Peer Assessments (https://class.coursera.org/algorithmicthink1-003/human_grading/)
/ Application 1 - Analysis of citation graphs
Help Center (https://accounts.coursera.org/i/zendesk/courserahelp?return_to=https://learner.coursera.help/hc)

Submission Phase

1. Do assignment ☑ (/algorithmicthink1-003/human_grading/view/courses/975649/assessments/31/submissions)

Evaluation Phase

2. Evaluate peers ☑ (/algorithmicthink1-003/human_grading/view/courses/975649/assessments/31/peerGradingSets)

Results Phase

3. See results ☑ (/algorithmicthink1-003/human_grading/view/courses/975649/assessments/31/results/mine)

Your effective grade is 12.5

Your unadjusted grade is 12.5, which is simply the grade you received from your peers.

See below for details.

Overview

In the Module 1 Application, we will combine the mathematical analysis that we began in the Homework with the code that you have written in the Project to analyze a real-world problem: How do scientific papers get cited? This part of the module will probably be much more unstructured than you are accustomed to in an on-line class. Our goal is to provide a more realistic simulation of how the concepts that you are learning are actually used in practice. Your key task in this part of the module is to **think** about the problem at hand as you answer each question.

As part of this portion of the module, you'll need to write code that processes medium-sized datasets. You are welcome to use either desktop Python or CodeSkulptor when writing this code. To process the data in CodeSkulptor, you will need to be careful in how you implement your code and will probably need to increase the default timeout from 5 secs to around 20-30 secs. You can reset the timeout using:

```
import codeskulptor
codeskulptor.set_timeout(20)
```

Citation graphs

Our task for this application is to analyze the structure of graphs generated by citation patterns from scientific papers. Each scientific paper cites many other papers, say 20-40, and sometimes (e.g., review papers) hundreds of other papers. But, let's face it: It is often the case that the authors of a paper are superficially familiar with some (many?) of the papers they cite. So, the question is: Are the cited papers chosen randomly (from within the domain of the paper) or is there some "hidden pattern"?

Given that we will be looking at "paper i cites paper j" relationships, it makes sense to represent the citation data as a **directed graph** (a citation graph) in which the nodes correspond to papers, and there is an edge from node *i* to node *j* if the paper corresponding to node *i* cites the paper corresponding to node *j*. Since we're interested in understanding how papers get cited, we will analyze the in-degree distribution of a specific graph, and contrast it to those of graphs generated by two different random processes.

Important: Please use Coursera's "Attach a file" button to attach your plots/images for this Application as required. For each question you can attach more than one image as well as including text and math (LaTeX) in the same answer box. In particular, please do not host your solution plots/images on 3rd party sites. This practice exposes your peers to extra security risks and has the potential for abuse since the contents of a link to an external site may be modified after the hard deadline. Failure to follow this policy may lead to your plots/images being counted as "not submitted".

Question 1 (4 pts)

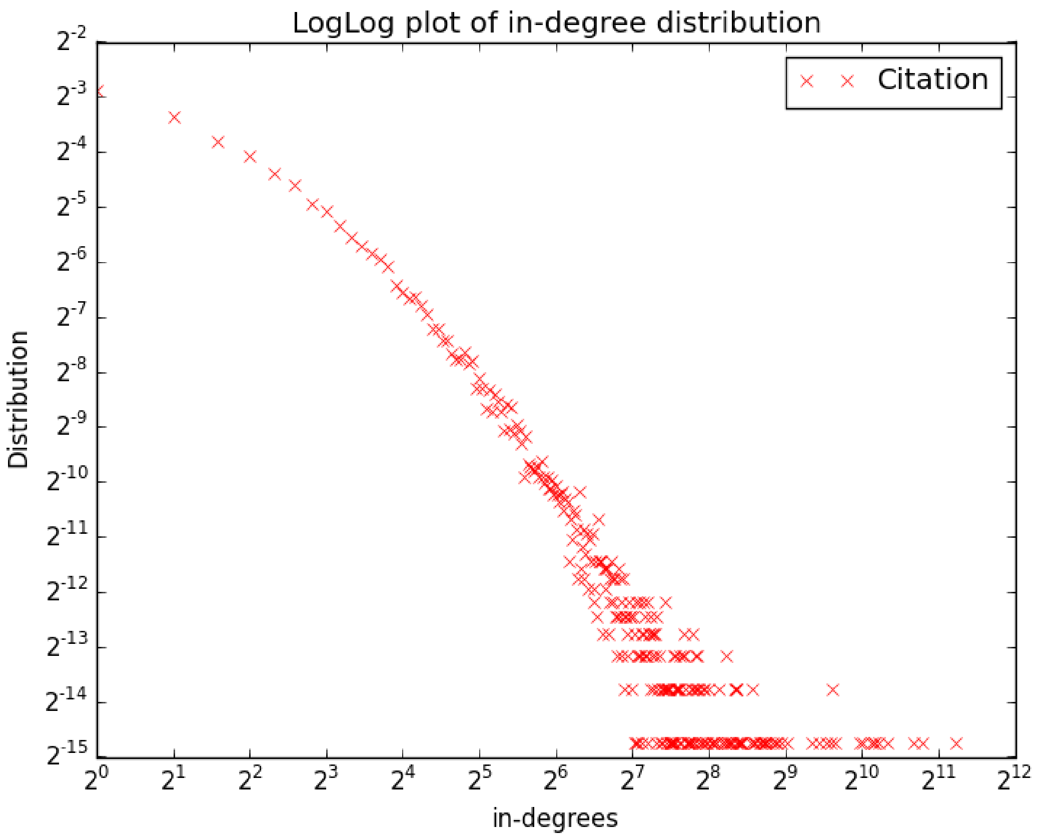
For this question, your task is to load a provided citation graph for 27,770 high energy physics theory papers. This graph has 352,768 edges. You should use the following code (http://www.codeskulptor.org/#alg_load_graph.py) to load the citation graph as a dictionary. In CodeSkulptor, loading the graph should take 5-10 seconds. (For an extra challenge, you are welcome to write your own function to create the citation graph by parsing this text representation (http://storage.googleapis.com/codeskulptor-alg/alg_phys-cite.txt) of the citation graph.)

Your task for this question is to compute the in-degree distribution for this citation graph. Once you have computed this distribution, you should normalize the distribution (make the values in the dictionary sum to one) and then compute a log/log plot of the **points** in this normalized distribution. How you create this point plot is up to you. You are welcome to use a package such as `matplotlib` for desktop Python, use the `simpleplot` module in CodeSkulptor, or use any other method that you wish. This class page (https://class.coursera.org/algorithmicthink1-003/wiki/ides?page=plotting) on "Creating, formatting, and comparing plots" gives an overview of some of the options that we recommend for creating plots.

Since `simpleplot` does not support direct log/log plotting, you may simulate log/log plotting as shown in this example (http://www.codeskulptor.org/#poc_mystery_plot.py) from the PoC video on "Plotting data" (https://class.coursera.org/algorithmicthink1-003/lecture/185). However, be sure to include an indication on the labels for the horizontal and vertical axes that you are plotting the log of the values and note the base that you are using. (Nodes with in-degree zero can be ignored when computing the log/log plot since $log(0) = -\infty$.)

Once you have created your plot, upload your plot in the box below using "Attach a file" button (the button is disabled under the 'html' edit mode; you must be under the 'Rich' edit mode for the button to be enabled). Please review the class guidelines for formatting and comparing plots on the "Creating, formatting, and comparing plots" class page. These guidelines cover the basics of good formatting practices for plots. Your plot will be assessed based on the answers to the following three questions:

- Does the plot follow the formatting guidelines for plots?
- Is the plot that of a normalized distribution on a log/log scale?
- Is the content of the plot correct?



Evaluation/feedback on the above work

Note: this section can only be filled out during the evaluation phase.

Item a (1 pt) Does the plot follow the formatting guidelines for plots?

The formatting guidelines include the following items:

- The plot is an image and not a text file.
- The plot is appropriately trimmed. Showing the boundary of the plot's window is fine. However, the plot should not include part of the desktop.
- The elements of the plot are of the correct type. Line plots are not the same as point plots.
- Both axes should have tick marks labeled by regularly-spaced coordinate values.
- Both axes have appropriate text labels that describe the quantities being plotted.
- The plot has an appropriate title that describes the content of the plot.

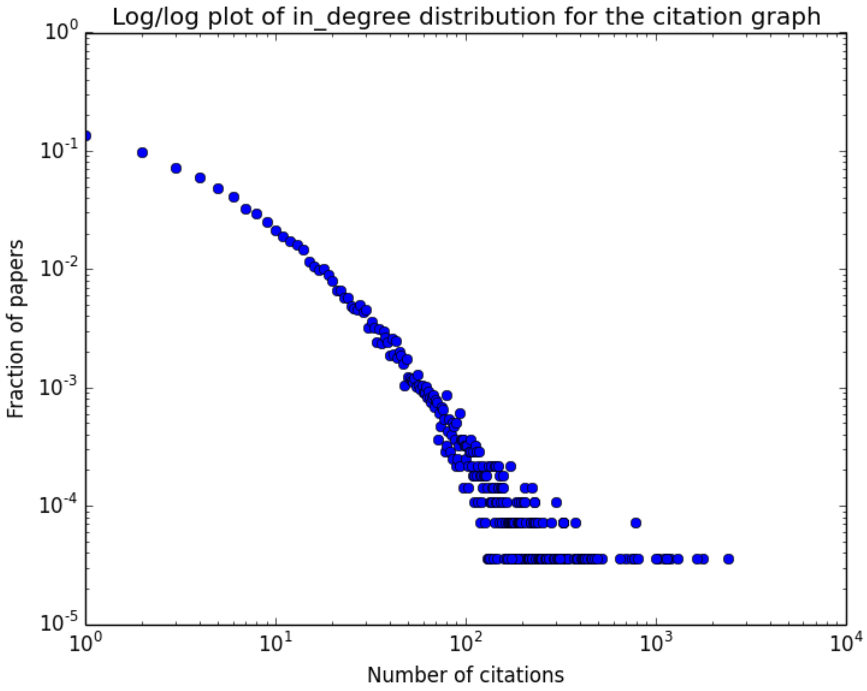
Assess the submitted plot based on these guidelines.

Score from your peers: **1**

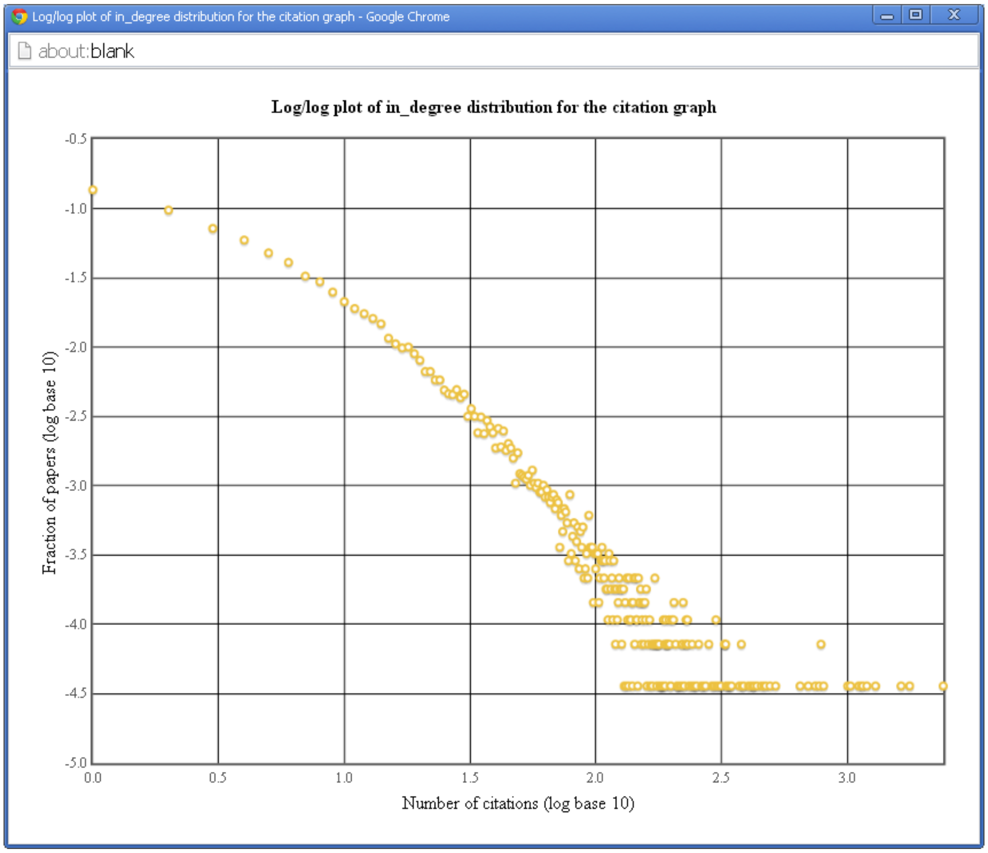
Item b (1 pt) Is the submitted plot that of a normalized distribution on a log/log scale?

Here are several examples of correct and incorrect plots that you should review while assessing this item.

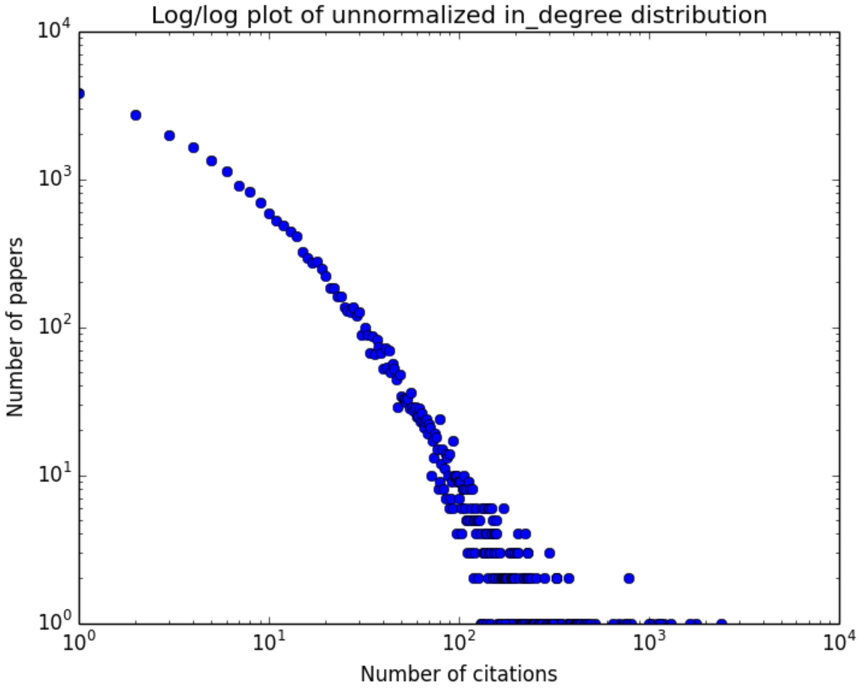
- A **correct** plot of a normalized distribution on a log/log scale using `matplotlib`.



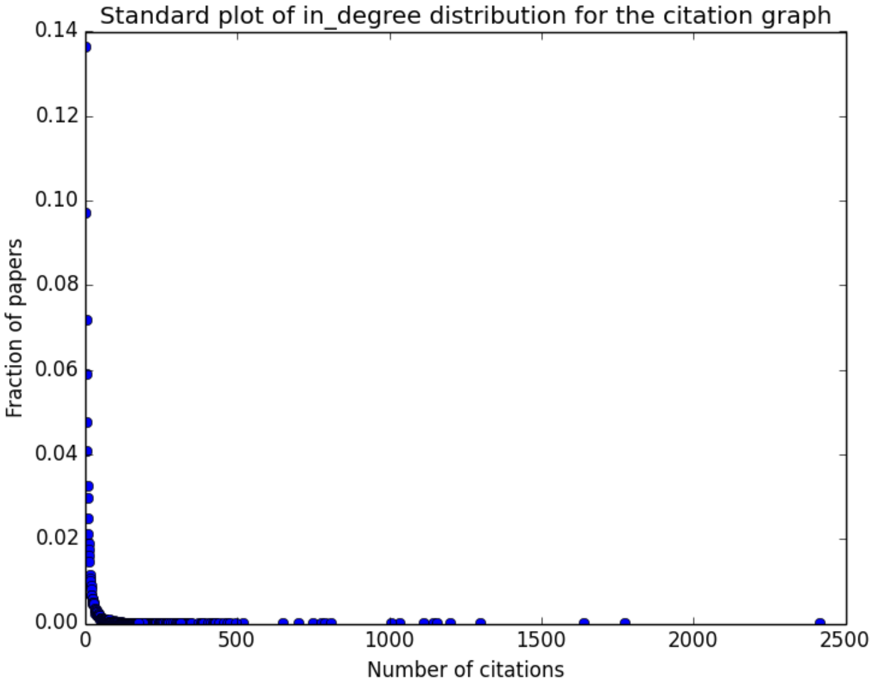
- A **correct** plot of a normalized distribution on simulated log/log scale using `simpleplot`. Note that the `log` may be computed using base 10 as in the figure below or using the another base such as 2 or `e`. For base 2, the horizontal axis would range from 0 to 12. For base `e`, the horizontal axis would range from 0 to 8.


Log/log plot of in_degree distribution for the citation graph - Google Chrome
about:blank
Log/log plot of in_degree distribution for the citation graph
-0.5
-1.0
-1.5
-2.0
-2.5
-3.0
-3.5
-4.0
-4.5
-5.0
Fraction of papers (log base 10)
0.0
0.5
1.0
1.5
2.0
2.5
3.0
Number of citations (log base 10)

- An **incorrect** plot of a distribution that is not normalized. Note that the range of the vertical axis of an unnormalized plot is entirely greater than one while the range of a normalized plot is between zero and one.



- An **incorrect** plot using a linear scale (not log/log). Note the "L" shape of the plot with the range of the horizontal axis being 0 to 2500. (One paper has almost 2500 citations).



Score from your peers: **1**

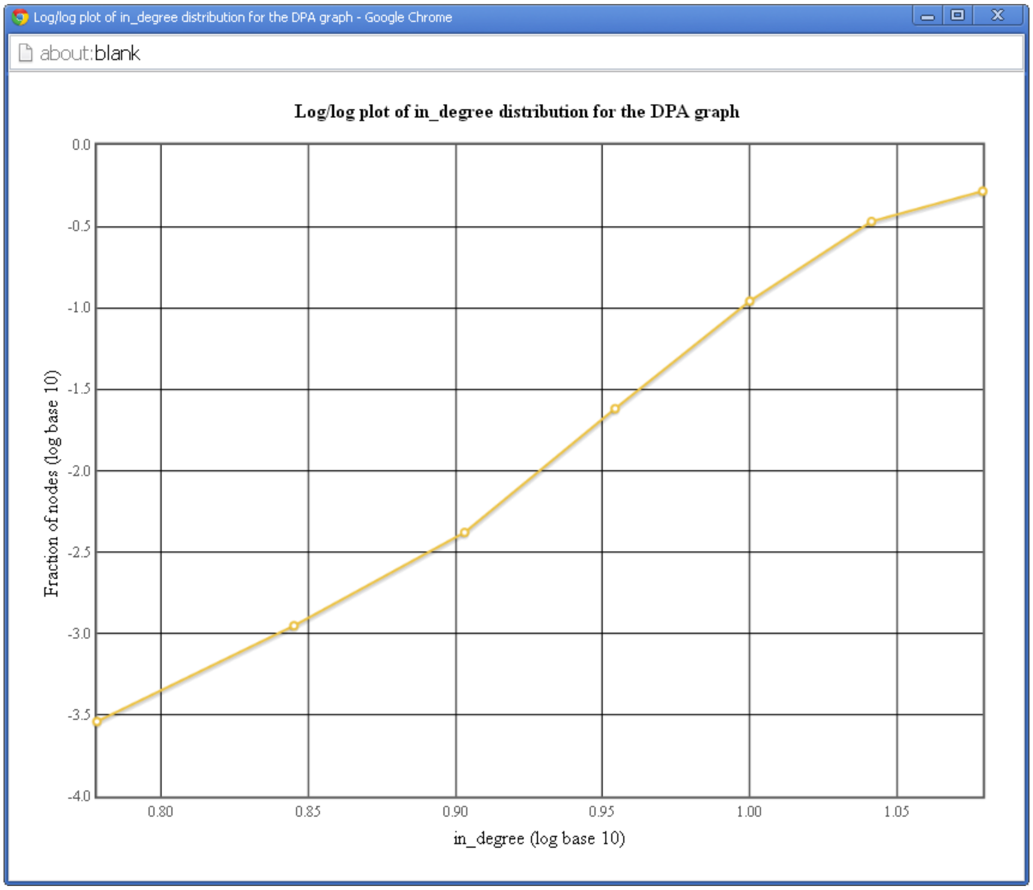
Item c (2 pts) Is the content of the plot correct?

Answer: The first two plots in item b) are correct plots for this question. Please review them.

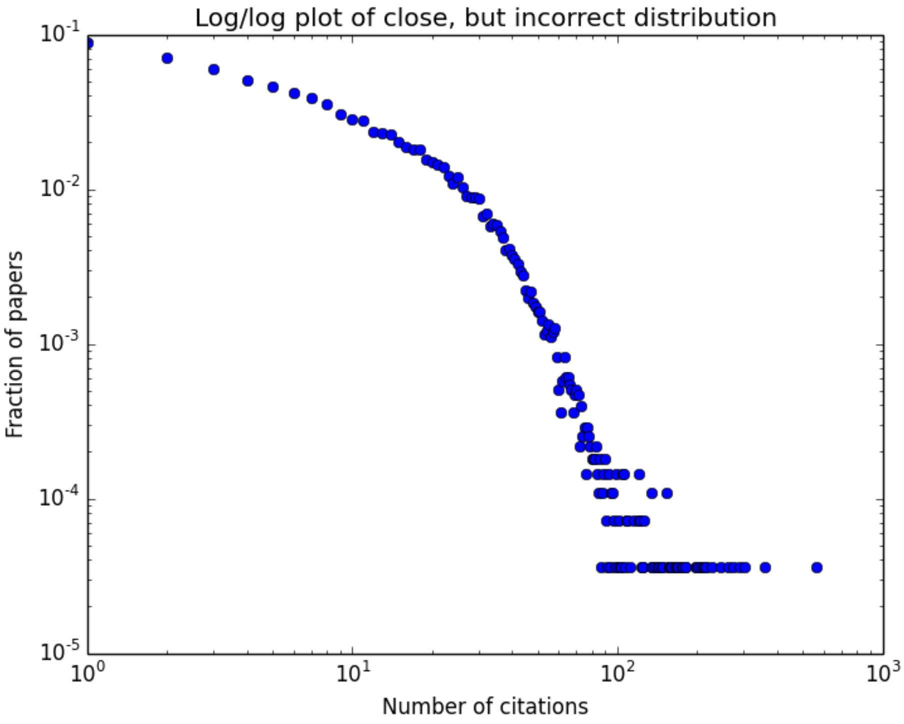
When evaluating whether the submitted plot is correct, compare the submitted plot to the correct plots using the class guidelines for comparing plots. The guidelines include comparing the following features of the plot:

- Compare the number of points/lines in each plot.
- Compare the coherence of the points/lines in each plot. Are the points/lines scattered randomly or can they be approximated by a curve?
- If the points in each plots can be approximated by a curve, compare the shape of the curves. Is one curve linear and the other curve non-linear?
- If the points in each plots can be approximated by a line, compare the slopes of the lines. Is one line rising and the other line falling?

To aid in your assessment, here are two incorrect plots. The first plot below should receive no credit since it is dissimilar from the correct plot. The number of plotted points is substantially different and their trend is rising.



The second plot below should receive 1 pt credit since it is similar in many ways to the correct plot. For example, the spread of the points increases as the fraction of nodes decreases. However, the points in this plot "bend" significantly while the correct plot can be approximated fairly accurately by a line.



Score from your peers: **2**

Comments: Please enter an explanation for your scoring, especially if you deducted any points for one of the rubric items for this question.

peer 1 → *[This area was left blank by the evaluator.]*

peer 2 → *[This area was left blank by the evaluator.]*

peer 3 → *[This area was left blank by the evaluator.]*

peer 4 → *[This area was left blank by the evaluator.]*

Question 2 (3 pts)

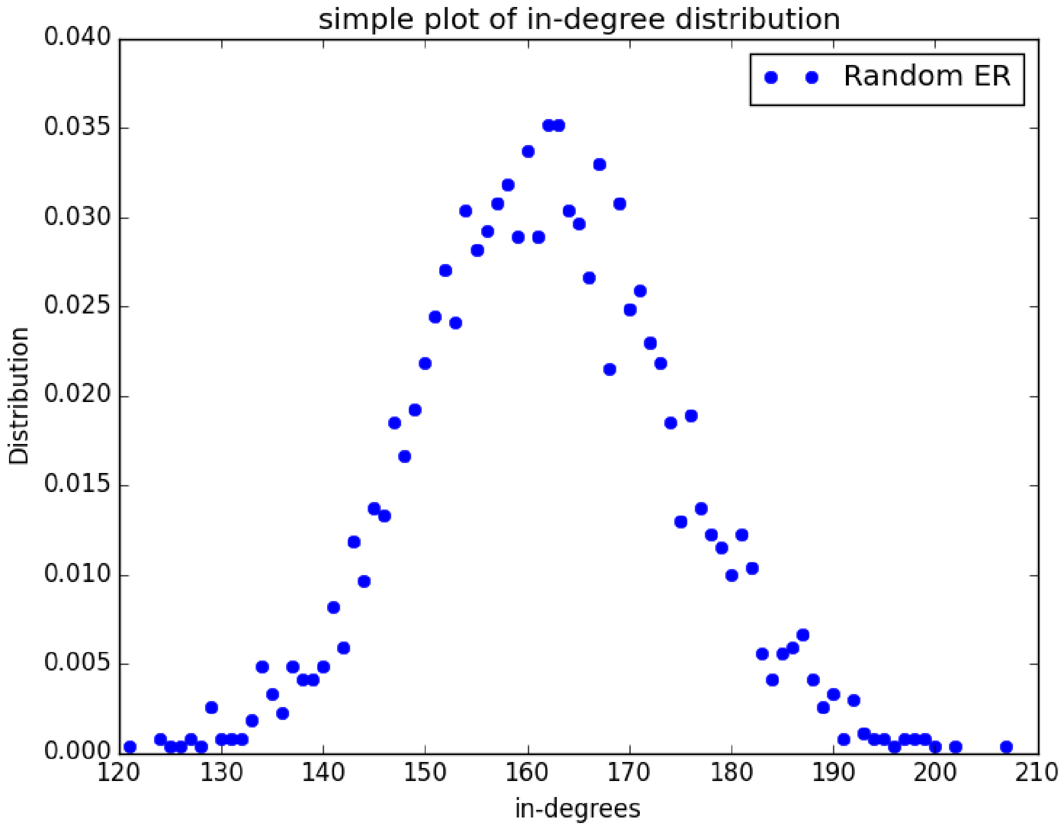
In Homework 1, you saw Algorithm **ER** for generating random graphs and reasoned analytically about the properties of the ER graphs it generates. Consider the simple modification of the algorithm to generate random *directed* graphs: For every ordered pair of distinct nodes (i, j), the modified algorithm adds the directed edge from i to j with probability p.

For this question, your task is to consider the shape of the in-degree distribution for an ER graph and compare its shape to that of the physics citation graph. In the homework, we considered the probability of a specific in-degree, k, for a single node. Now, we are interested in the in-degree distribution for the entire ER graph. To determine the shape of this distribution, you are welcome to compute several examples of in-degree distributions or determine the shape mathematically.

Once you have determined the shape of the in-degree distributions for ER graphs, compare the shape of this distribution to the shape of the in-degree distribution for the citation graph. When answering this question, make sure to address the following points:

- Is the expected in-degree the same for every node in an ER graph? Please answer yes or no and include a short explanation for your answer.
- What does the in-degree distribution for an ER graph look like? You may either provide a plot (linear or log/log) of the degree distribution for a small value of n or a short written description of the shape of the distribution.
- Does the shape of the in-degree distribution plot for ER look similar to the shape of the in-degree distribution for the citation graph? Provide a short explanation of the similarities or differences. Focus on comparing the shape of the two plots as discussed in the class page on "Creating, formatting, and comparing plots".

Yes, the expected in-degree is same for a given probability. This is because of uniform random distribution.
It looks like a Normal (Gaussian) bell curve.



The shape of random ER graph is different from citation graph.

Evaluation/feedback on the above work

Note: this section can only be filled out during the evaluation phase.

Item a (1 pt) Is the expected in-degree the same for each node in an ER graph? Please answer yes or no and include a short explanation for your answer.

Answer: Yes. The ER algorithm treats each node in the graph in the same manner. So the expected in-degree for each node must be the same. In fact, the in-degree distribution is the same for each node.

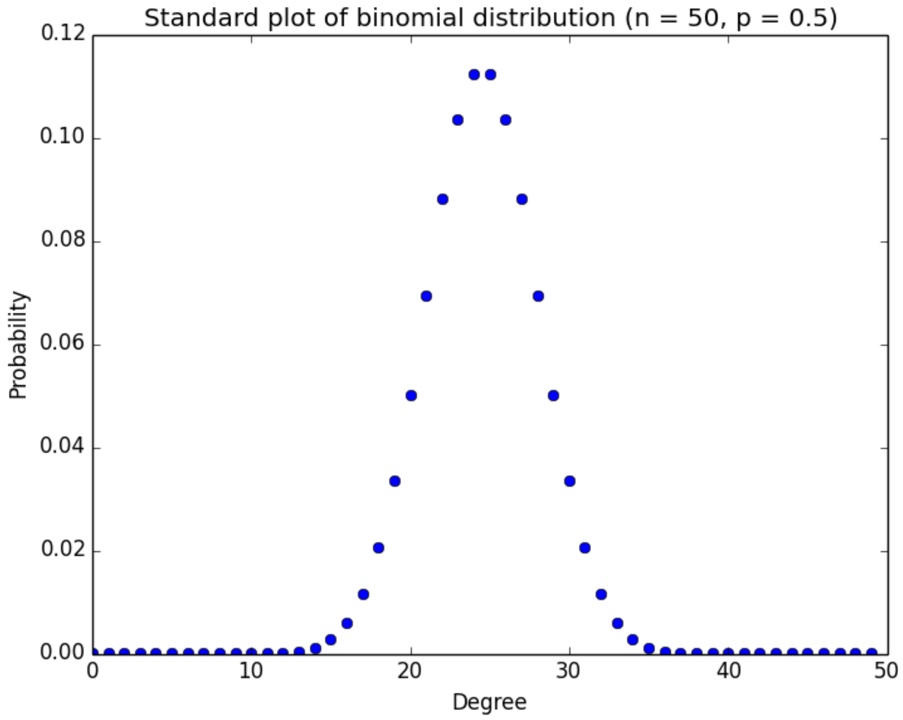
Score from your peers: **1**

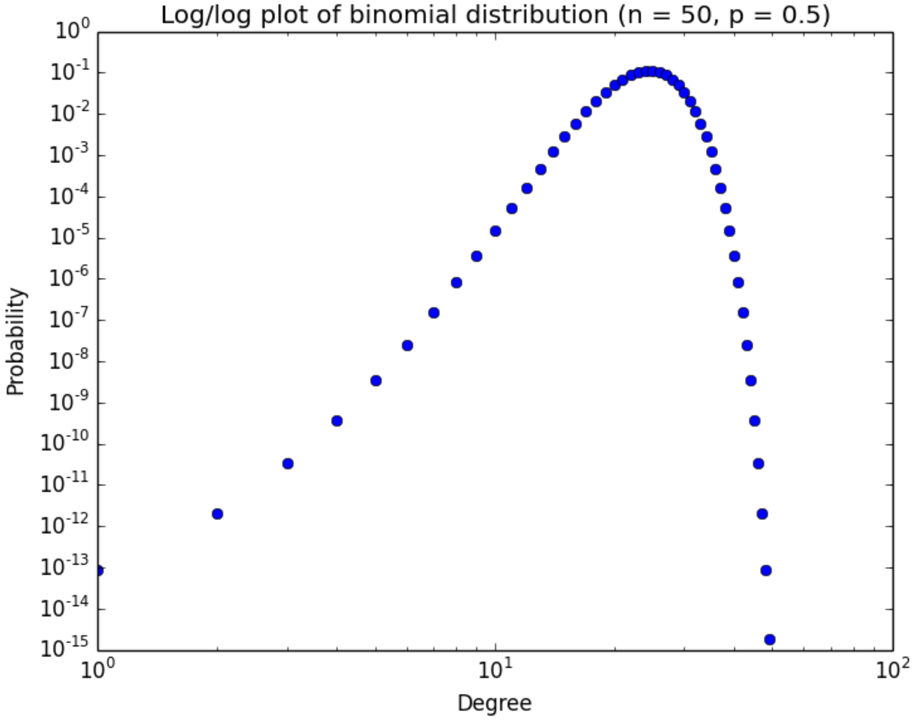
Item b (1 pt) What does the in-degree distribution for an ER graph look like? You may either provide a plot (linear or log/log) of the degree distribution for a small value of n or a short written description of the shape of the distribution.

Answer: As noted in the Homework, the in-degree of an arbitrary node in a directed ER graph is binomially distributed. Therefore, the in-degree distribution for the entire graph must also be binomial.

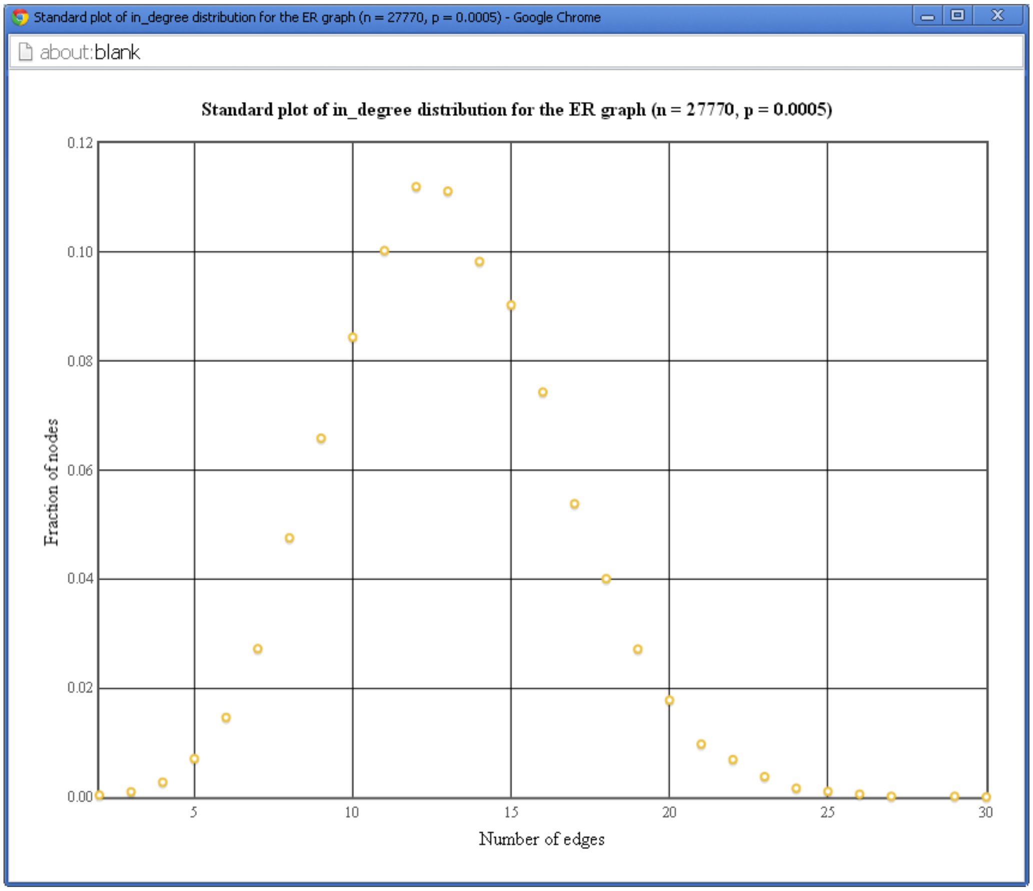
This distribution is often described as "bump-shaped" or "bell-shaped". Correct descriptions of the shape may include terms like "binomial distribution", "normal distribution", "Gaussian", "Poisson distribution", even though it is important to note that normal (Gaussian) is a continuous distribution. (Note that correct answers don't need to include these terms though.) For those submitting a plot of this distribution, here are two plots of the binomial distribution for $n = 50$ and $p = 0.5$. The left plot is in linear scale while the right plot is in log/log scale.

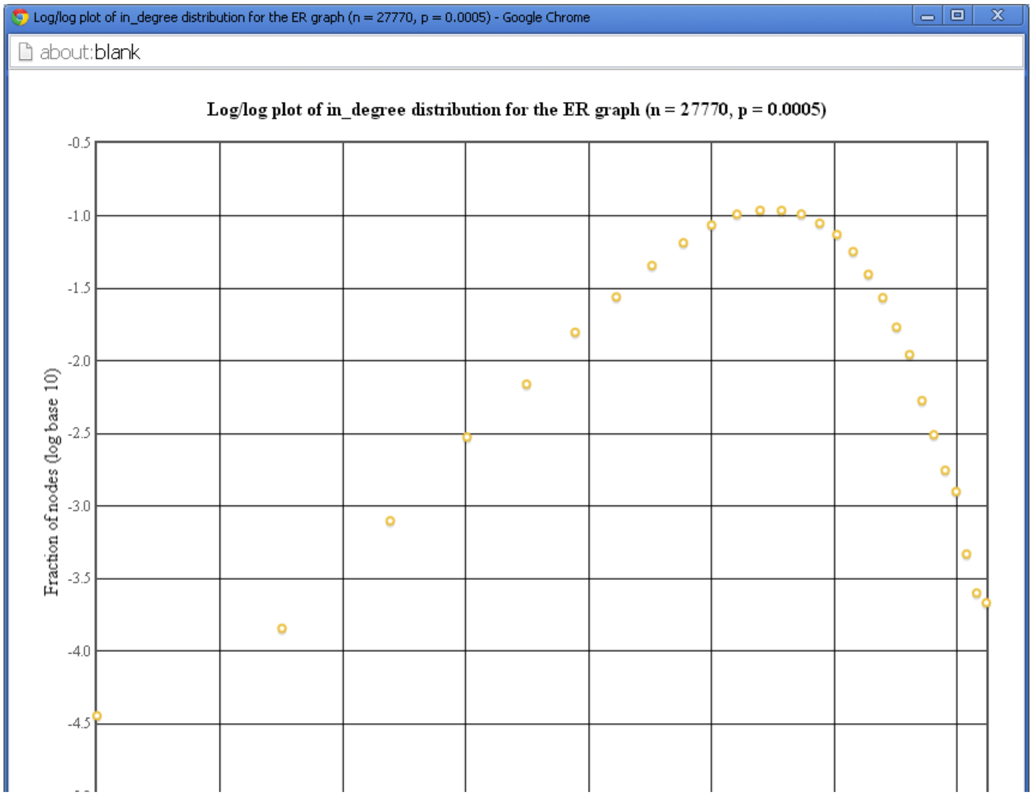
Note that the points in the bump-shaped curve may be rather noisy if the number of trials used in the generating the plot is small.

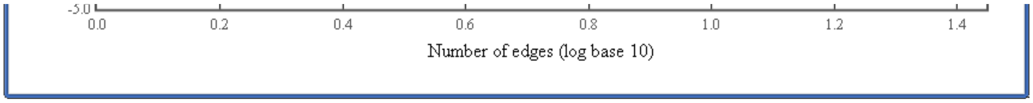




The in-degree distributions for an ER graph with $n = 27770$ and $m = 13$ has a similar binomial shape (with $p \ = 0.0005)$ as shown by the plots below:


Standard plot of in_degree distribution for the ER graph (n = 27770, p = 0.0005) - Google Chrome
about:blank
Standard plot of in_degree distribution for the ER graph (n = 27770, p = 0.0005)
Fraction of nodes
0.12
0.10
0.08
0.06
0.04
0.02
0.00
5
10
15
20
25
30
Number of edges


Log/log plot of in_degree distribution for the ER graph (n = 27770, p = 0.0005) - Google Chrome
about:blank
Log/log plot of in_degree distribution for the ER graph (n = 27770, p = 0.0005)
Fraction of nodes (log base 10)
-0.5
-1.0
-1.5
-2.0
-2.5
-3.0
-3.5
-4.0
-4.5



Evaluate the correctness of the submitted plot (if one is provided) versus one of these four plots using the criteria given in rubric item c on Question 1. Give full credit if the general trend of the provided plot is bump-shaped.

Score from your peers: **1**

Item c (1 pt) Does the shape of the in-degree distribution plot for ER look similar to the shape of the in-degree distribution for the citation graph? Provide a short explanation of the similarities or differences. Focus on comparing the shape of the two plots as discussed in the class page on "Creating, formatting, and comparing plots".

No. The two distributions are not similar. The in-degree distributions for ER graphs are binomial (bump-shaped) and are substantially different from the linear shape of in-degree distributions for citation graphs.

Score from your peers: **0.5**

Conclusion: Citation graphs are not generated by a purely random process. If they were, we would expect the in-degree distribution for the citation graph to be similar to the in-degree distribution for the ER graphs. However, the distributions for ER graphs are binomial (bump-shaped) while the distribution for the citation graph is almost linear.

Comments: Please enter an explanation for your scoring, especially if you deducted any points for one of the rubric items for this question.

peer 1 → *[This area was left blank by the evaluator.]*

peer 2 → No explanation provided for the differences between the graphs.

peer 3 → *[This area was left blank by the evaluator.]*

peer 4 → *[This area was left blank by the evaluator.]*

Question 3 (2 pts)

We next consider a different process for generating synthetic directed graphs. In this process, a random directed graph is generated iteratively, where in each iteration a new node is created, added to the graph, and connected to a subset of the existing nodes. This subset is chosen based on the in-degrees of the existing nodes. More formally, to generate a random directed graph in this process, the user must specify two parameters: n, which is the final number of nodes, and m (where $m \leq n$), which is the number of existing nodes to which a new node is connected during each iteration. Notice that m is fixed throughout the procedure.

The algorithm starts by creating a *complete directed graph* on m nodes. (Note, you've already written the code for this part in the Project.) Then, the algorithm grows the graph by adding $n - m$ nodes, where each new node is connected to m nodes randomly chosen from the set of existing nodes. As an existing node may be chosen more than once in an iteration, we eliminate duplicates (to avoid *parallel edges*); hence, the new node may be connected to fewer than m existing nodes upon its addition.

The full description of the algorithm for generating random directed graphs with this process is given below, and is called Algorithm DPA (note that the m in the input is a parameter that is specified to this algorithm, and it does not denote the total number of edges in the resulting graph). The notation $\sum_{x \in S} x$ means the "sum of all elements x in set S." For example, if $S = \{1, 7, 12\}$, then $\sum_{x \in S} x \equiv 1 + 7 + 12 = 20$.

Algorithm 3: DPA.

```
Input: Number of nodes $n$ ($n \geq 1$); integer $m$ ($1 \leq m \leq n$).
Output: A directed graph $g = (V, E)$.
$V \leftarrow \{0, 1, \ldots, m-1\}$;                              // Start a graph on $m$ nodes
$E \leftarrow \{(i, j) : i, j \in V, i \neq j\}$;                  // Make the graph complete
for $i \leftarrow m$ to $n - 1$ do
    $totindeg = \sum_{j \in V} indeg(j)$;                // sum of the in-degrees of existing nodes
    $V' \leftarrow \emptyset$;
    Choose randomly $m$ nodes from $V$ and add them to $V'$, where the probability of choosing node $j$ is
    $(indeg(j) + 1)/(totindeg + |V|)$;        // The $m$ nodes may not be distinct; hence $|V'| \leq m$
    $V \leftarrow V \cup \{i\}$;                                   // new node $i$ is added to set $V$
    $E \leftarrow E \cup \{(i, j) : j \in V'\}$;           // connect the new node to the randomly chosen nodes
return $g = (V, E)$;
```

Notice that this algorithm is more complex than the **ER** algorithm. As a result, reasoning about the properties of the graphs that it generates analytically is not as simple. When such a scenario arises, we can implement the algorithm, run it, produce graphs, and visually inspect their in-degree distributions. In general, this is a powerful technique: When analytical solutions to systems are very hard to derive, we can simulate the systems and generate data that can be analyzed to understand the properties of the systems.

For this question, we will choose values for n and m that yield a DPA graph whose number of nodes and edges is roughly the same to those of the citation graph. For the nodes, choosing n to be the number of nodes as the citation graph is easy. Since each step in the DPA algorithm adds m edges to the graph, a good choice for m is an integer that is close to the average out-degree of the physics citation graph.

For this question, provide numerical values for n and m that you will use in your construction of the DPA graph.

n = 27770
m = 14

Evaluation/feedback on the above work

Note: this section can only be filled out during the evaluation phase.

Item a (1 pt) What is the number of nodes n in the desired DPA graph?

The value of n is the number of papers in the citation graph, which is 27770. Since the text asked for a rough value, score values for n between 27000 and 28000 as being correct.

Score from your peers: **1**

Item b (1 pt) What is the value of m in the desired DPA graph?

The citation graph has 352768 total edges and 27770 total nodes. So the average out-degree is approximately 12.7. Since the value of m must be an integer, the answer may be rounded either up of down to the nearest integer. So, either $m = 12$ or $m = 13$ is an acceptable answer.

The answers $m = 14$ and $m = 15$ should also receive full credit since, in practice, the actual number of edges in the final DPA graph is often closer to 352768 for these values due to the fact that fewer than m edges are sometimes added during each iteration of the algorithm.

Score from your peers: **1**

Comments: Please enter an explanation for your scoring, especially if you deducted any points for one of the rubric items for this question.

peer 1 → *[This area was left blank by the evaluator.]*

peer 2 → *[This area was left blank by the evaluator.]*

peer 3 → *[This area was left blank by the evaluator.]*

peer 4 → *[This area was left blank by the evaluator.]*

Question 4 (3 pts)

Your task for this question is to implement the DPA algorithm, compute a DPA graph using the values from Question 3, and then plot the in-degree distribution for this DPA graph. Creating an efficient implementation of the DPA algorithm from scratch is surprisingly tricky. The key issue in implementing the algorithm is to avoid iterating through every node in the graph when executing Line 6. Using a loop to implement Line 6 leads to implementations that require on the order of 30 minutes in desktop Python to create a DPA graph with 28000 nodes.

To avoid this bottleneck, you are welcome to use this provided code (http://www.codeskulptor.org/#alg_dpa_trial.py) that implements a `DPATrial` class. The class has two methods:

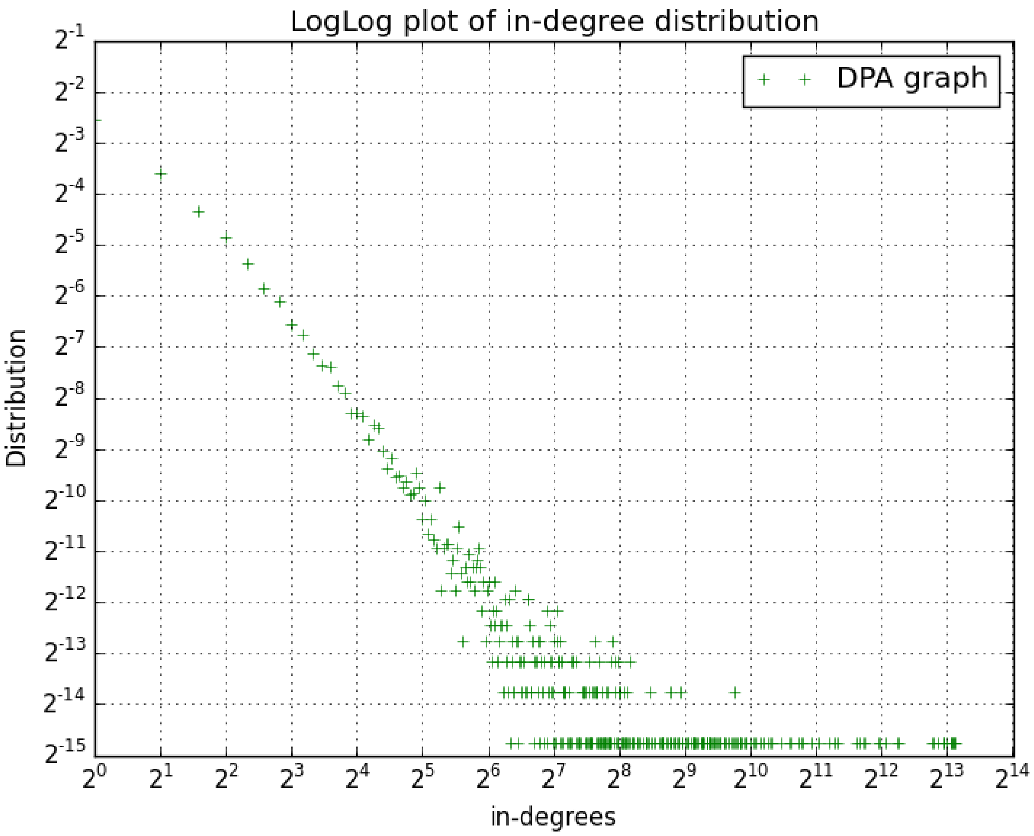
- `__init__(num_nodes):` Create a `DPATrial` object corresponding to a complete graph with `num_nodes` nodes.
- `run_trial(num_nodes):` Runs `num_nodes` number of DPA trials (lines 4- 6). Returns a set of the nodes, computed with the correct probabilities, that are neighbors of the new node.

In the provided code, the `DPATrial` class maintains a list of node numbers that contains multiple instances of the same node number. If the number of instances of each node number is maintained in the same ratio as the desired probabilities, a call to `random.choice()` produces a random node number with the desired probability.

Using this provided code, implementing the DPA algorithm is fairly simple and leads to an efficient implementation of the algorithm. In particular, computing a DPA graph with 28000 nodes should take on the order of 10-20 seconds in CodeSkulptor. For a challenge, you are also welcome to develop your own implementation of the DPA algorithm that does not use this provided code. However, we recommend that you use desktop Python as your development environment since you are likely to encounter long running times.

Once you have created a DPA graph of the appropriate size, compute a (normalized) log/log plot of the **points** in the graph's in-degree distribution, and upload your plot in the box below using the "Attach a file" button. (Note that you do not need to upload or machine-grade your DPA code.) Your submitted plot will be assessed based on the answers to the following three questions:

- Does the plot follow the formatting guidelines for plots?
- Is the plot a log/log plot of a normalized distribution?
- Is the content of the plot correct?



Evaluation/feedback on the above work

Note: this section can only be filled out during the evaluation phase.

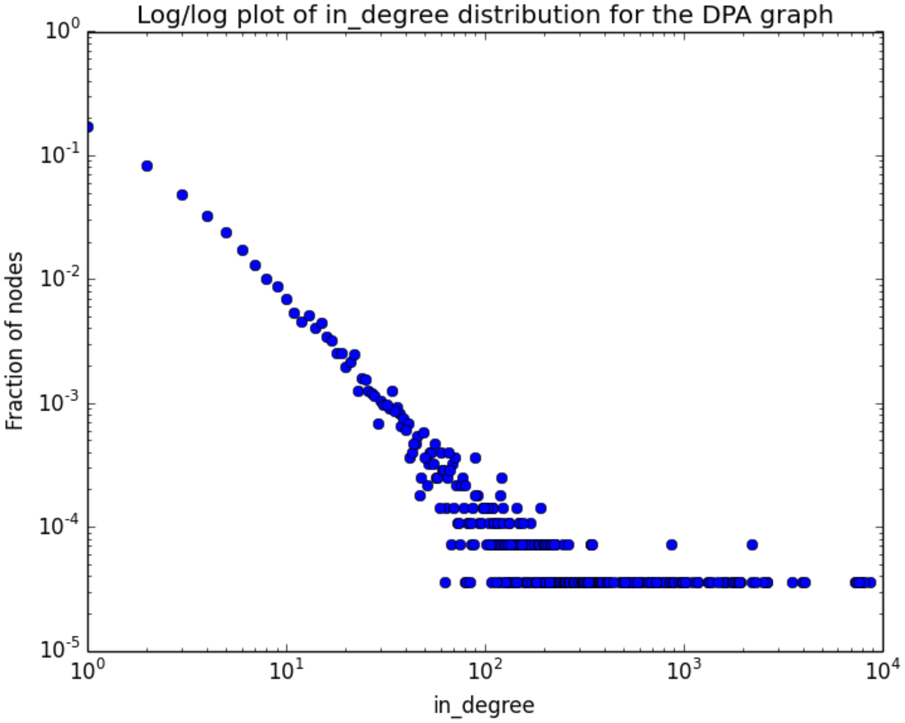
Item a (1 pt) Does the plot follow the formatting guidelines for plots? Is the plot a log/log plot of a normalized distribution?

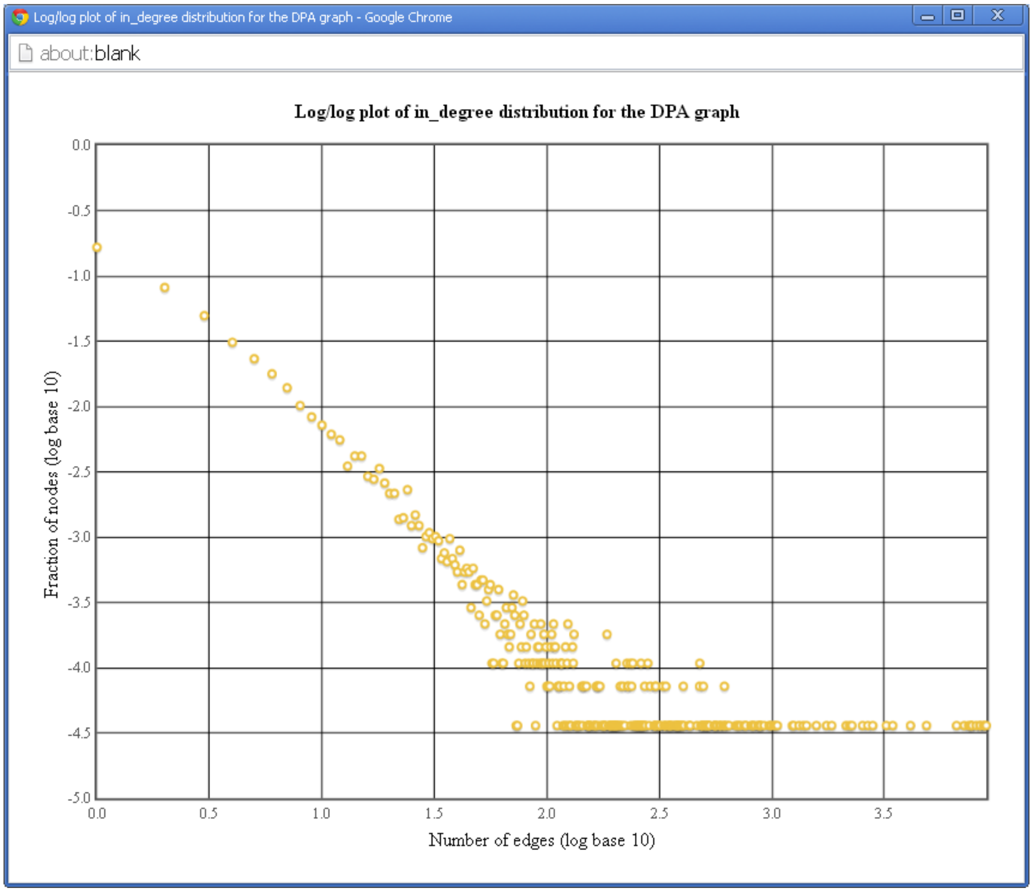
Review the rubrics for items a and b on Question 1 for details on evaluating this item.

Score from your peers: **1**

Item b (2 pts) Is the content of the plot correct?

Below are correct plots for the in-degree distribution of a DPA graph of the appropriate size. The left plot was computed using `matplotlib` while the right plot was computed using `simpleplot` using a simulated log/log scale.




Log/log plot of in_degree distribution for the DPA graph - Google Chrome
about:blank
Log/log plot of in_degree distribution for the DPA graph
Fraction of nodes (log base 10)
Number of edges (log base 10)
0.0
-0.5
-1.0
-1.5
-2.0
-2.5
-3.0
-3.5
-4.0
-4.5
-5.0
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5

Since the DPA algorithm is not deterministic, the submitted plot need not be identical to these plots. In fact, the two plots above have slight differences due to the fact that they were computed from two different randomly-generated DPA graphs. However, the submitted plot should be very similar to these two plots. To assess how close the submitted plot is to the correct plots, use the plot comparison guidelines described in item c from Question 1.

Score from your peers: **2**

Comments: Please enter an explanation for your scoring, especially if you deducted any points for one of the rubric items for this question.

peer 1 → *[This area was left blank by the evaluator.]*

peer 2 → *[This area was left blank by the evaluator.]*

peer 3 → *[This area was left blank by the evaluator.]*

peer 4 → *[This area was left blank by the evaluator.]*

Question 5 (3 pts)

In this last problem, we will compare the in-degree distribution for the citation graph to the in-degree distribution for the DPA graph as constructed in Question 4. In particular, we will consider whether the shape of these two distributions are similar and, if they are similar, what might be the cause of the similarity.

To help you in your analysis, you should consider the following three phenomena:

- The "six degrees of separation" phenomenon,
- The "rich gets richer" phenomenon, and
- The "Hierarchical structure of networks" phenomenon.

If you're not familiar with these phenomena, you can read about them by conducting a simple Google or Wikipedia search. Your task for this problem is to consider how one of these phenomena might explain the structure of the citation graph or, alternatively, how the citations patterns follow one of these phenomena.

When answering this question, please include answers to the following:

- Is the plot of the in-degree distribution for the DPA graph similar to that of the citation graph? Provide a short explanation of the similarities or differences. Focus on the various properties of the two plots as discussed in the class page on "Creating, formatting, and comparing plots".
- Which one of the three social phenomena listed above mimics the behavior of the DPA process? Provide a short explanation for your answer.
- Could one of these phenomena explain the structure of the physics citation graph? Provide a short explanation for your answer.

The plot of in-degree distribution for the DPA graph is similar to that of the citation graph. The DPA graph lies slightly below the citation graph but the shape follows a similar pattern as that of citation graph.

The "Hierarchical structure of networks" mimics the bheavior of DPA process.

The "Hierarchical structure of networks" can provide insights on the physic citation graph as to how the correlated papers are clustered together.

Evaluation/feedback on the above work

Note: this section can only be filled out during the evaluation phase.

Item a (1 pt) Is the plot of the in-degree distribution for the DPA graph similar to that of the citation graph? Provide a short explanation of the similarities or differences. Focus on the various properties of the two plots as discussed in the class page on "Creating, formatting, and comparing plots."

The plot of the in-degree distribution of the DPA graph is indeed similar to that of the citation graph. They agree on all of the items listed in item c for Question 1. In particular, the points in both plots are accurately approximated by a line with falling (negative) slope. In both cases, the points tend to scatter more as the fraction of points (papers) decreases.

Score from your peers: **1**

Item b (1 pt) Which one of the three social phenomena listed above mimics the behavior of the DPA process? Provide a short explanation for your answer.

The correct phenomenon is the "rich gets richer". In Algorithm DPA, a node with a higher degree (rich) has a higher probability of getting a new edge (richer). This process modeled by Algorithm DPA mimics the rich gets richer model, but is also used to *explain* the six degrees of separation phenomenon. Therefore, for this item, also give credit to answers that cite the "six degrees of separation."

Score from your peers: **0**

Item c (1 pt) Could one of these phenomena explain the structure of the physics citation graph? Provide a short explanation for your answer.

The "Rich get richer" phenomenon provides an explanation for the structure of the citation graph. Papers (nodes) that have lots of citations (incoming edges) are more visible and, therefore, more likely to draw new citations (incoming edges) due to their visibility.

Score from your peers: **0**

Conclusion: In general, the in-degree distribution for citation graphs follows the power law (http://en.wikipedia.org/wiki/Power_law) which captures the statistical behavior of many types of phenomena. Distributions following the power law are well-approximated by an expression of the form k^α where k is the free variable.

Note that the in-degree distribution for the citation graphs follows the power law (with $\alpha < 0$) since the log/log plot is nearly linear. As observed in the class notes on "Logs and exponentials", the value for α corresponds to the negative slope of a line that approximates the points in the plot.

Comments: Please enter an explanation for your scoring, especially if you deducted any points for one of the rubric items for this question.

peer 1 → *[This area was left blank by the evaluator.]*

peer 2 → Wrong phenomenon stated as being an explanation for items B and C.

peer 3 → *[This area was left blank by the evaluator.]*

peer 4 → *[This area was left blank by the evaluator.]*