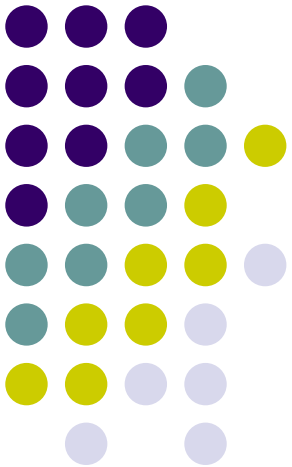
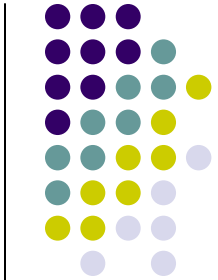
Reducing The Computational Cost of Bayesian Indoor Positioning Systems

Konstantinos Kleisouris, Richard P. Martin
Computer Science Department
Rutgers University



WINLAB Research Review
May 15th, 2006

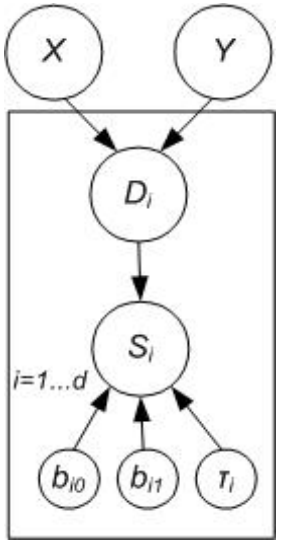
Motivation



- Bayesian networks (BNs)
  - Have recently been used for location estimation in wireless networks
    - Networks M1, M2, M3, A1 [Madigan'05, Elnahrawy]
  - Can incorporate radio properties
    - Received signal strength (RSS)
    - Angle of Arrival of the signal (AoA)
  - Attractive approach
    - Similar performance with smaller training set when compared to other solutions
    - Training set: Radio properties measured at specific locations

Motivation (cont'd)

- M1 network
  - X, Y: latent variables (location)
  - S_i: observable data (RSS)
  - b_{0i}, b_{1i}, τ_i: stochastic variables
  - Linear regression model for RSS
    - $S_i = b_{0i} + b_{1i} \log(1 + D_i)$

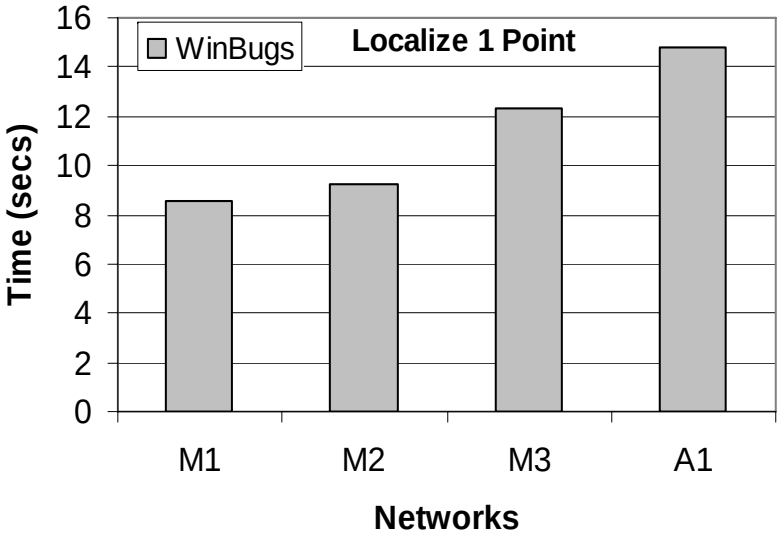


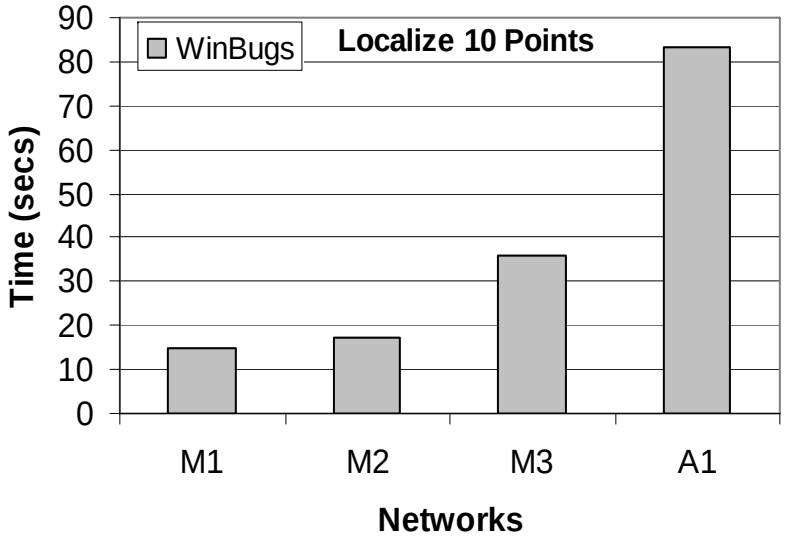
M1

- We want to compute the probability density function (PDF) of X, Y given the training set

Motivation (cont'd)

- Computational cost of using BNs with statistical packages like WinBugs is large
- Platform: Pentium 4 PC, 2.80 GHz, 1 GB RAM, Windows XP
- Training set size: 253

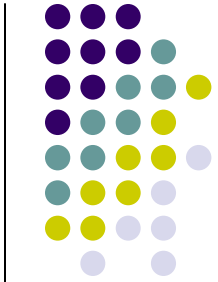




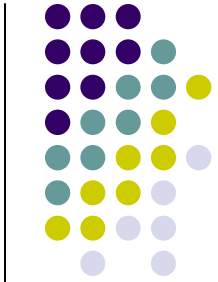
- Can we be faster?

Talk Outline

- Motivation
- Our Approach & Contributions
- Experimental Results
- Analytic Analysis
- Conclusions & Future Work

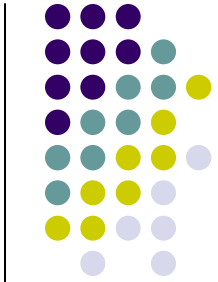


Our Approach



- We have implemented our own solvers
- Our Bayesian networks have no closed-form solution
  - We use Markov Chain Monte Carlo (MCMC) simulation
  - MCMC explores the PDF of the variables in a BN using sampling

MCMC methods

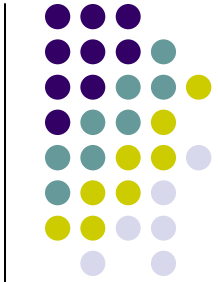


- Gibbs sampling
  - It draws a new value for a variable v from its "full conditional", $f(v) \equiv p(v|V \setminus v)$
    - Conditional PDF given all the other quantities are fixed at their current value

$$p(v|V \setminus v) \propto p(v|pa(v)) \prod_{w \in child(v)} p(w|pa(w))$$

  - E.g. Conjugate sampling, Slice sampling
- Metropolis algorithm
  - Draws candidate values for v from a proposal distribution

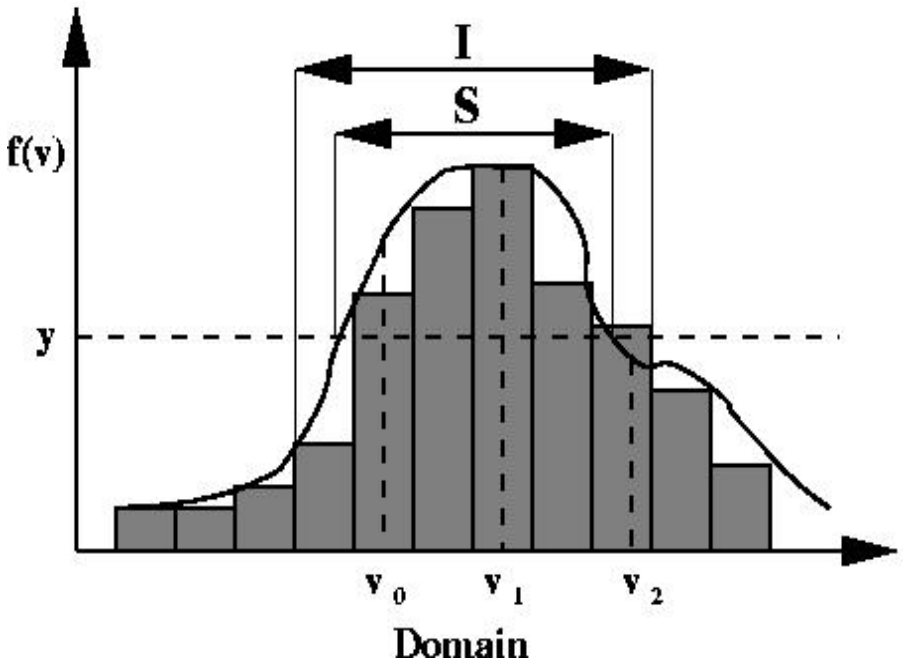
Contributions



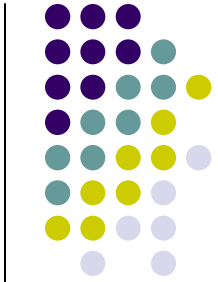
- Low latency in localization
  - 1 or 10 points $\Rightarrow$ 0.5 sec
  - 51 points with no location info in the training set $\Rightarrow$ 6 secs
  - Over 10 times faster than WinBugs
- Full conditionals are flat
  - Most efficient algorithm: Whole domain sampling
- Analytic model
  - How flat a distribution should be in order to use whole domain sampling

Slice Sampling

- Goal: Compute the histogram of a variable v given we can compute only $f(v)$
- Challenge: unknown shape of f

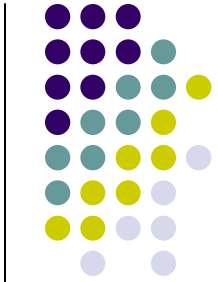


Slice Sampling (cont'd)



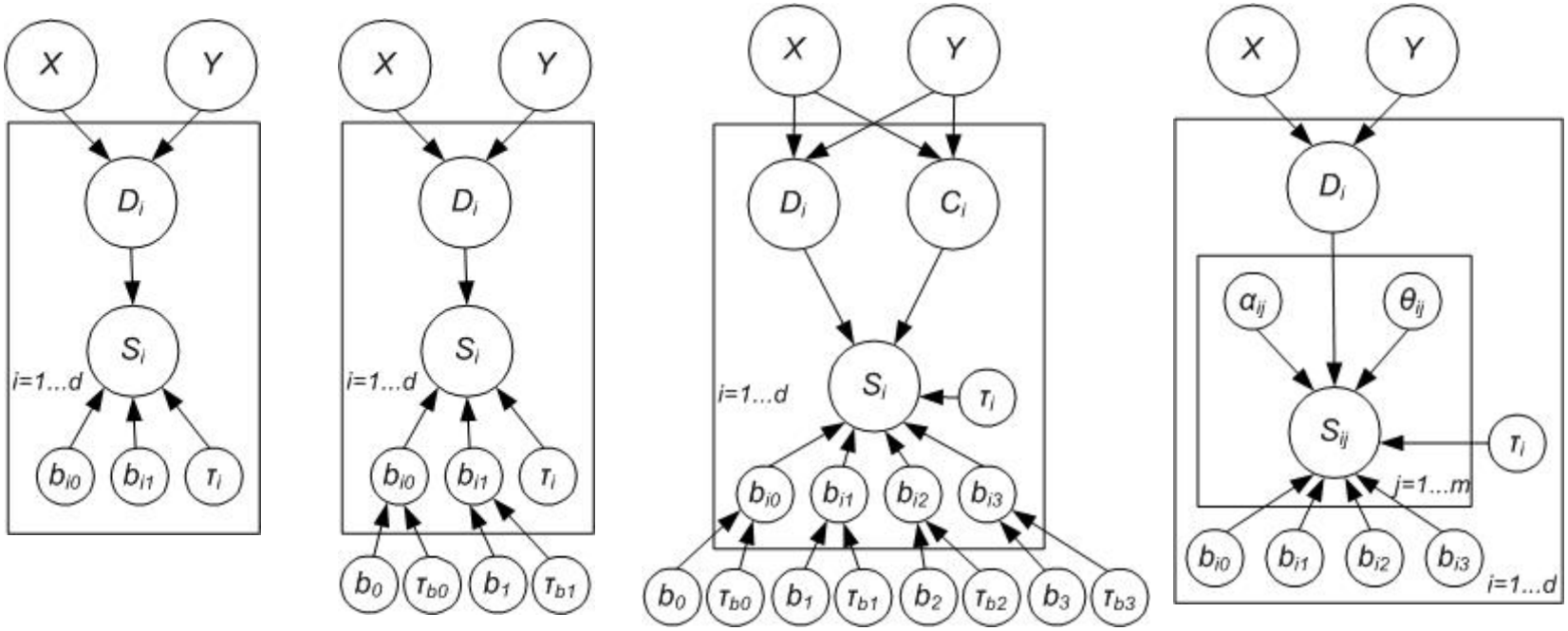
- Defines an interval S given the current value v_0 of a variable v
  - Hard to estimate the edges of S. It is approximated by I
- Schemes to find I
  - Whole domain $\Rightarrow$ Whole Domain Sampling
  - Step out
  - Double out
- The new value of v is chosen from $S \bigcap I$
- As I approximates S better, the fewer the rejections (i.e. values not in $S \bigcap I$)

Tradeoffs of Estimating I



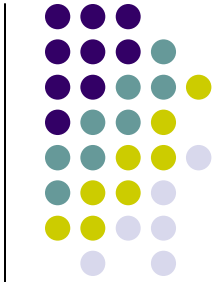
- Whole Domain
  - Pros: Easy to compute I
  - Cons: Potentially, a lot of rejections
- Step Out
  - Pros: A few rejections
  - Cons: Many evaluations of $f(v)$ to compute I
- Double Out: Intermediate solution

Localization Models



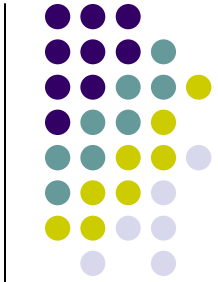
M1 M2 M3 A1

Talk Outline



- Motivation
- Our Approach & Contributions
- Experimental Results
- Analytic Analysis
- Conclusions & Future Work

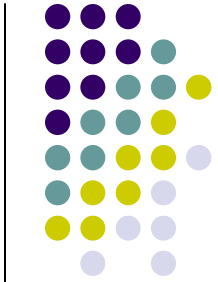
MCMC Algorithms



- Met algorithms
  - Metropolis for X, Y, angle
  - Conjugate sampling for the other variables
- Slice algorithms
  - Slice sampling for X, Y, angle
  - Conjugate sampling for the other variables

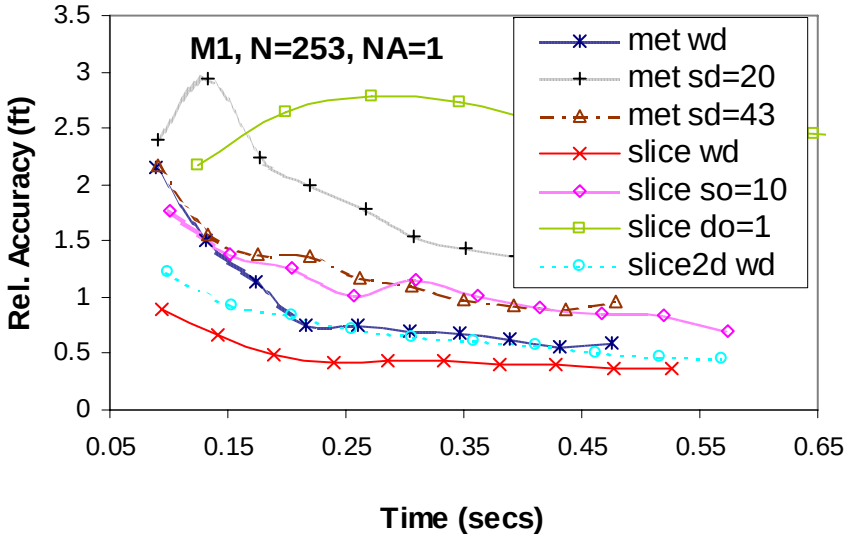
Algorithm	Description
met wd	Metropolis with proposal distribution uniform over the whole domain of X, Y (angle)
met sd = k (or sd=k, l)	Metropolis with proposal distribution Gaussian whose standard deviation = k ft (l rands for angle)
slice wd	Slice sampling over the whole domain X, Y (angle)
slice so = k (or so=k, l)	Slice sampling by stepping out with w = k ft (w=l rands for angle)
slice do = k (or do=k, l)	Slice sampling by doubling out with w = k ft (w = l rands for angle)
slice2d	2d slice sampling by updating (X, Y) together and using the whole domain of X, Y (1d for angle over whole domain)

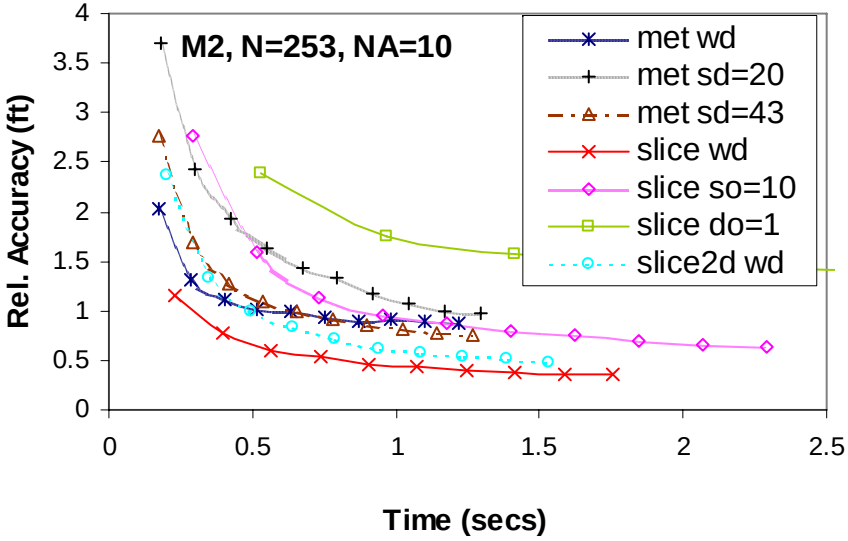
Comparing Algorithms

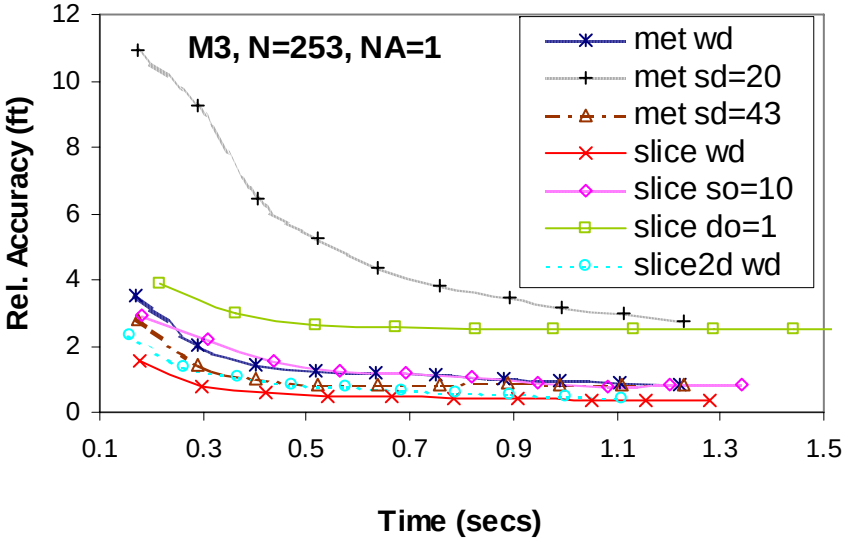


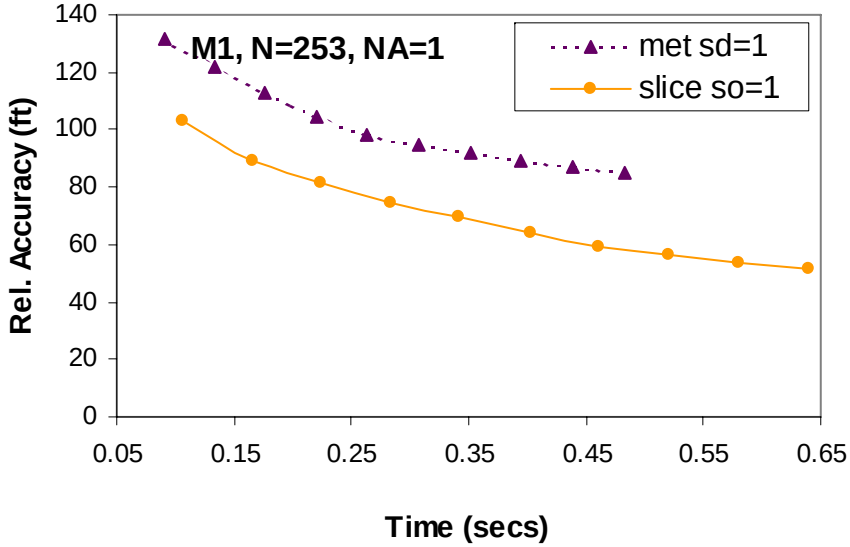
- Metric: Relative Reference
  - Euclidean distance of the mean of our results to the ones from WinBugs
  - WinBugs results
    - 10000 iterations burnin
    - 100000 iterations additional
- Intuition: The results of our solver should converge to the statistics of a well-tested solver after a long run

Comparing Algorithms (cont'd)



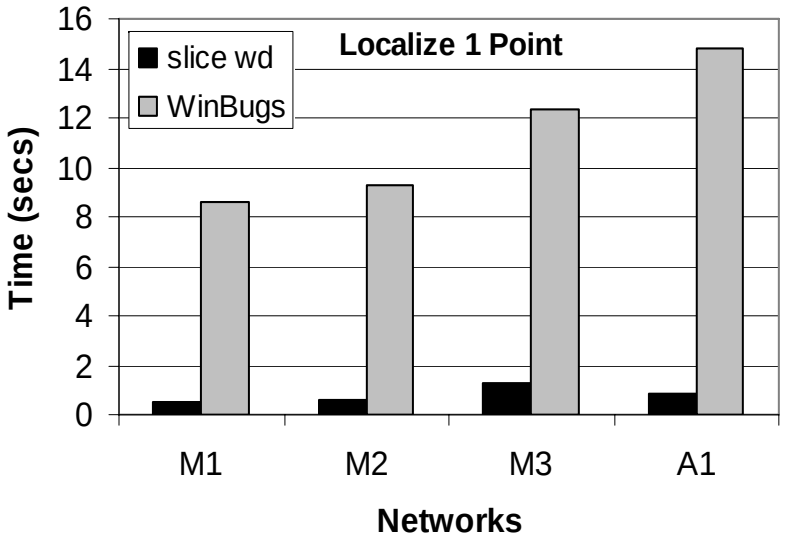


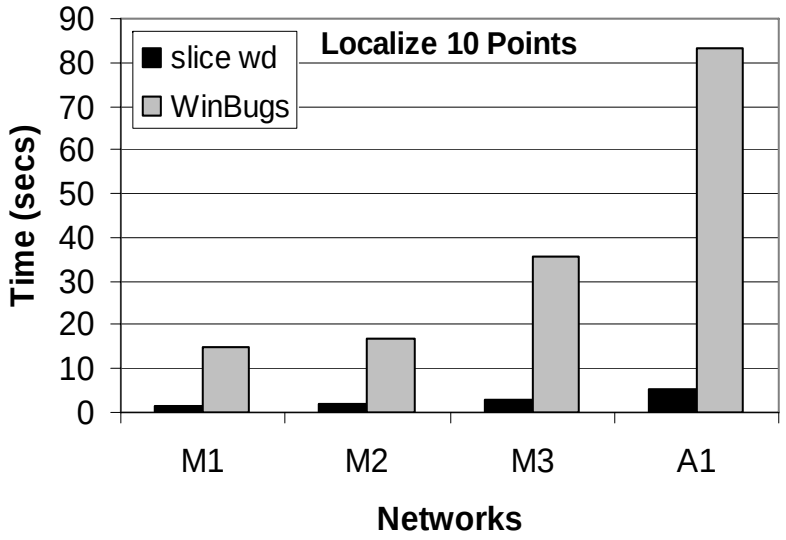




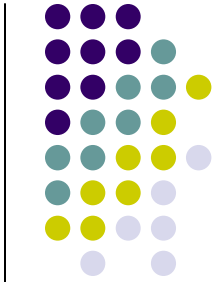
Comparing Against WinBugs

- Whole Domain sampling is faster than WinBugs by a factor that ranges:
  - Localize 1 point: 9.8 (M1) – 17.9 (A1)
  - Localize 10 points: 9.1 (M1) – 16.1 (A1)





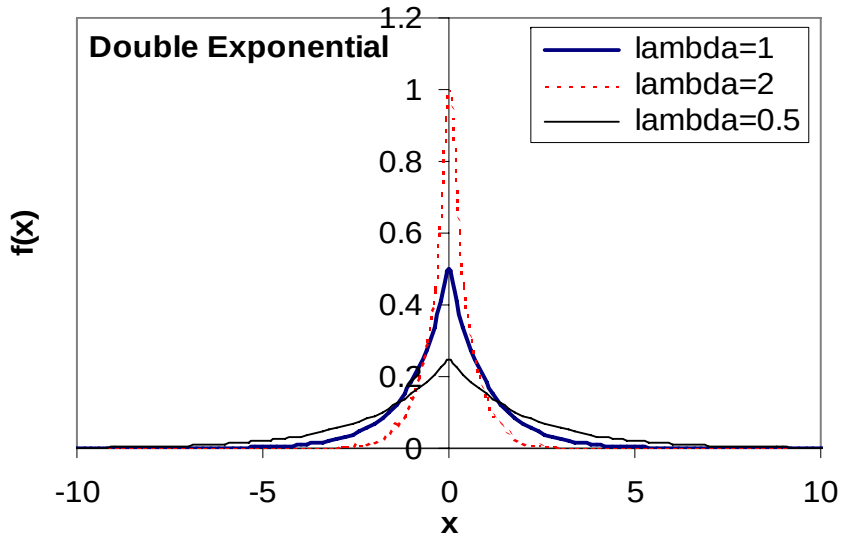
Talk Outline



- Motivation
- Our Approach & Contributions
- Experimental Results
- Analytic Analysis
- Conclusions & Future Work

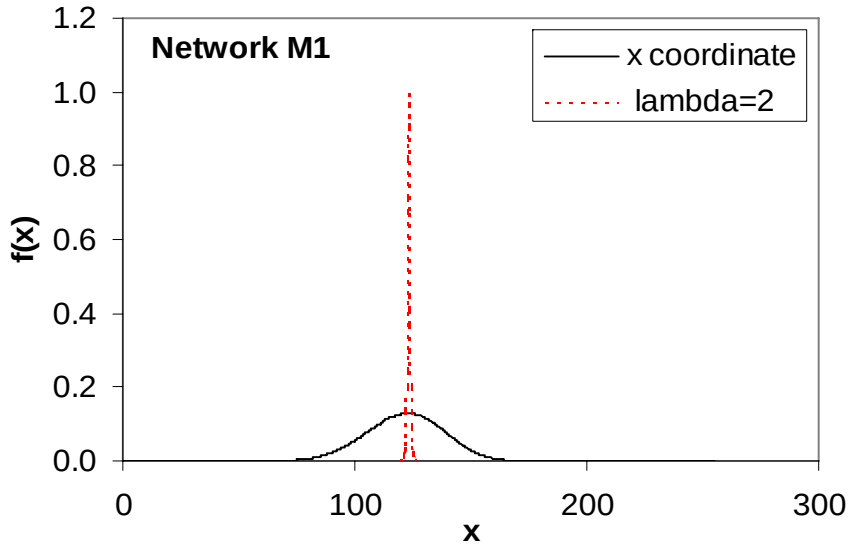
Analytic Analysis

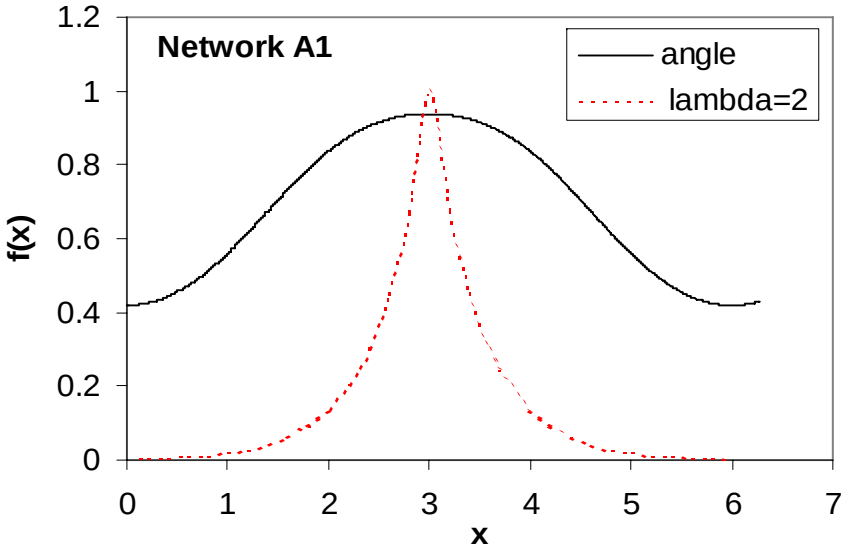
- When whole domain sampling is computationally more efficient than step out?
- We use the double exponential distribution
  - PDF: $$H\ (x;\lambda) = \frac{\lambda e^{-\lambda x}}{2},\ \lambda > 0$$
  - λ determines how peaky the distribution is



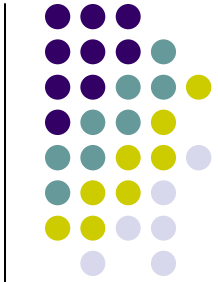
Comparison of flatness

- Analytic model: when $\lambda < 2$ whole domain is faster
- Full conditionals fall into this regime



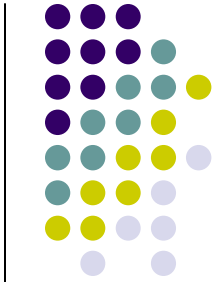


Talk Outline

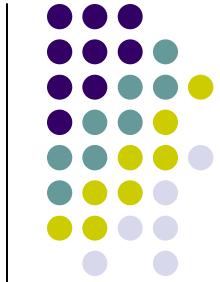


- Motivation
- Our Approach & Contributions
- Experimental Results
- Analytic Analysis
- Conclusions & Future Work

Conclusions & Future Work



- Conclusions
  - Low latency in localization
  - Whole domain sampling has the best performance
    - Relative accuracy vs. time
    - Requires no tuning
  - Full conditionals are flat
- Future Work
  - Parallel implementation of solvers



Thank you!