ESE 5410 Week 2 Practice Problems - Solutions

**Problem 1**. Consider the additive model $Y = X^2 + \varepsilon$, where $\varepsilon \sim \text{Unif}[-1, 1]$, i.e., a uniform random variable in the range $[-1, 1]$. Answer the following questions:

1. Plot the conditional distributions $f_{Y|X}(y, 1/2)$ (i.e., the random variable $X$ is equal to $1/2$).

We have that $(Y|X = 1/2) = \varepsilon + \frac{1}{4} \sim \text{Unif}\left[-1 + \frac{1}{4}, 1 + \frac{1}{4}\right] = \text{Unif}\left[-\frac{3}{4}, \frac{5}{4}\right]$.

2. Compute and plot the regression function $f(x) = \mathbb{E}(Y|X = x)$ and the conditional variance $\text{Var}(Y|X = x)$.

We have that $(Y|X = x) = x^2 + \varepsilon$, which is uniform in the range $\left[x^2 - 1, x^2 + 1\right]$. The expectation of this conditional density is the middle point of the extremes of this range, hence,

$$\mathbb{E}(Y|X = x) = \frac{x^2 - 1 + x^2 + 1}{2} = x^2.$$

$$Var(Y|X = x) = \frac{[(x^2+1)-(x^2-1)]^2}{12} = \frac{1}{3}$$

3. Consider the estimator $\hat{Y}(X) = X^3$. Compute $\mathbb{E}[(Y - \hat{Y})^2|X = x]$.

$$\begin{aligned}
\mathbb{E}[(Y - \hat{Y})^2|X = x] &= \mathbb{E}[(X^2 + \varepsilon - X^3)^2|X = x] \\
&= \mathbb{E}[(x^2 - x^3 + \varepsilon)^2] \\
&= \mathbb{E}[(x^2 - x^3)^2 + 2\varepsilon(x^2 - x^3) + \varepsilon^2] \\
&= \mathbb{E}[(x^2 - x^3)^2] + \mathbb{E}[2\varepsilon(x^2 - x^3)] + \mathbb{E}[\varepsilon^2] \\
&= (x^2 - x^3)^2 + 2(x^2 - x^3)\mathbb{E}[\varepsilon] + \mathbb{E}[\varepsilon^2] \\
&= \left(x^2 - x^3\right)^2 + \mathbb{E}[\varepsilon^2]
\end{aligned}$$

The term $\mathbb{E}[\varepsilon^2]$ is the variance of the distribution $\text{Unif}[-1, 1]$, which is $1/3$; hence,

$$\mathbb{E}[(Y - \hat{Y})^2|X = x] = \left(x^2 - x^3\right)^2 + \frac{1}{3}.$$

4. Indicate the reducible errors for the estimator in sub part (3). Can you find a different estimator with a smaller reducible error?

For $\hat{Y}$, the reducible error is $(x^2 - x^3)$ and the irreducible error is $Var(\epsilon) = \frac{1}{3}$.

Since the noise $\varepsilon$ has zero mean, it makes sense to estimate $Y$ using the estimator $X^2$. One can easily validate that the reducible error in this case is zero, which is the best one can do.

**Problem 2**. For each of sub parts (1) through (4), indicate whether we would generally expect the performance of a **flexible** statistical leaning method to be better or worse than an **inflexible** method. Justify your answer.

1. The sample size N is extremely large, and the number of predictors p is small.

The flexible method would be better when the same size is large, because it would contain a larger number of points and fit the data closer (thus provide a closer approximation to f). Moreover, increasing n will reduce variance because the samples will be closer to each other.

2. The number of predictors p is extremely large, and the number of observations n is small.

The flexible method would be worse. This is because it would over-fit the small number of datapoints, try to connect every single one of them when they are further away, and be quite susceptible to noise. This doesn't provide a general pattern.

3. The relationship between the predictors and response is highly non-linear.

The flexible method would be better. If the method is inflexible, we would increase bias by not giving providing more degrees of freedom. We would be trying to approximate a complex system using a very simple and linear model, which would not be accurate. Therefore, the flexible model would better fit the non-linear relationship.

4. The variance of the error terms, i.e. $\sigma^2 = \text{Var}(\epsilon)$, is extremely high.

The flexible method would be worse, because it would be very variant to noise and error and fit closer to that instead of estimating the true function.

**Problem 3**. You will now think of some real-life applications for statistical learning. (This question is open-ended, many solutions are possible.)

1. Describe three real-life applications in which classification might be useful. Describe the response, as well as the predictors. Is the goal of each application inference or prediction? Explain your answer.

One application: classifying water as safe or unsafe. The predictors are pH level, bacteria level, intake temperature, dissolved oxygen, turbidity, etc. This is a prediction problem because we are interested in understanding how the predictors will identify the output rather than what the relationships between them are.

2. Describe three real-life applications in which regression might be useful. Describe the response, as well as the predictors. Is the goal of each application inference or prediction? Explain your answer.

One application: predicting students' future incomes. The predictors are number of degrees, the major, the number of internships, GPA, personality traits, etc. This is a prediction problem because we are trying to understand the effect of these predictors on the future income and predict using data of income of students who have graduated and got a job.

3. Describe three real-life applications in which cluster analysis might be useful.

One application: customer segmentation. Predictors: needs, attitudes, demographics, and behavior of customers. The researcher then may use cluster analysis to identify homogenous groups of customers that have similar needs and attitudes. This is a prediction model.

**Problem 4**. Recall the bias-variance decomposition. (The solution is provided in one of the quizzes.)

1. Provide a sketch of typical (squared) bias, variance, training error, test error, and Bayes (or irreducible) error curves, on a single plot, as we go from less flexible statistical learning methods towards more flexible approaches. The x-axis should represent the amount of flexibility in the method, and the y-axis should represent the values for each curve. There should be five curves. Make sure to label each one.

2. Explain why each of the five curves has the shape displayed in sub part (1).

**Problem 5**. In this exercise, you shall see the curse of dimensionality in action on a randomly generated dataset. Let's consider a $p$-dimensional hypersphere of radius 1, denoted by $S_p$, contained in a $p$-dimensional hypercube of side length = 2, denoted by $C_p$, both centered at the origin. The dataset under consideration corresponds to $n$ independent points chosen at random from the uniform probability distribution inside the $p$-dimensional hypercube.

1. Find expressions for the volume of the hypercube and the hypersphere to calculate the expected number of points inside the hypersphere $S_p$. Since there are $n$ points in the hypercube, what is the fraction of point contained inside the hypersphere? Show a plot of this fraction as a function of $p$, with $p$ going from 2 to 12. Hint: You might find math.gamma() useful to calculate the volume of the sphere.

2. We are now going to verify that these theoretical predictions work in practice using Python. Using the command numpy.random.uniform(), generate 100 points uniformly at random in the 2-dimensional hypercube (i.e., $C_2$ is just a square).

3. Using the above cloud of 100 points, calculate the fraction of points inside $S_2$ (i.e., the unit circle) with respect to the total number of points inside $C_2$.

4. Now repeat (2) and (3) with $p$ going from 2 to 12, and plot the resulting fraction of points as a function of $p$. What do you see? How does it compare to the theoretical result from (1)? Try repeating with more than 100 samples.

5. Finally, we are going to verify that, as $p$ increases, the distances between pairs of points increase. Using the datasets generated in the previous questions, compute numerically the average pairwise distance among all the points. Plot your result as a function of $p$. What does the curve look like? How does this relate to the results of (4)?

# ESE 5410 Week 2 Practice Problems P5

January 18, 2021

## 1 Problem 5

In this exercise, you shall see the curse of dimensionality in action on a randomly generated dataset. Let's consider a $p$-dimensional hypersphere of radius 1, denoted by $S_p$, contained in a $p$-dimensional hypercube of side length = 2, denoted by $C_p$, both centered at the origin. The dataset under consideration corresponds to $n$ independent points chosen at random from the uniform probability distribution inside the $p$-dimensional hypercube.

[1]:
```
import math
import numpy as np
import matplotlib.pyplot as plt
from scipy.spatial.distance import pdist
```

### 1.1 Part 1

Find expressions for the volume of the hypercube and the hypersphere to calculate the expected number of points inside the hypersphere $S_p$. Since there are $n$ points in the hypercube, what is the fraction of point contained inside the hypersphere? Show a plot of this fraction as a function of $p$, with $p$ going from 2 to 12. Hint: You might find math.gamma() useful to calculate the volume of the sphere.

#### 1.1.1 Solution

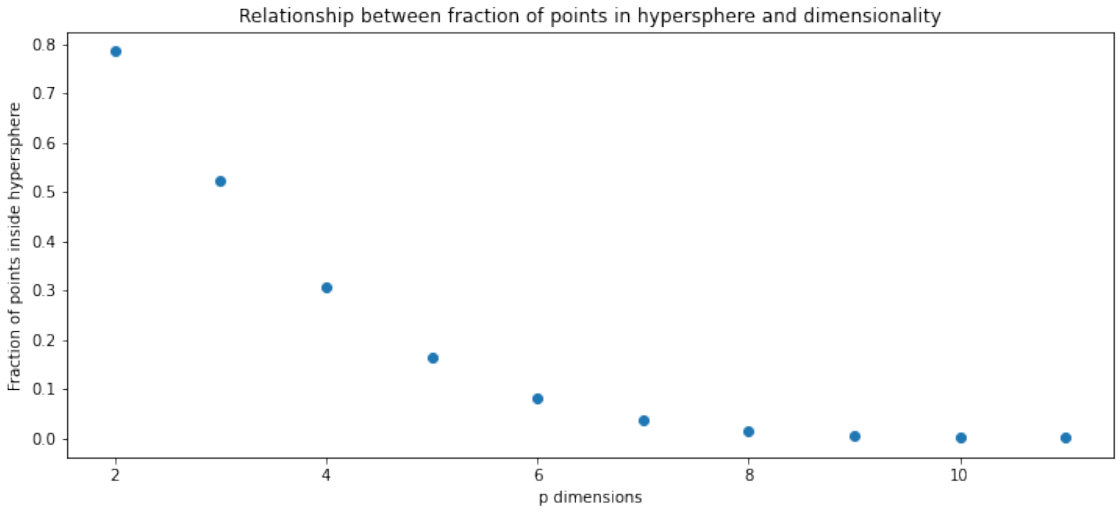
Let $V_{S_p}$ denote the volume of a $p$-dimensional hypersphere with $n$ independent points chosen at random from a uniform probability distribution with a radius of 1, and $V_{C_p}$ denote the volume of a $p$-dimensional hypercube with $n$ independent points chosen at random from a uniform probability distribution with side length 2.

It is a fact that the volume of any cube is its side length raised to its dimensional power. Thus, $V_{C_p} = 2^p$.

Furthermore, we know that the volume of a hypersphere can be written as $V_{S_p} = \pi^{\frac{p}{2}} \times \frac{R^p}{\Gamma(\frac{p}{2}+1)}$, where $R$ is the radius of the hypersphere. Read more about the formula here. If we define a point on the hypersphere as a unit with volume 1, then a hypersphere of radius 1 is expected to have $\pi^{\frac{p}{2}} \times \frac{1}{\Gamma(\frac{p}{2}+1)}$.

Out of the $n$ points in the hypercube, the fraction of points contained in the hypersphere must then be $\frac{V_{S_p}}{V_{C_p}} = \pi^{\frac{p}{2}} \times \frac{1}{\Gamma(\frac{p}{2}+1)} \div 2^p = \frac{\pi^{\frac{p}{2}}}{2^p \times \Gamma(\frac{p}{2}+1)}$.

[2]:
```
percentage_in_sphere = []
for p in range(2,12):
    percentage_in_sphere.insert(p,(math.pow(math.pi,p/2) / (math.gamma(p/
 ↪2+1)*(math.pow(2,p)))))

plt.figure(figsize=(12,5))
plt.plot(range(2,12),percentage_in_sphere,'o')
plt.title("Relationship between fraction of points in hypersphere and␣
 ↪dimensionality")
plt.xlabel("p dimensions")
plt.ylabel("Fraction of points inside hypersphere")
plt.show()
```




As shown from the graph above, as $p$ increases, the fraction of points inside the hypersphere decreases dramatically. That is, given a set of $n$ points drawn from the $p$-dimensional hypercube, fewer and fewer points remain in the inscribed hypersphere as the dimensionality of both the hypersphere and hypercube increases.

## 1.2 Part 2

We are now going to verify that these theoretical predictions work in practice using Python. Using the command numpy.random.uniform(), generate 100 points uniformly at random in the 2-dimensional hypercube (i.e., $C_2$ is just a square).

### 1.2.1 Solution

We take a square of side length 2, centered at 0, with x-axis support from -1 to 1 and y-axis support from -1 to 1, both from a uniform distribution.

[3]:
```
np.random.seed(seed = 8)
C_2_x = np.random.uniform(-1,1,100)
C_2_y = np.random.uniform(-1,1,100)
```

## 1.3 Part 3

Using the above cloud of 100 points, calculate the fraction of points inside $S_2$ (i.e., the unit circle) with respect to the total number of points inside $C_2$.

### 1.3.1 Solution

To compute the fraction of points inside $S_2$ from the cloud of points generated from $C_2$, $|C_2|^2$ must be less than $|S_2|^2$. In general, $C_p$ is a $n$-length set of $p$-length coordinates: $C_2 = \{(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)\}$. Therefore, $|C_p|^2$ is the sum of the square of each coordinate per realization of history.

[4]:
```
C_2 = ((C_2_x)**2)+((C_2_y)**2)
S_2 = 1**2

print("There are " + str(sum(C_2<S_2)) + " points inside S2 from C2.")

frac_2 = sum(C_2<S_2) / 100
```

```
There are 82 points inside S2 from C2.
```

## 1.4 Part 4

Now repeat (2) and (3) with $p$ going from 2 to 12, and plot the resulting fraction of points as a function of $p$. What do you see? How does it compare to the theoretical result from (1)? Try repeating with more than 100 samples.

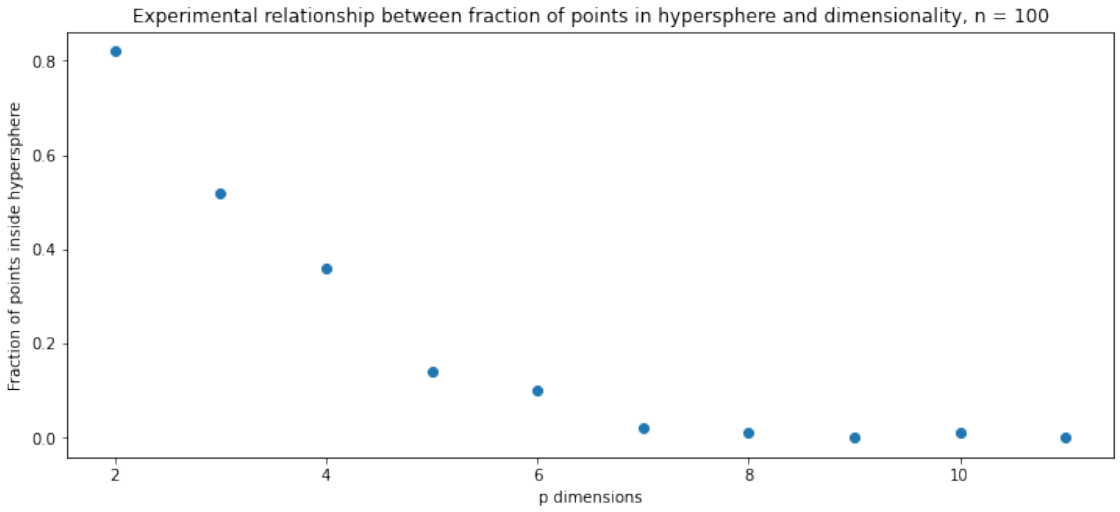
### 1.4.1 Solution

[5]:
```
def experiment(n=100):

    np.random.seed(seed = 8)
    p = 12

    C_component = []
    for i in range(p-2):
        C_component.append([0]*p)

    C = np.zeros((p-2,n)) #initialization
    frac = np.zeros(p-2) #initialization

    for x in range(2,p):
        for i in range(x):
            C_component[x-2][i] = np.random.uniform(-1,1,n) #generate each␣
↪component of C_p uniformly
```

```
        for j in range(x):
            C[x-2] += ((C_component[x-2][j])**2)
        frac[x-2] = sum(C[x-2]<(1**2)) / n

    print("Fraction of points: ", frac)

    plt.figure(figsize=(12,5))
    plt.plot(range(2,12),frac,'o')
    plt.title("Experimental relationship between fraction of points in␣
↪hypersphere and dimensionality, n = " + str(n))
    plt.xlabel("p dimensions")
    plt.ylabel("Fraction of points inside hypersphere")
    plt.show()

    return C_component
```

[6]:
```
C_component = experiment(100)
```

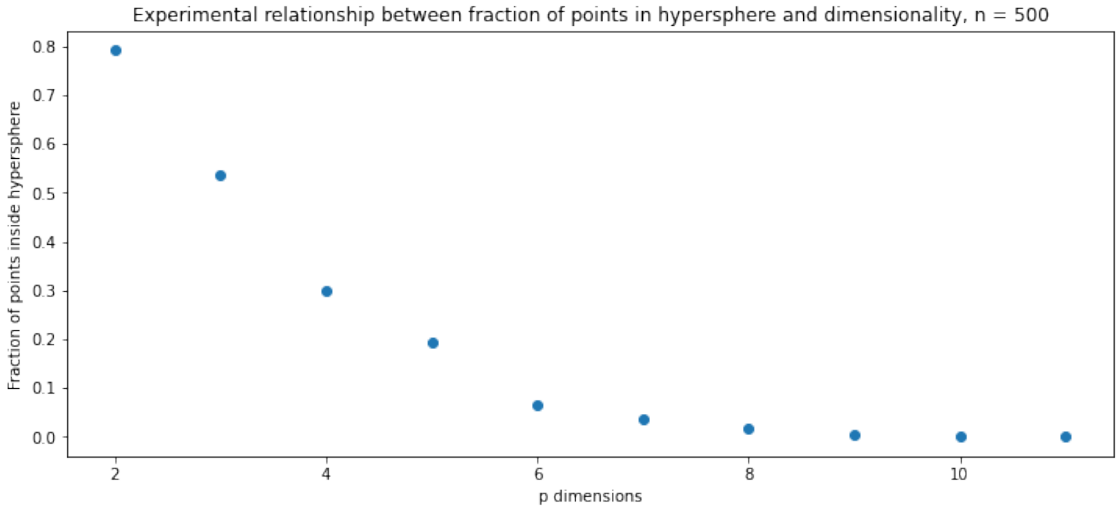
```
Fraction of points:  [0.82 0.52 0.36 0.14 0.1  0.02 0.01 0.   0.01 0.  ]
```




The result from above is roughly similar to that in Part 1. For $p = 2$, the first result is still roughly 80% and the fraction diminishes quickly as seen with the theoretical plot. Notice that at $p = 9$, there is a little bump above the expected relationship. We will now try $n = 500$, $n = 1000$, and $n = 10000$.

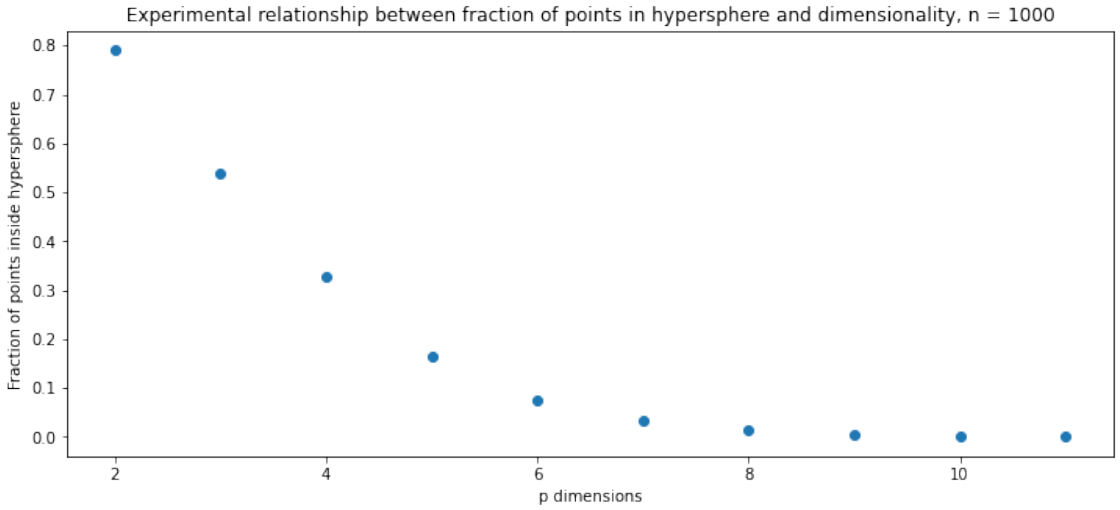
[7]:
```
C_component = experiment(500)
```

```
Fraction of points:  [0.792 0.538 0.298 0.194 0.066 0.038 0.016 0.006 0.    0.
]
```



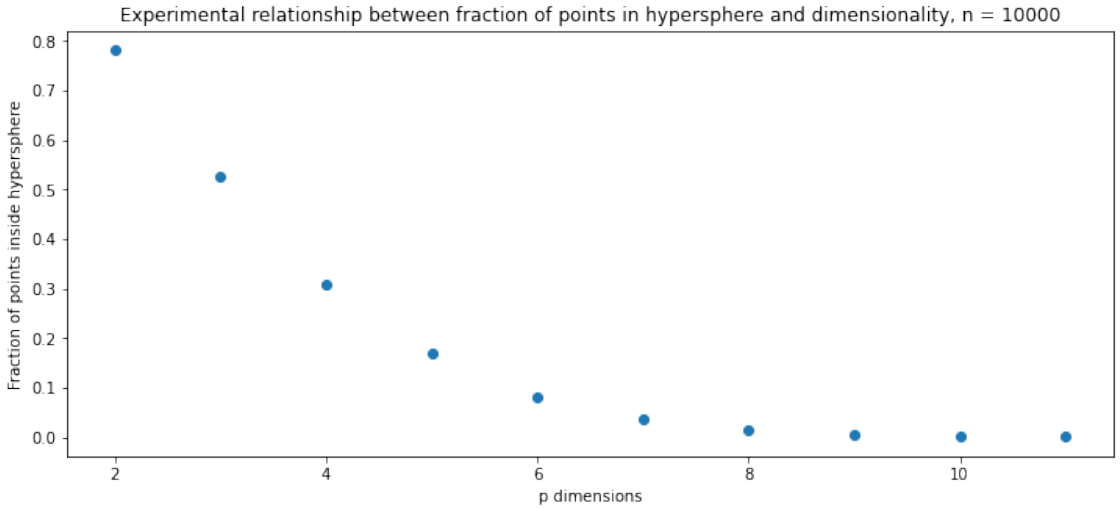

[8]:
```
C_component = experiment(1000)
```

```
Fraction of points:  [0.79  0.538 0.327 0.165 0.076 0.034 0.013 0.006 0.003 0.
]
```




[9]:
```
C_component = experiment(10000)
```

```
Fraction of points:  [0.7809 0.5249 0.3072 0.1694 0.0817 0.0375 0.0159 0.0061
0.0017 0.001 ]
```




As $n$ increases, the experimental relationship becomes closer to that of the theoretical. However, there the relationship is still not as pronounced as the theoretical one, as there is still noise in these random draws. Still, as dimensionality increases, fewer and fewer points are contained in the hypersphere. The curse of dimensionality takes effect.

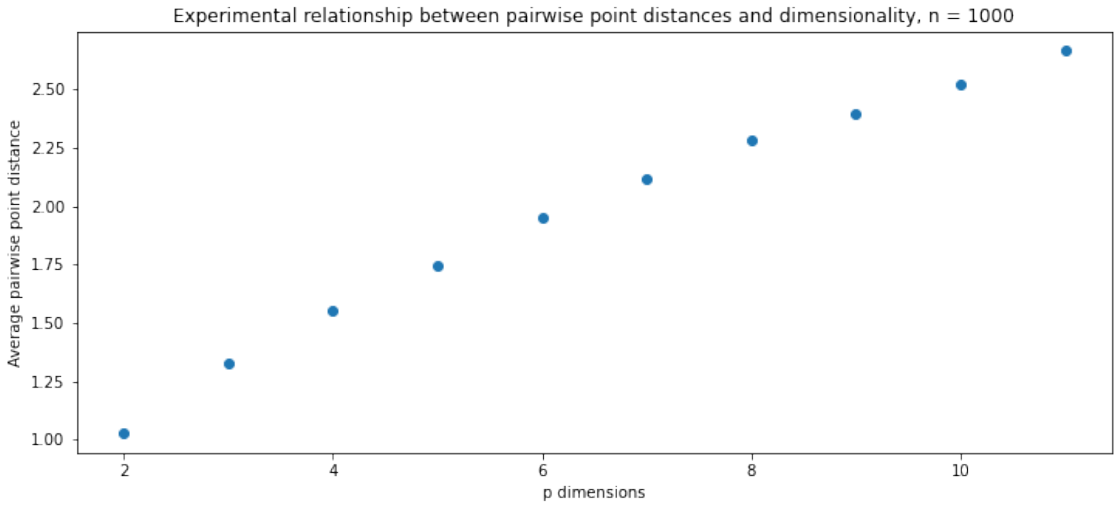
## 1.5 Part 5

Finally, we are going to verify that, as $p$ increases, the distances between pairs of points increase. Using the datasets generated in the previous questions, compute numerically the average pairwise distance among all the points. Plot your result as a function of $p$. What does the curve look like? How does this relate to the results of (4)?

### 1.5.1 Solution

[10]:
```
avg_dist = np.zeros(10)
n = 1000
p = 12

points = []
for i in range(p-2):
    points.append([0]*n)

#Rearrange the data
for x in range(10):
    for j in range(0,n):
        set_of_points = np.zeros(2+x)
        for i in range(0,2+x): #first must be a tuple
            #print(set_of_points[i])
            set_of_points[i] = C_component[x][i][j]
        points[x][j] = set_of_points
```

```
for x in range(10):
    avg_dist[x] = np.mean(pdist(points[x],'euclidean')) #cdist gives the␣
 ↪pairwise distance between pairs of points, in matrix form

plt.figure(figsize=(12,5))
plt.plot(range(2,12),avg_dist,'o')
plt.title("Experimental relationship between pairwise point distances and␣
 ↪dimensionality, n = 1000")
plt.xlabel("p dimensions")
plt.ylabel("Average pairwise point distance")
plt.show()
```




As seen from the graph above, as the dimensionality increases, the pairwise distances increase. This means that as dimensionality increases, points get farther and farther from each other, making it more unlikely for points to be near the spherical center.