2: Is it true that every tetrahedralization of $S_n$ has $t(K) \geq \Omega(n^2)$?

We have the following relationships:

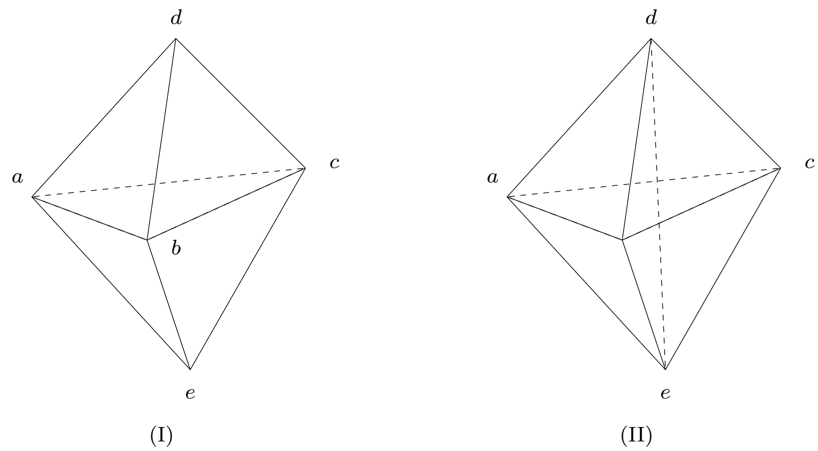


Figure 1: Two tetrahedralizations of $S = \{a, b, c, d, e\}$: (I) $K = \{abcd, abce\}$ (II) $K = \{abde, bcde, cade\}$

LEMMA 1 (FUHRING)
*(a)* $4t = f^B + 2f^I$.
*(b)* $f^B = 2v^B - 4$.
*(c)* $f^I = v^B + 2(e^I - v^I) - 4$.
*(d)* $2e^B = 3f^B$.
*(e)* $t = v^B - v^I + e^I - 3$.

PROBLEM 3: Prove these relationships.

We do not expect there to be a *single* $K = K_n$ that simultaneously achieve the bounds $e^I(K) = e^I(n), f^I(K) = f^I(n), t(K) = t(n)$. Nevertheless, it follows from (c) and (d) that $e^I(K), f^I(K), t(K)$ are linearly related. In particular, the statements $e^I(K) \geq \Omega(n^2)$, $f^I(K) \geq \Omega(n^2)$ and $t(K) \geq \Omega(n^2)$ are equivalent (cf. PROBLEM 2).

PROBLEM 4: A **hexahedron** is a polyhedron with 6 faces. The cube is another hexahedron, but it has 8 vertices. The **skeleton** of a polyhedron $P$ is the graph $(V, E)$ where $V$ are the vertices of $P$ and $E$ corresponding to the edges of $P$. Let us distinguish $P$ up to isomorphism of its skeleton. Show the following:
(a) The triangular dipyramid is the unique (up to skeleton type) hexahedron on 5 vertices.
(b) List all the skeleton types for hexadrons (there are 7).

**Angles of a tetrahedron.** There are 12 planar angles (three in each of the 4 triangular faces), 6 dihedral angles (one at each of the 6 faces), and 4 solid or trihedral angles (one at each of the 4 vertices). How do we measure dihedral angles? If we intersect the two faces forming the dihedral angle with a plane that is normal to the edge of the dihedral angle, we see a planar angle. This planar angle is defined as the measure of the dihedral angle.

How do we measure solid angles? If we intersect the three faces forming the solid angle with a unit sphere centered at the vertex $v_0$ of the dihedral angle, we see a triangular patch on the unit sphere. The area of this patch is defined as the measure of the solid angle. It is generally nontrivial to measure solid angles – they can be expressed as double integrals. However, in case of a tetrahedron, the solid angle can be given by the simple formula

$$\alpha + \beta + \gamma - \pi$$

where $\alpha, \beta, \gamma$ are the dihedral angles at the three edges incident at $v_0$.

**Criteria for Delaunay Tetrahedralization.** Let $K$ be any tetrahedralization of $S$.

We say $K$ satisfy the **sphere criterion** if for any tetrahedron $abcd$ of $K$, the circumsphere $C(a, b, c, d)$ contains no points of $S$ in its interior.

Let $\min(K)$ denote the minimum solid angle in the tetrahedrons of $K$. We say $K$ satisfy the **maxmin solid angle criterion** if $\min(K)$ is maximum over all tetrahedralizations of $S$.

Assume no five points in $S$ are co-spherical. In this case, $S$ has a unique Delaunay tetrahedralization. Moreover, $K$ satisfies the sphere criterion iff $K$ is Delaunay.

In two dimensions, Lawson showed that the analogous circle criterion and maxmin angle criterion are equivalent, and both yields the Delaunay triangulation.

In three dimensions, consider this example from Barry Joe: $S_5 = \{a, b, c, d, e\}$ where

$$a = (0,0,0), \quad b = (2,0,0), \quad c = (2,2,0), \quad d = (1.5, 0.5, 2), \quad e = (1.5, 0.5, -0.5).$$

PROBLEM 5: Show that the Delaunay triangulation of $S_5$ does not satisfy the maxmin solid angle criterion.

**Local Optimality.** Let *abcd* and *abce* be two tetrahedons with common face *abc* and $d, e$ on opposite sides of this face. The face *abc* is **locally optimal** if the circumsphere $C(a, b, c, d)$ does not contain $e$ in its interior.

We observe that the interior of $C(a, b, c, d)$ does not contain $e$ iff interior of $C(a, b, c, e)$ does not contain $d$. [Proof: Use the fact that the intersection of the circumspheres of *abcd* and *abce* is a circumscribed circle of triangle *abc*.]

LEMMA 2 (JOE) *Let $a, b, c, d, e$ be 5 points in convex position, as in Figure 1.*
*(a) Either the interior abc is locally optimal in (I) or the three interior faces $ade, bde, cde$ are all locally optimal in (II).*
*(b) If the five vertices are not co-spherical, then exactly one of the two local optimality conditions in (a) can hold.*

PROBLEM 6: Prove this.