TSKS33 Hands-On Session 4

Fall 2023

Version of this document: October 17, 2023

Preparation Work (to be Done Before Coming to the Lab)

- Study the course material on sampling, Chapter 6 in the course notes by E. G. Larsson.
- Solve tutorial problems EGL-24 and EGL-27.

Objective

This lab deals with exploration of networks via sampling and random walks. The task requires some programming.

Two networks are given:

- A modified 10-star, with ground truth (used for debugging).
- The LiveJournal dataset from [1]. (LiveJournal is an on-line community with almost 10 million members; a significant fraction of these members are highly active. For example, roughly 300,000 update their content in any given 24-hour period. LiveJournal allows members to maintain journals, individual and group blogs, and it allows people to declare which other members are their friends they belong.)

Each node n has an attribute, x_n, as in the course material. The attributes have been generated synthetically, in such a way that they are correlated with the node degrees.

To load the dataset in Python, use `load_data.py`.[1] The variable `G` contains the network, and `h` contains the attributes.

Throughout, use $S = 10^4$.

[1] For the LiveJournal dataset, this program can take a long time to run.

Task 1: Exact average

Compute $\langle x \rangle$.

Task 2: Uniform node sampling

Sample S nodes uniformly at random and estimate $\langle x \rangle$. Compare your estimate, $\widehat{\langle x \rangle}$, to the true value $\langle x \rangle$.

Run the sampling algorithm five times (with different random seeds), and compare the output. Comment on the variability in the results.

Task 3: Neighbor-of-neighbor sampling

Sample S nodes by picking a node uniformly at random and examining a random neighbor of it. Estimate $\langle x \rangle$ by computing the arithmetic average of the x-values seen this way. Compare to (i) the true value, and (ii) the theoretical expected value. If the values differ, explain why and if they are the same, explain why.

Run the sampling algorithm five times (with different random seeds) and compare the output. Comment on the variability in the results.

Task 4: Uniform random-walk sampling

Sample S nodes by a uniform random walk. Pick a starting node arbitrarily. Run the random walk for S steps to ensure that the sampler is in "steady-state". Then run the random walk for another S steps to collect S x-values.

Estimate $\langle x \rangle$ by taking the arithmetic average of the x-values seen during the random walk. Compare to (i) the true value, (ii) the theoretical expected value. If the values differ, explain why and if they are the same, explain why.

Run the uniform random walk sampler five times (with different random seeds), and compare the outputs. Comment on the variability in the results.

Task 5: Metropolis-Hastings random walk sampling

Same as task 4, but implement the Metropolis-Hastings sampler instead.

Estimate $\langle x \rangle$ by taking the arithmetic average of the x-values seen. Compare to (i) the true value, (ii) the theoretical expected value. If the values differ, explain why and if they are the same, explain why.

Run the Metropolis-Hastings sampler five times (with different random seeds), and compare the output. Comment on the variability in the results.

Task 6: Reflection

- Compare, numerically, the results of Tasks 1, 2, 3 and 4 and 5.
- In what practical situations would one use the different samplers?
- A drawback of random-walk samplers is that they can “backtrack”, i.e., go back to a node that was just visited. Can you think of a way of avoiding that? If one were to “forbid” backtracking, for example by modifying the M-H sampler with an extra “if statement”, do you think that would introduce an observation bias or not? (A speculative answer is sufficient.)

Sample output from the reference solution with the modified 10-star (for debugging)

```
-- expected values of <x>-hat -----
uniform sampling: 2.000
random connection of random node: 7.522
uniform random walk: 4.800
M-H random walk: 2.000
---estimated <x> -----
uniform sampling: 2.001
uniform sampling: 1.987
uniform sampling: 1.952
uniform sampling: 2.005
uniform sampling: 1.999
random connection of random node: 7.547
random connection of random node: 7.549
random connection of random node: 7.551
random connection of random node: 7.514
random connection of random node: 7.526
uniform random walk: 4.796
uniform random walk: 4.802
uniform random walk: 4.786
uniform random walk: 4.782
uniform random walk: 4.800
M-H random walk: 1.996
M-H random walk: 1.997
M-H random walk: 2.000
M-H random walk: 2.003
M-H random walk: 2.023
------------------------
```

Examination

- Individual oral examination takes place in class (computer lab). Students are also expected to be able to answer questions relating to the course material. All students must study the pertinent course material before coming to the lab.
- Before asking for oral examination, make sure to have all your plots, figures and code readily available. It is suggested to save all plots and code prepared in the lab by collating them into a document, for example, in PowerPoint or LibreOffice.
- Collaboration on this homework in small groups is encouraged, but each student should perform programming work individually, and individually demonstrate understanding of all tasks.
- Library functions from the Python standard library, numpy, scipy, matplotlib, and SNAP may be used freely. Copying of code from the Internet or from other students is prohibited.
- The program code you have written should be uploaded to Lisam (go to "submissions" and then select the lab session number) for an anti-plagiarism check.

  Plagiarism (copying of code from other students, from the Internet, or other sources) is a serious offense at LiU and normally leads to the filing of a disciplinary case.

References

[1] J. Leskovec and A. Krevl, "SNAP Datasets: Stanford large network dataset collection," http://snap.stanford.edu/data, Jun. 2014.