Finding a "Kneedle" in a Haystack: Detecting Knee Points in System Behavior

Ville Satopää and **Jeannie Albrecht**, *Williams College*

David Irwin, *University of Massachusetts Amherst*

Barath Raghavan, *International Computer Science Institute*

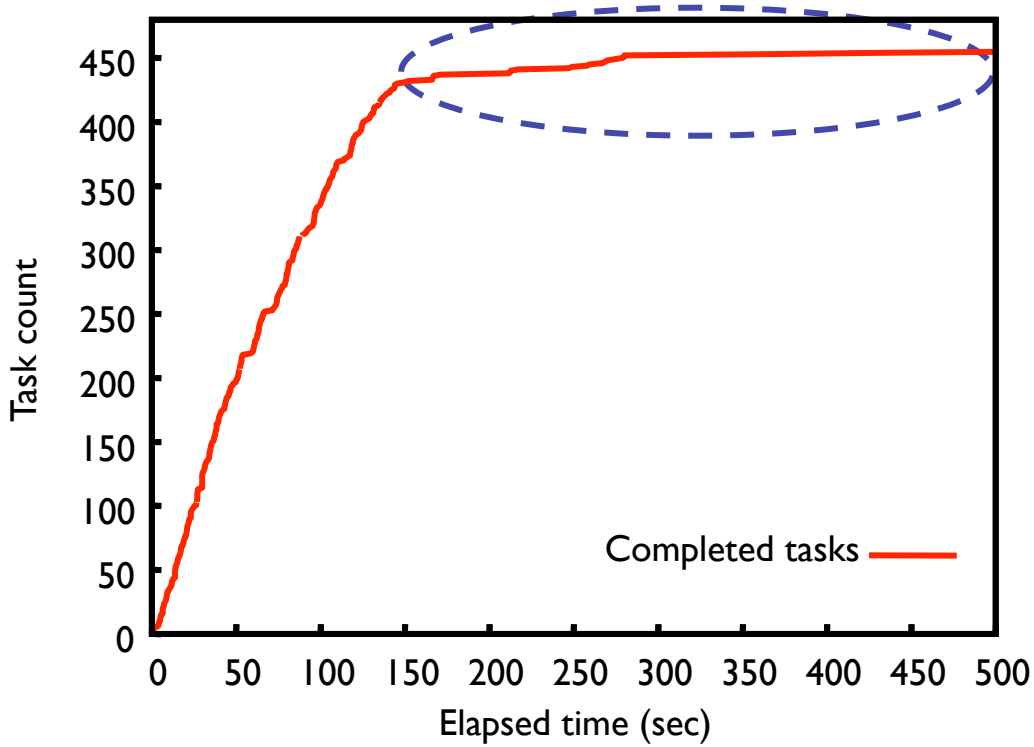
Ville Satopää

Williams College, Class of 2011
B.A., Math and Computer Science

The Wharton School, UPenn
Ph.D. program in Statistics

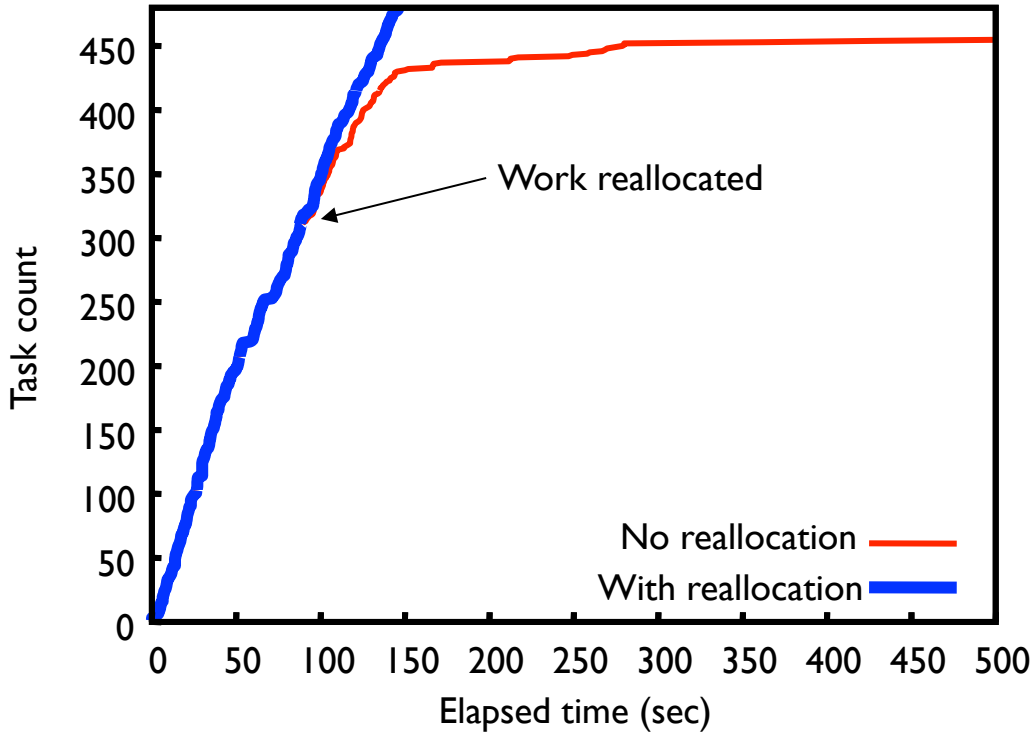
Background

- First noticed problem back in 2006 when building a distributed application management framework (USENIX 2006)
- Needed a way to cope with *stragglers* in distributed computations



Dealing with Stragglers

- Detect slow and/or failed machines (that finish after **knee**)
- Continue computation without stragglers
- Reallocate work as needed



Related Problems

- "One of the common causes that lengthens the total time taken for a MapReduce operation is a '**straggler**': a machine that takes an unusually long time to complete one of the last few map or reduce tasks in the computation." – Dean & Ghemawat, 2004

- "If the load is small, throughput generally keeps up with the load. As the load increases, throughput increases. After the load reaches the network capacity, throughput stops increasing. This point is called the **knee**." – Jain, 1990

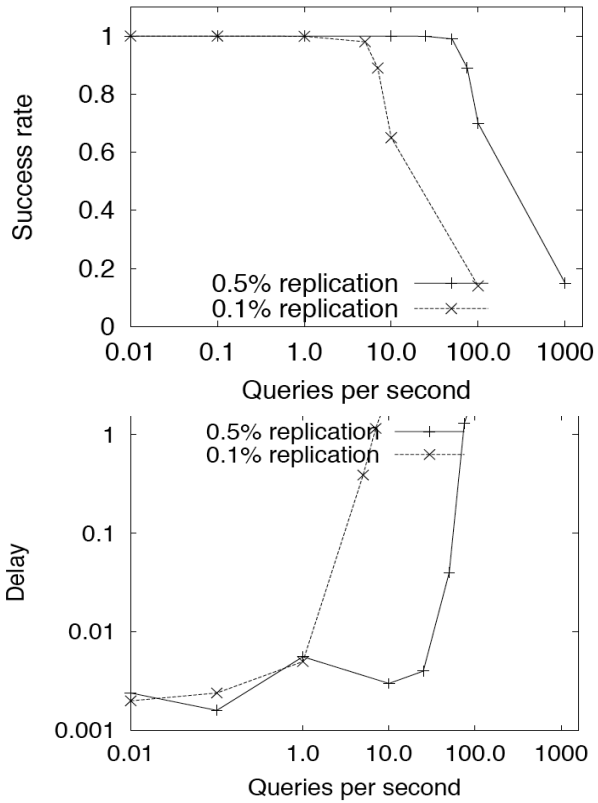
Related Problems

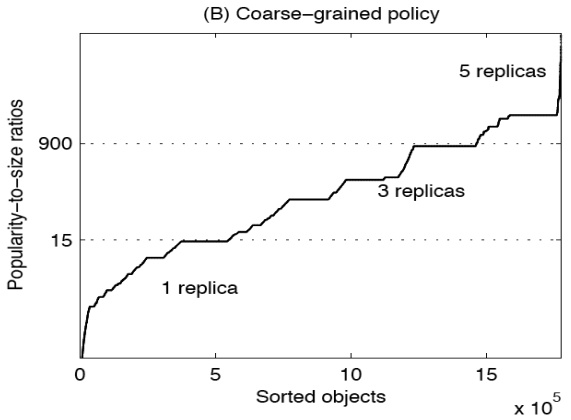
- "As each of the graphs in the figure shows, when the query load increases, we notice a sharp '**knee**' in the curves beyond which the success rate drops sharply and delays increase rapidly."

  – Chawathe et al, 2003

- "For the purpose of determining the thresholds of a coarse-grained policy, the **knee** points on the curve of object popularity-to-size ratios may serve as good candidates."

  – Zhong et al, 2008





Problem Statement

- What is a knee?
  - For simplicity, we assume "elbows" are equivalent to knees (via a trivial conversion)
- Can we develop a general purpose knee detection algorithm that does not require tuning for a specific system and is applicable in a wide range of settings?

Outline

- Background and motivation
- Problem statement
- Our approach
  - Defining knees
  - Detecting knees
- Results and evaluation
  - Synthetic and real data
  - Online and offline scenarios
- Conclusion

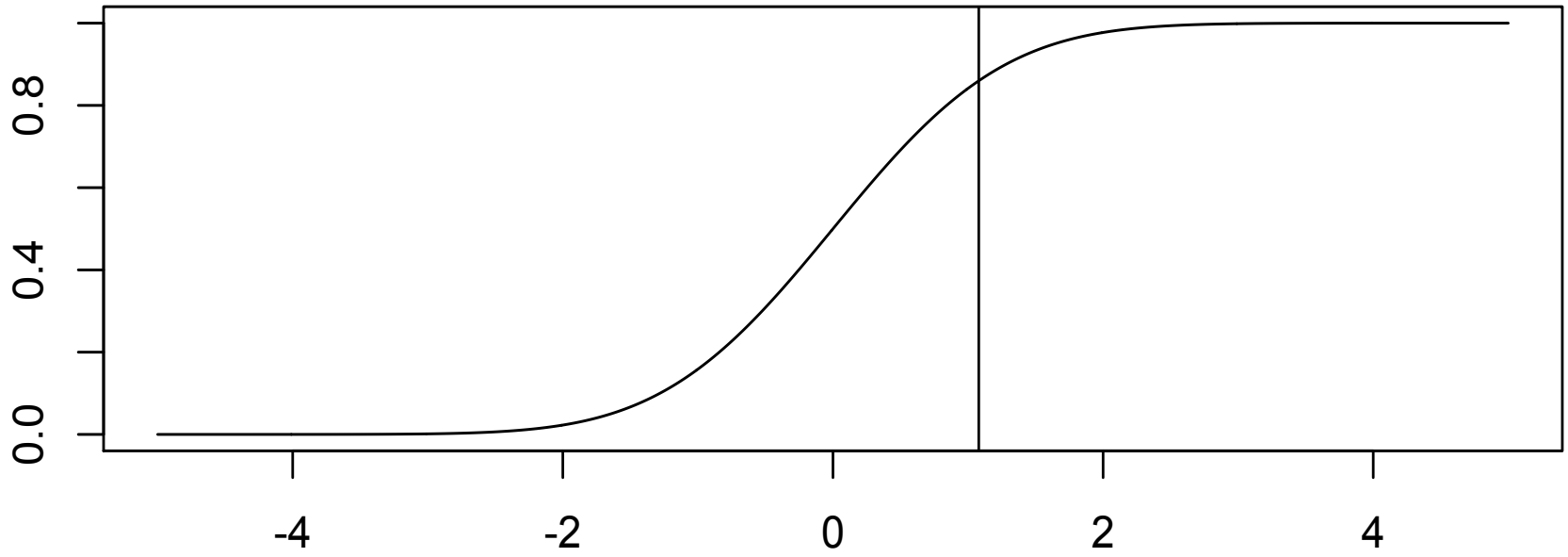
Defining Continuous Knees

- For continuous functions:
  - Knees = points of maximum curvature
  - Curvature based on first and second derivative

$$K_f(x) = \frac{f''(x)}{\left(1 + f'(x)^2\right)^{1.5}}$$

Defining Continuous Knees

- For CDF of Gaussian with mean=0 and std dev=1, max $K_f(x)$ occurs at x ≈ 1.
- Note: Inflection point occurs at x = 0, which is not the knee!



Limitations

- Problems with max curvature:
  - Data is often discrete and non-continuous
  - Data is not easily fit to a single continuous function
  - Max curvature can occur outside of data range
  - Does not work well in real-time/online scenarios

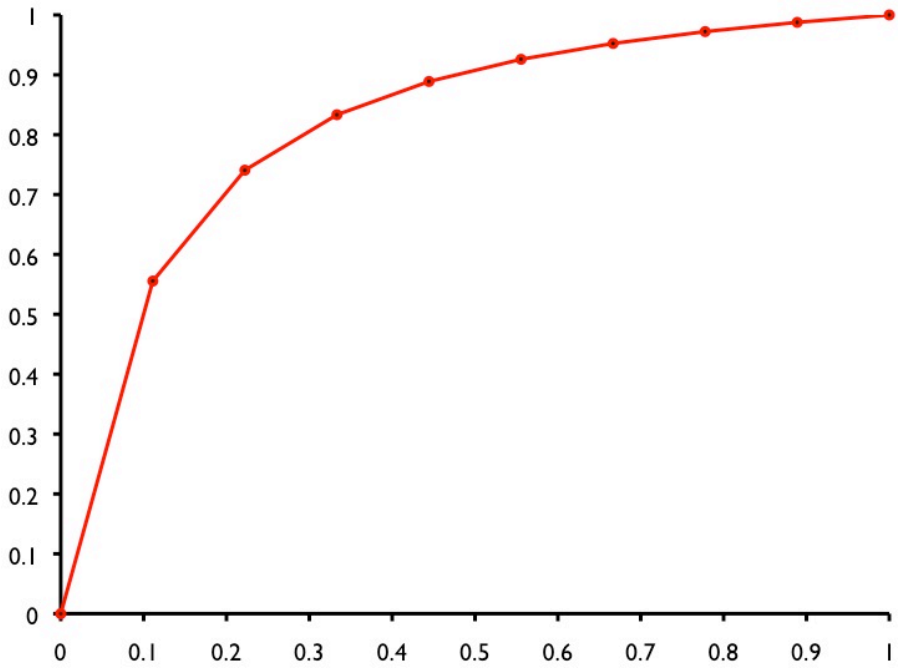
- How do we detect knees in discrete data?

Kneedle

- Our "Kneedle" algorithm approximates points of maximum curvature in discrete data
- Identifies points where data differs the most from a flat line
- Works in online and offline settings

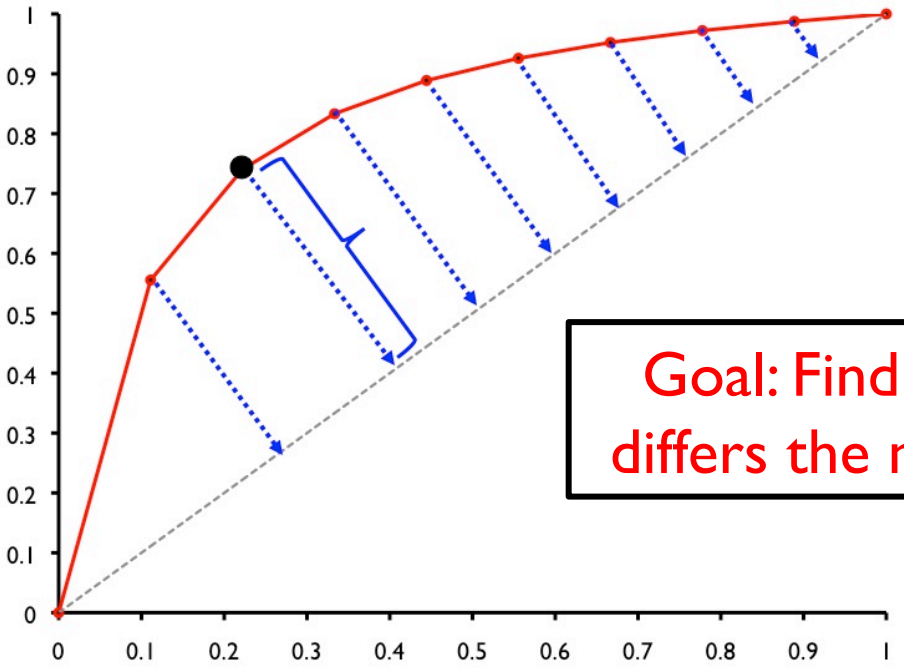
Kneedle in a Nutshell

1. Use smoothing spline to fit data to smooth curve while preserving original shape.
2. Normalize points to unit square.



Kneedle in a Nutshell

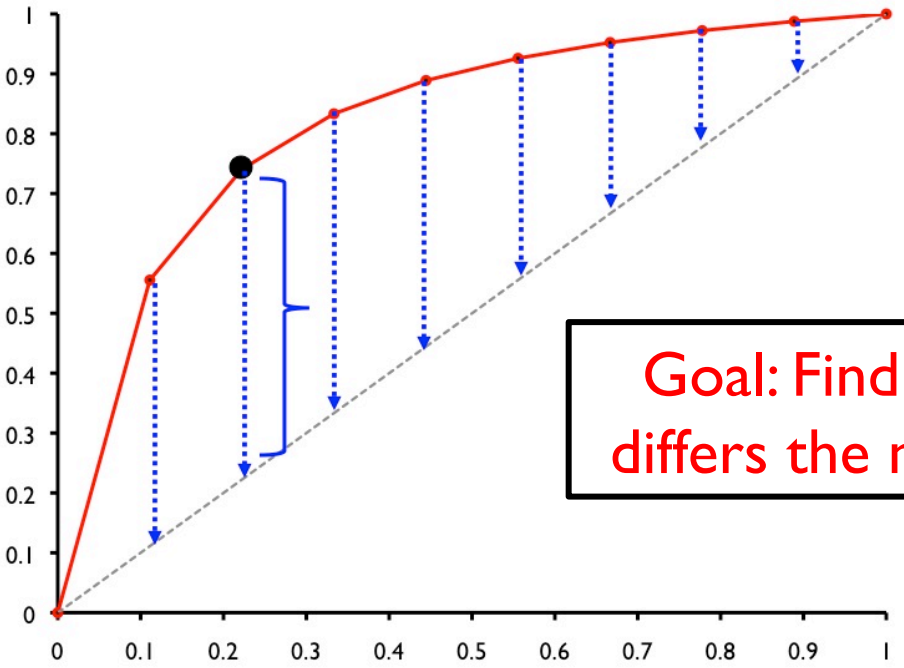
1. Use smoothing spline to fit data to smooth curve while preserving original shape.
2. Normalize points to unit square.



Goal: Find where curve differs the most from y=x

Kneedle in a Nutshell

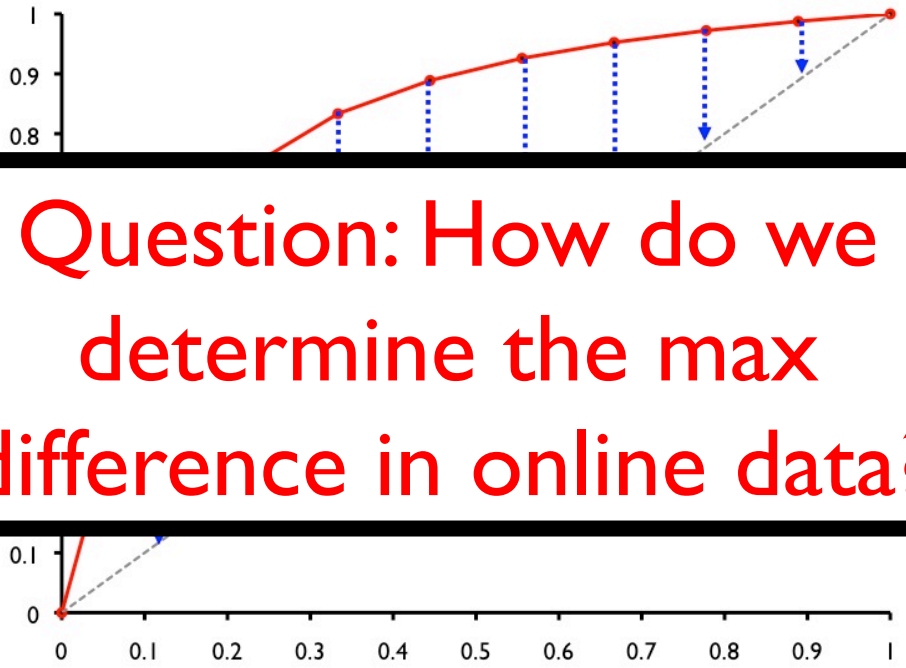
1. Use smoothing spline to fit data to smooth curve while preserving original shape.
2. Normalize points to unit square.



Goal: Find where curve differs the most from y=x

Kneedle in a Nutshell

1. Use smoothing spline to fit data to smooth curve while preserving original shape.
2. Normalize points to unit square.

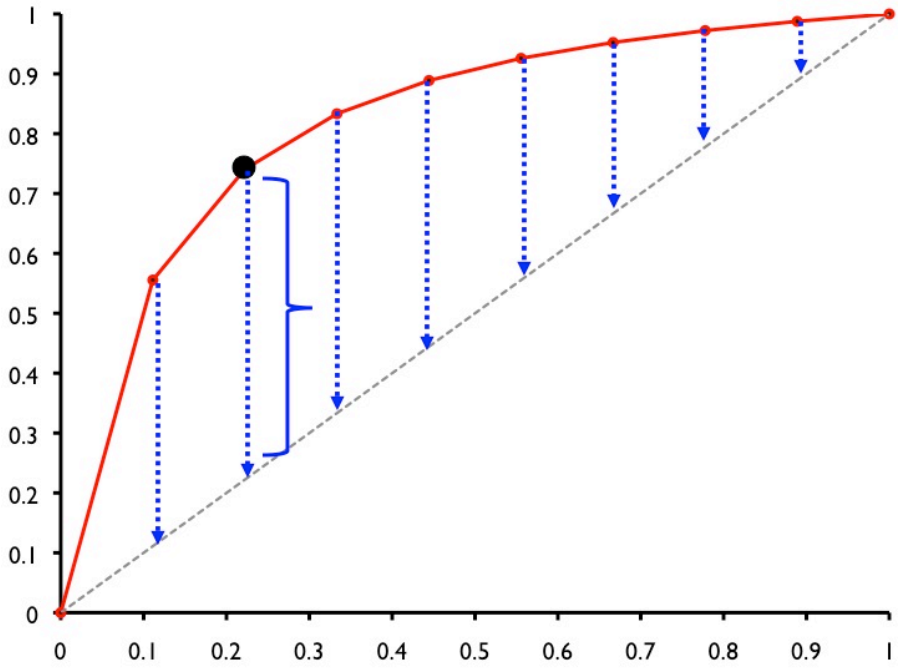


Question: How do we determine the max difference in online data?

ere curve
t from y=x

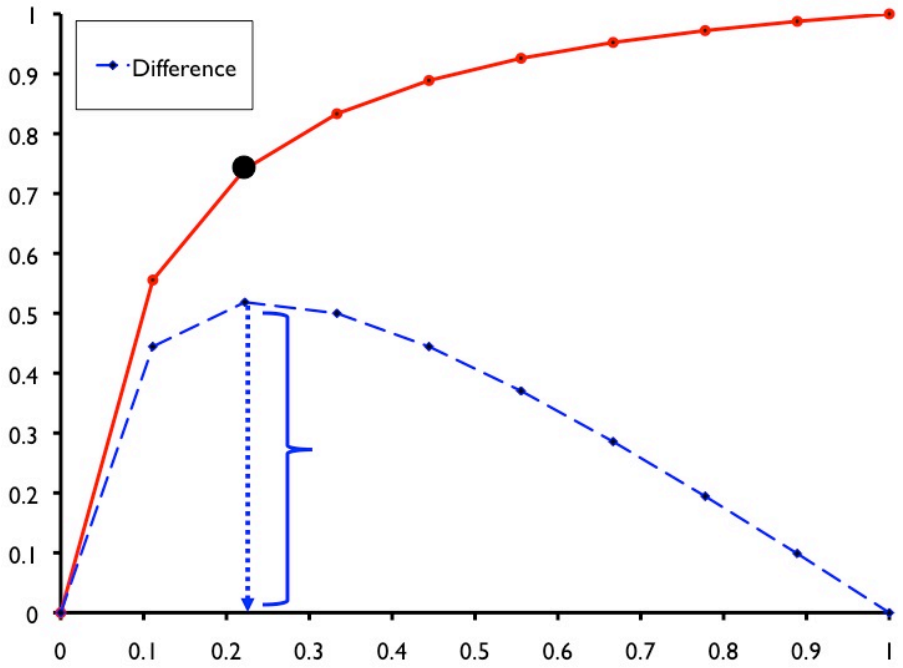
Kneedle in a Nutshell

3. Since we may not know the absolute max in online data, we observe trends and look for local maxima in difference curve.



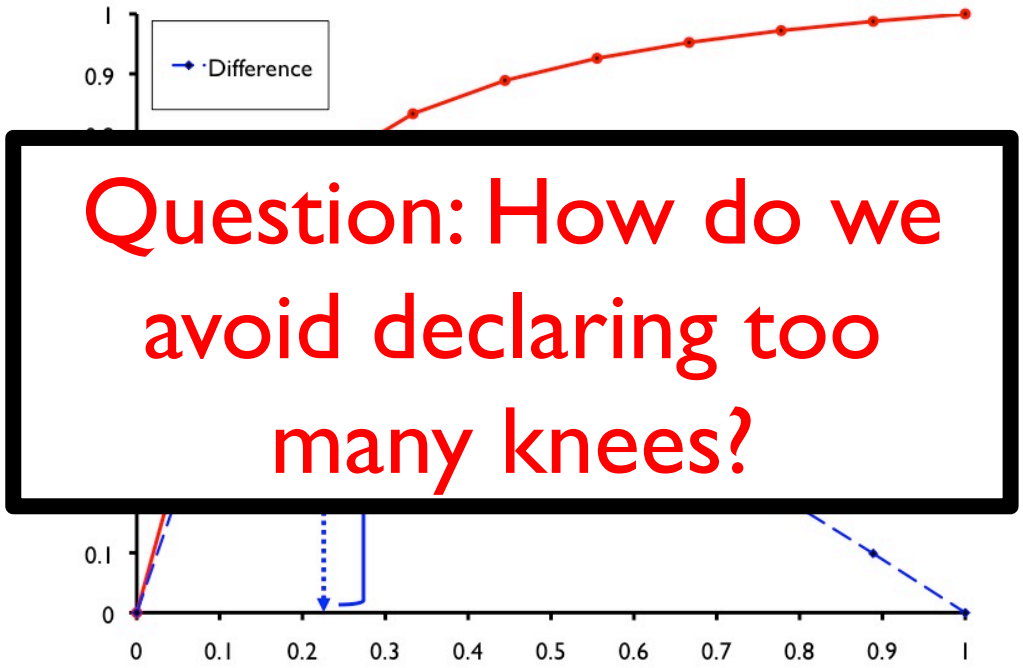
Kneedle in a Nutshell

3. Since we may not know the absolute max in online data, we observe trends and look for local maxima in difference curve.



Kneedle in a Nutshell

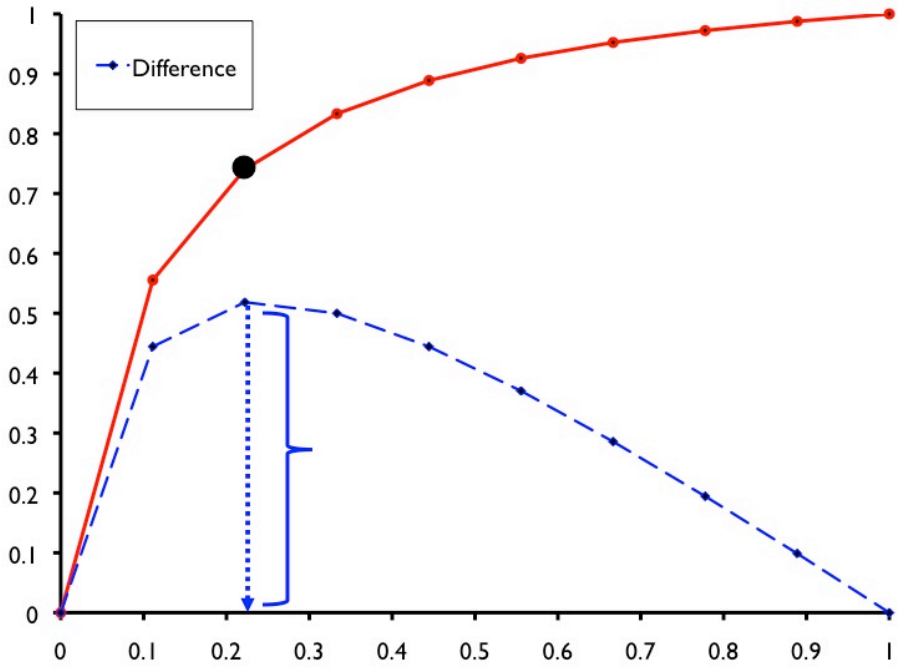
3. Since we may not know the absolute max in online data, we observe trends and look for local maxima in difference curve.



Question: How do we avoid declaring too many knees?

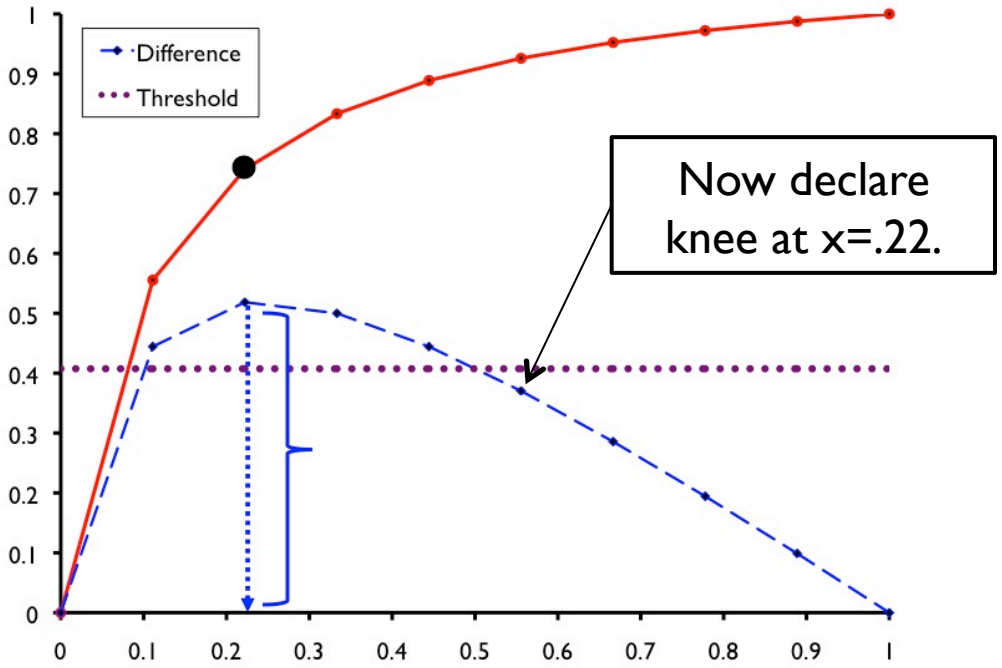
Kneedle in a Nutshell

4. Do not declare a knee immediately upon detecting a local max. Wait for diff values to drop below a configurable threshold.



Kneedle in a Nutshell

4. Do not declare a knee immediately upon detecting a local max. Wait for diff values to drop below a *configurable* threshold.



Evaluation Summary

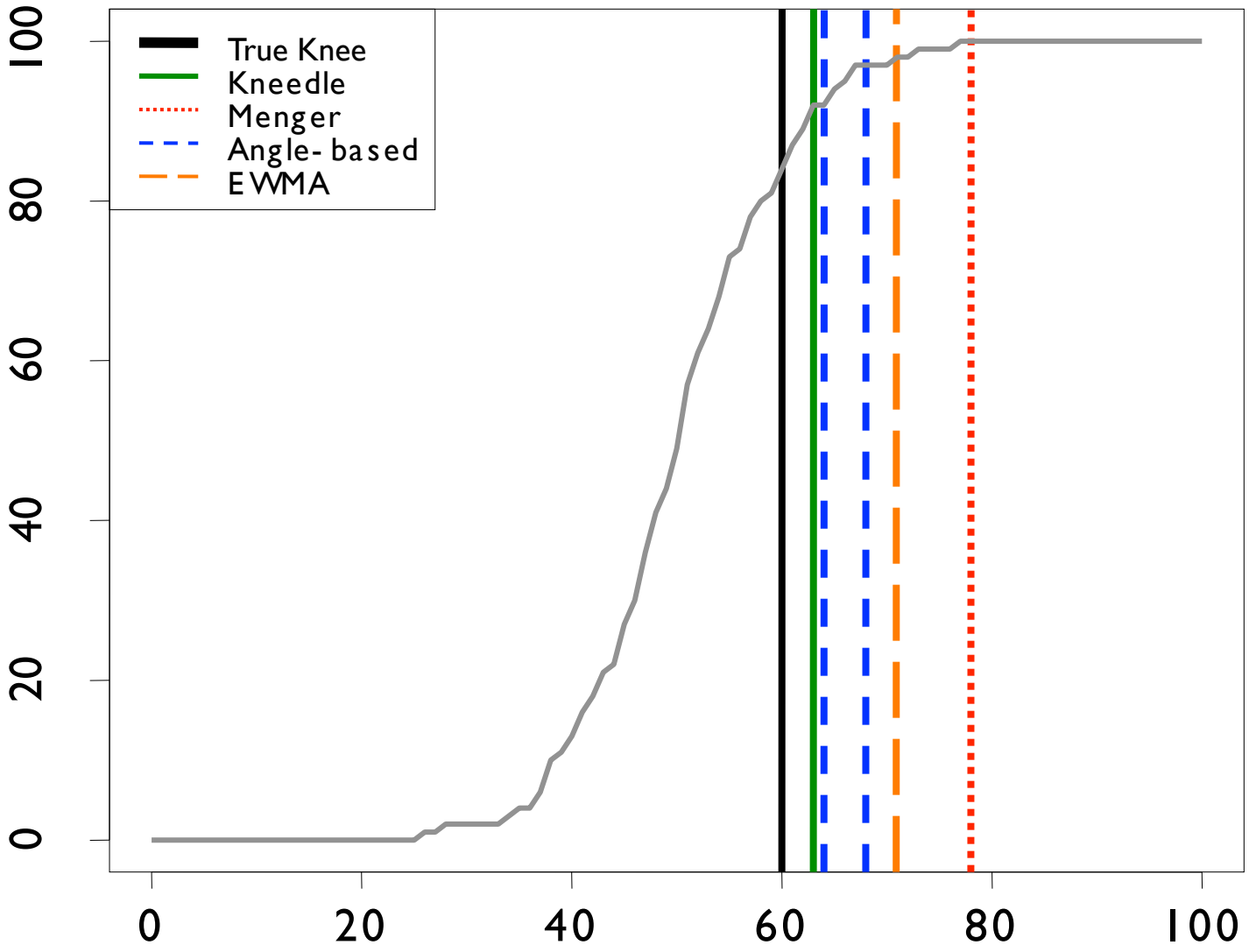
- Used synthetic and real data
  - Synthetic data set – based on a Gaussian CDF; allows us to approximate "true knees" based on curvature equation
  - Real data – extracted from graphs in published papers; extracted from real systems
- Tested in online and offline scenarios
  - Online – "real-time" data taken from running systems
  - Offline – all data points are known in advance

Evaluation Summary

- Metrics of interest:
  - Online scenarios – latency of detection
  - Offline scenarios – accuracy, precision, and recall

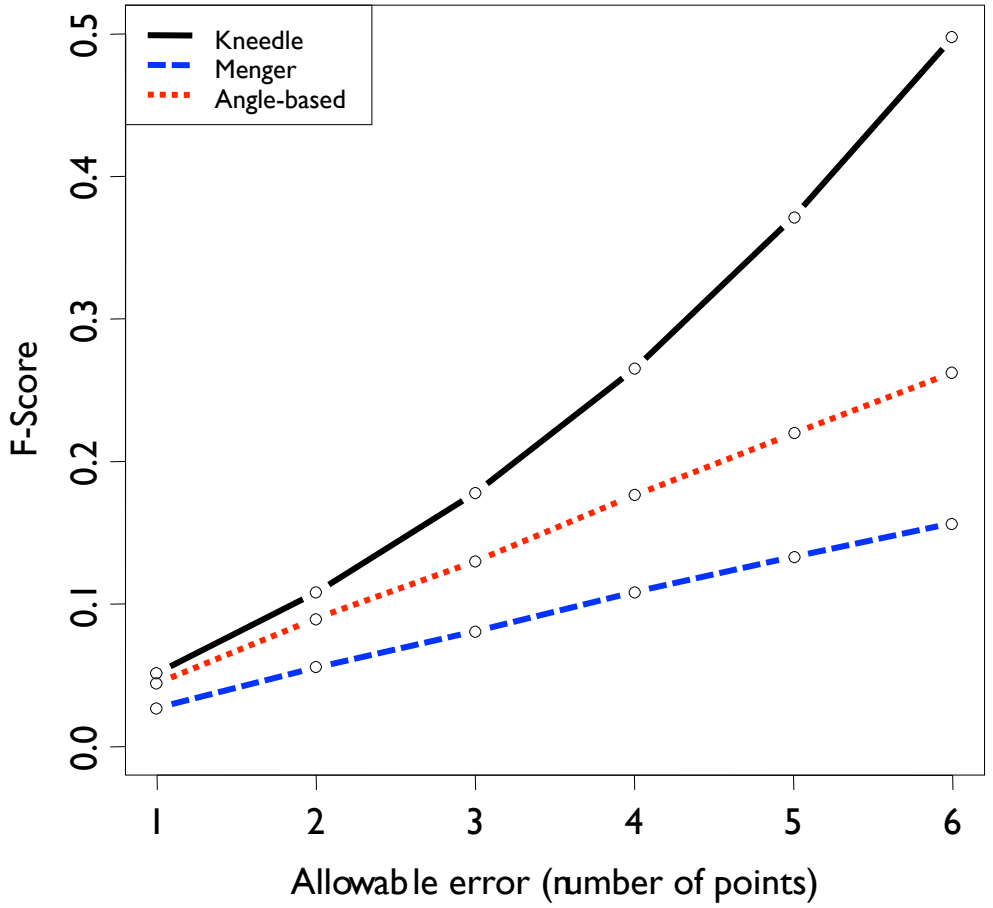
- Compared against other knee-finding algorithms
  - Online scenarios – EWMA
  - Offline scenarios – Menger, Angle-based

Overall Performance



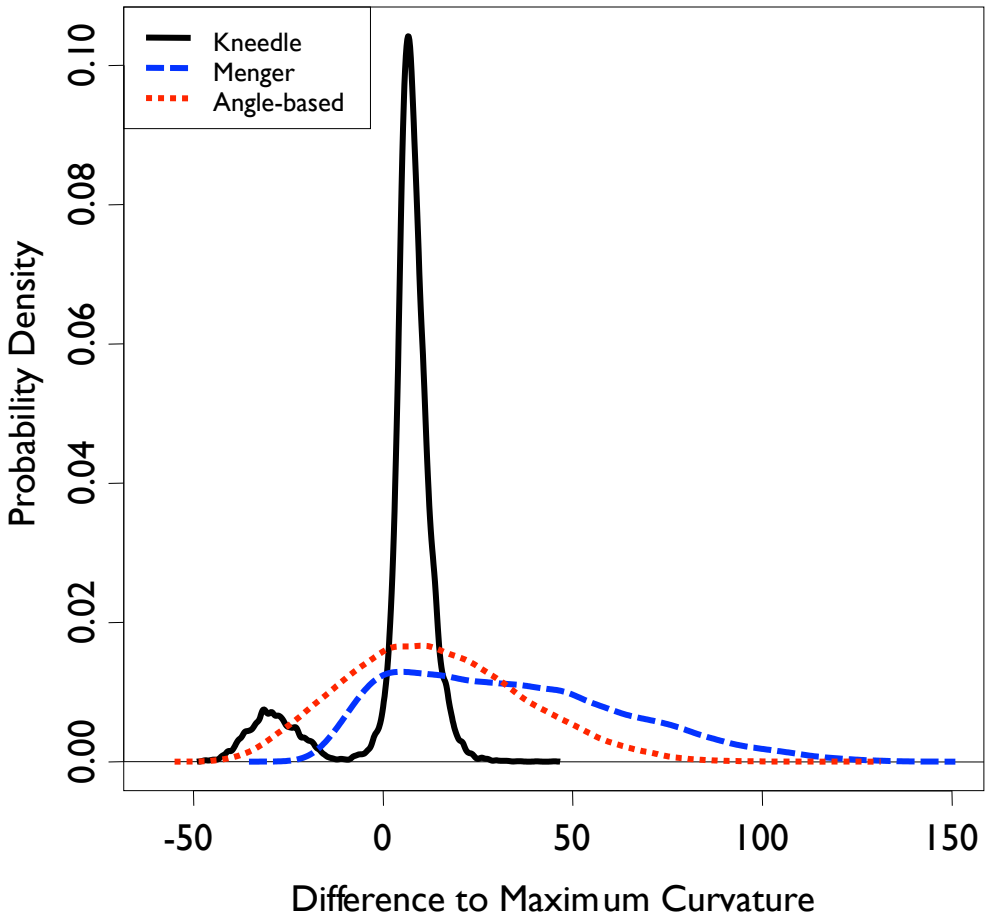
Offline Accuracy – Synthetic Data

- F-Score is harmonic mean of precision and recall
  - Captures correctness and completeness of knees
  - F-Score of 1 is best
- Evaluated against true knees



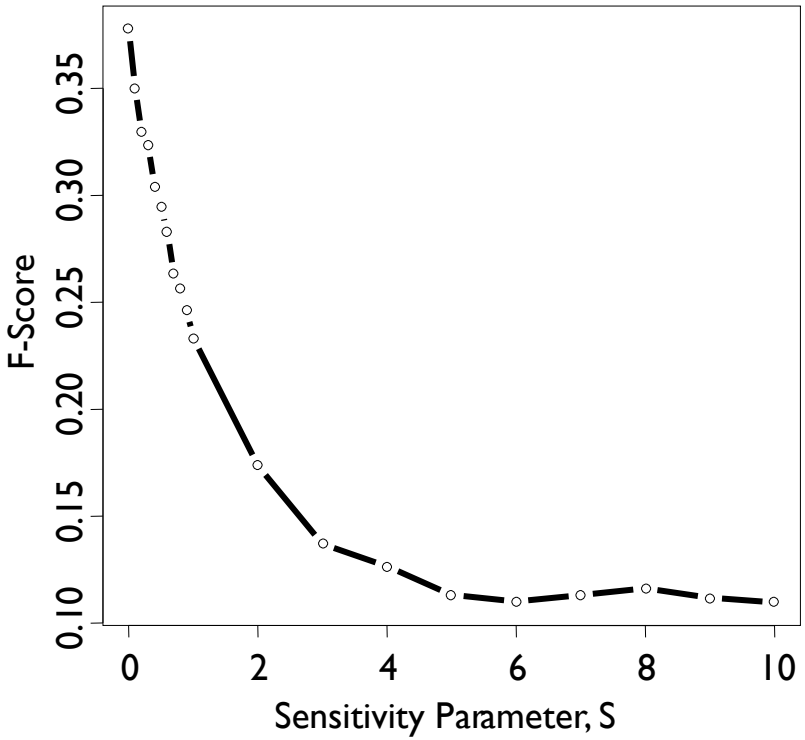
Offline Accuracy – Synthetic Data

- Difference between detected and true knees



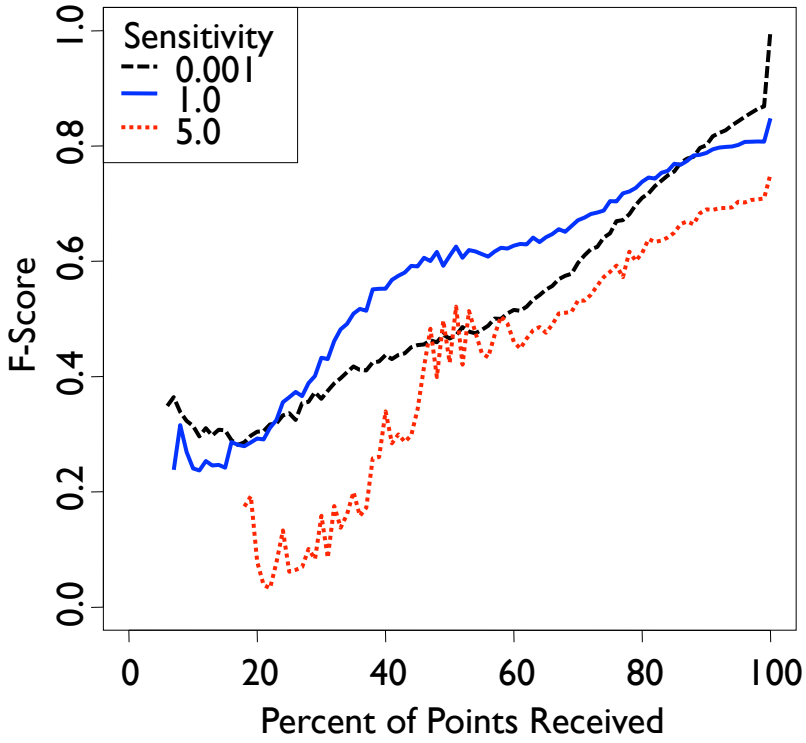
Sensitivity Parameter

Offline synthetic



S=0 is best!

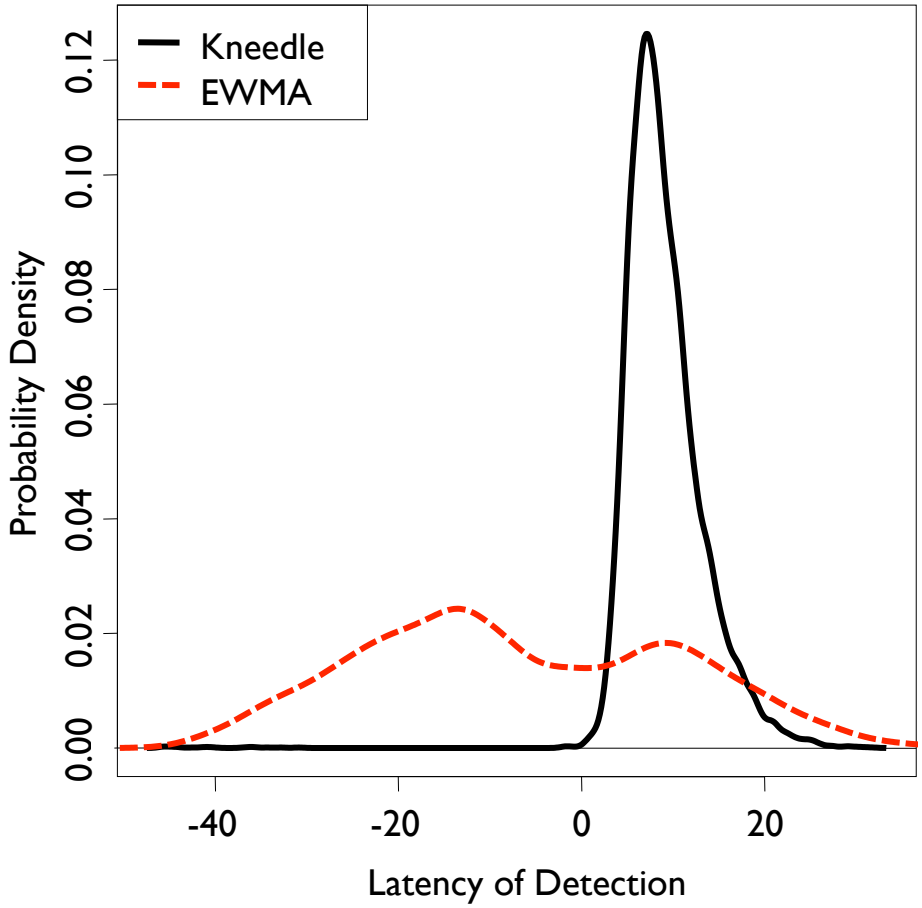
Online synthetic



S=1 is best!

Online Detection Latency

- How long (in number of points) does it take to detect knees



Offline Analysis – Real Data

- Detect knees in real published data

- Data source:

  M. Zhong, K. Shen, and J. Seiferas, "Replication Degree Customization for High Availability," in EuroSys, 2008.

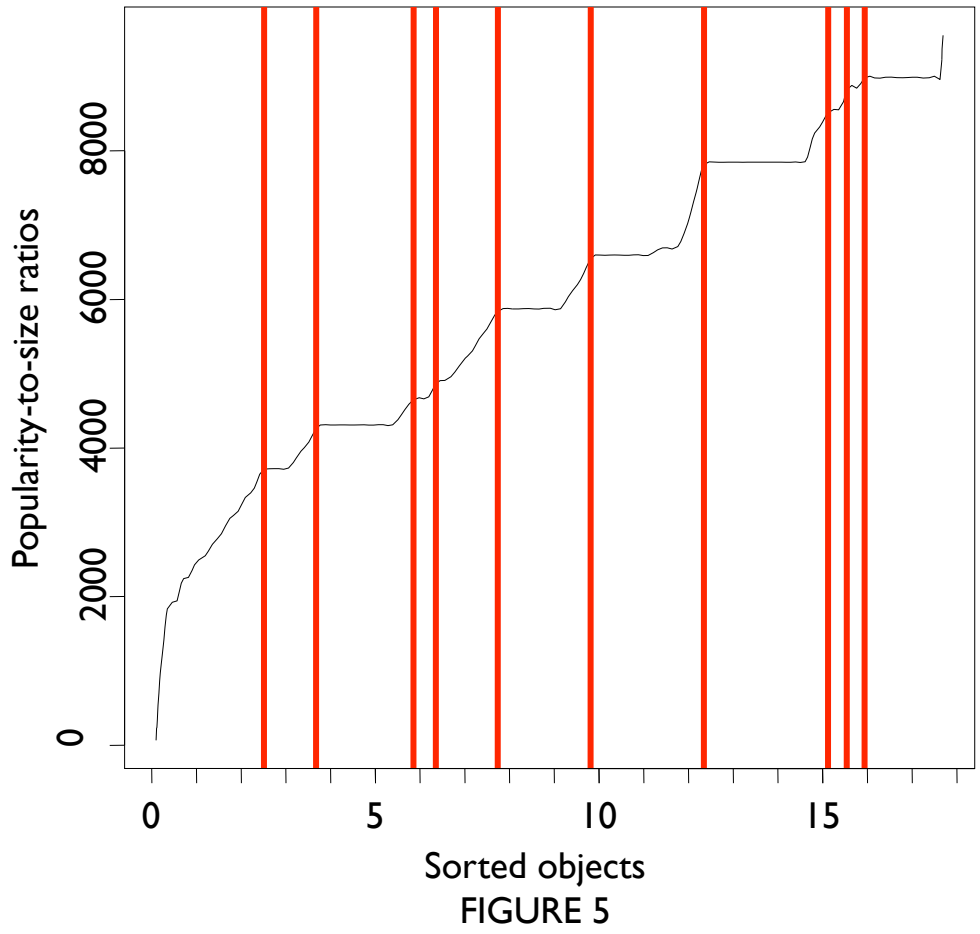
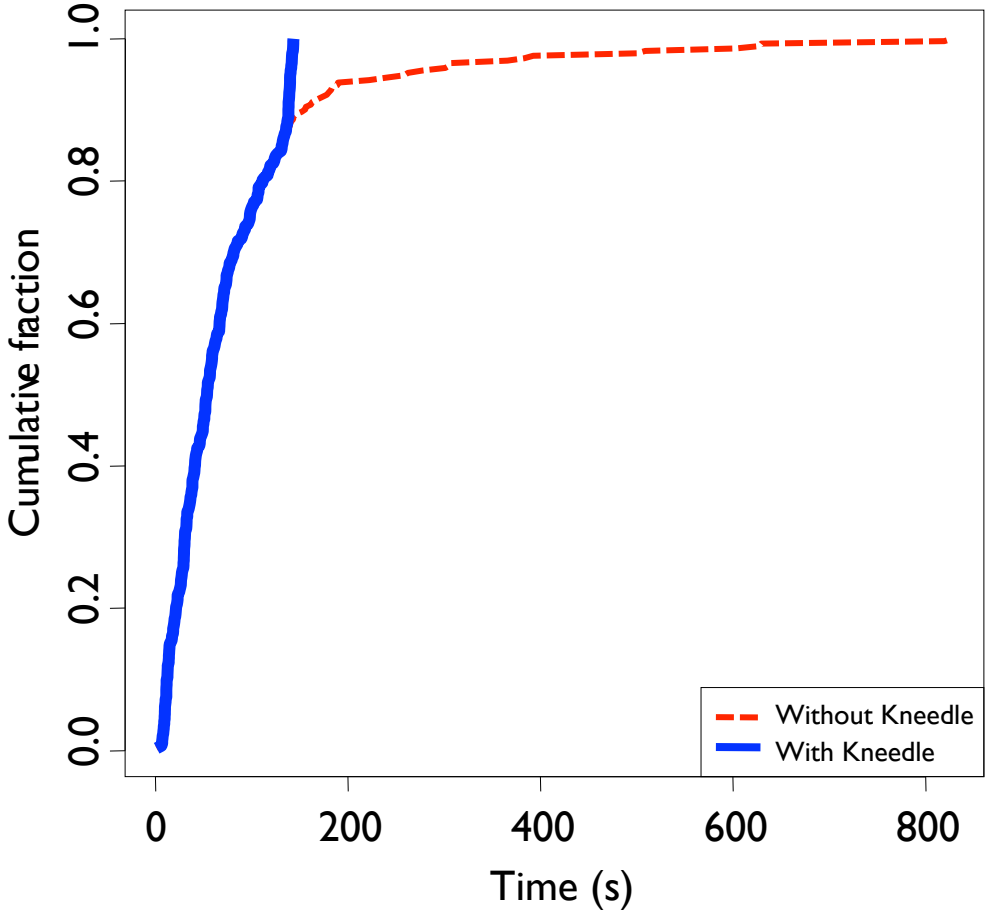


FIGURE 5

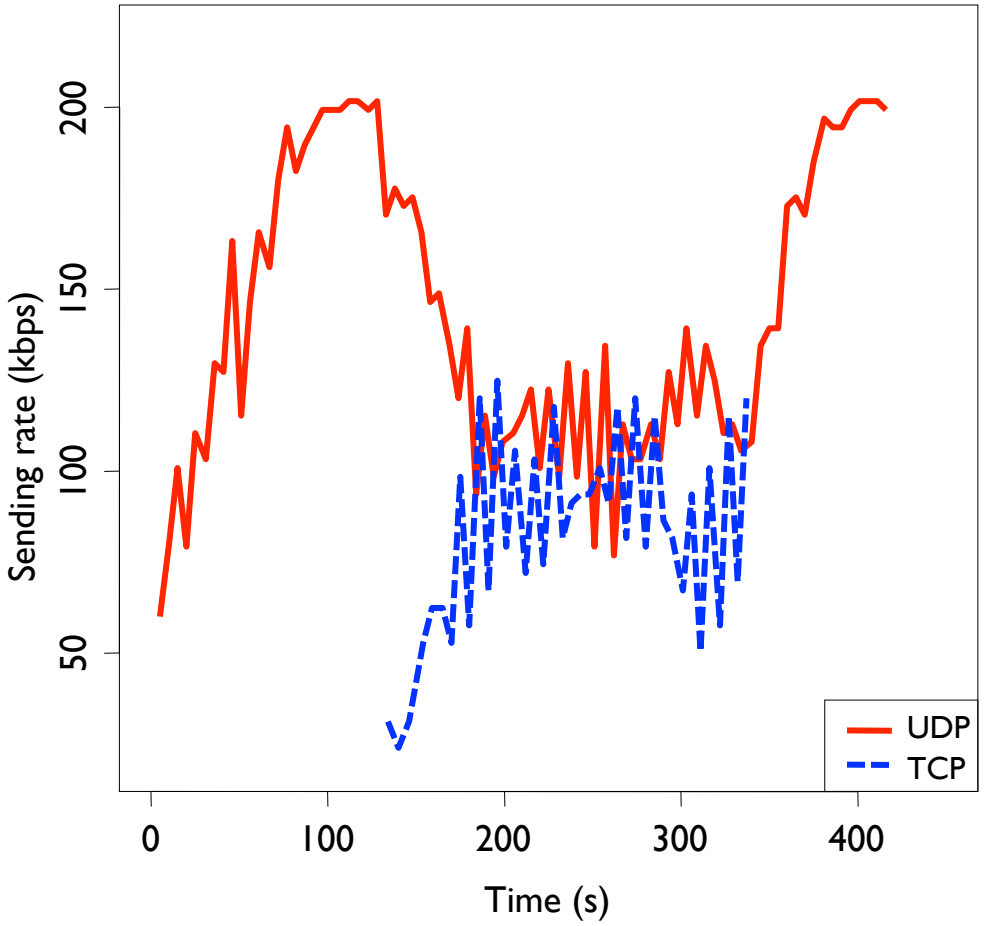
Online Analysis – Real Data

- Reallocate work in MapReduce-like batch scheduler running on PlanetLab



Online Analysis – Real Data

- Determine TCP-friendly sending rate in streaming application



Conclusion

- Contributions
  - Presented formal definition of a knee
  - Described Kneedle, an algorithm for finding knee points in offline and online scenarios
  - Evaluated Kneedle in a variety of settings
  - Integrated Kneedle into existing applications with minimal effort

Thanks!

For more info:
jeannie@cs.williams.edu