3 Probabilistic analyses: randomized algorithms, evenly distributed points.

Use Maple or Wolfram Alpha for computing integrals.

3.1 Nearest neighbors

3.1.1 First

Let X be Poisson point process of intensity 1 in the plane. What is the expected distance between the origin an its nearest neighbor?

3.1.2 Second

What is the expected distance between the origin an its second nearest neighbor?

3.1.3 Others

What is the expected distance between the origin an its k^{th} nearest neighbor?

3.2 Perimeter of the Delaunay star

3.2.1 One triangle

O denotes the origin. Let $p = (\cos\alpha_1, \sin\alpha_1)$, $q = (\cos\alpha_2, \sin\alpha_2)$, and $t = (\cos\alpha_3, \sin\alpha_3)$ be three points on the unit circle (α_i is not restricted to $[0, 2\pi]$ you can choose othe rintervals if more convenient).

— Give condition on α_1, α_2, and $alpha_3$ for $p_1p_2p_3$ counterclockwise.
— Give a simple expression of the length $\|p_1p_2\|$ in α_1, α_2, and $alpha_3$.
— Give condition on α_1, α_2, and $alpha_3$ for p_1p_2O counterclockwise .

3.2.2 One circle

Give a condition to have the origin inside the disk of center $R(\cos\theta, \sin\theta)$ and radius r.

3.2.3 All edges

Let X be Poisson point process of intensity n in the plane. What is the expected total length of the edges of the triangles in conflict with the origin? (Edges belonging to two triangles are counted twice).

Hint: use Slivnyak-Mecke formula and Blaschke-Petkantschin variable substition.

3.2.4 Perimeter

Let X be Poisson point process of intensity n in the plane. What is the expected perimeter of the union of the triangles in conflict with the origin? (Edges belonging to two triangles are not counted).

Hint: use Slivnyak-Mecke formula and Blaschke-Petkantschin variable substition. To cancel the contribution of inside edges appearing twice, count them once positively and once negatively.

3.3 Randomized median

Consider the following algorithm to find the k^{th} number in a set X of n numbers.

```
    Number_of_rank(k, X):
m= random element in X;
Y = numbers of X smaller than m
```

```
Z = numbers of X greater than m
l = rank of m in X (♯(Y) + 1);
if (k=l) return m;
if (k<l) return Number_of_rank(k, Y )
if (k>l) return Number_of_rank(k-l, Z )
```

3.3.1 Worst case

What is the worst case complexity of this algorithm?

3.3.2 Recursive call

Give an upperbound on the expected size of Y? of Z'? of the biggest of the two sets?

3.3.3 Randomized complexity

Denote $f(n)$ the expected complexity of the algorithm for n points. We can assume that f is an increasing function.

What is the behavior of f?