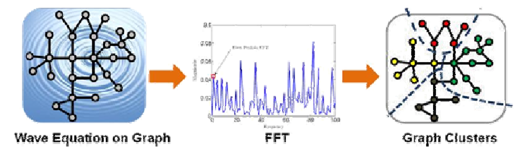
Scalable Algorithms for Complex Networks

Tuhin Sahai
United Technologies Research Center

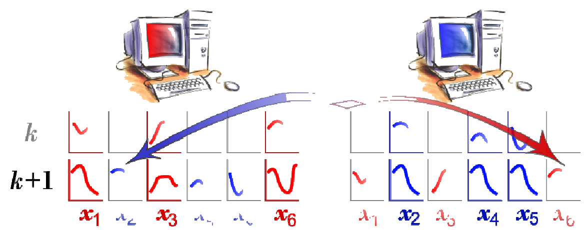
University of Connecticut

Outline / Acknowledgements

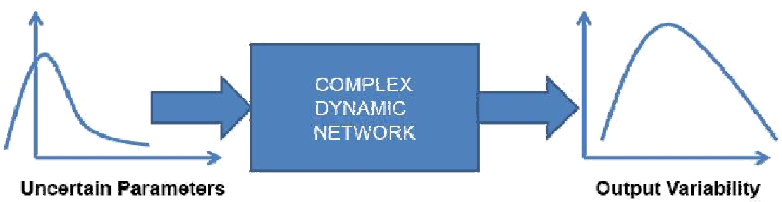
- Decentralized Clustering
  - Alberto Speranzon (UTRC)
  - Andrzej Banaszuk (UTRC)



- Distributed Computation
  - Stefan Klus (U. Paderborn/UTRC)
  - Michael Dellnitz (U. Paderborn)

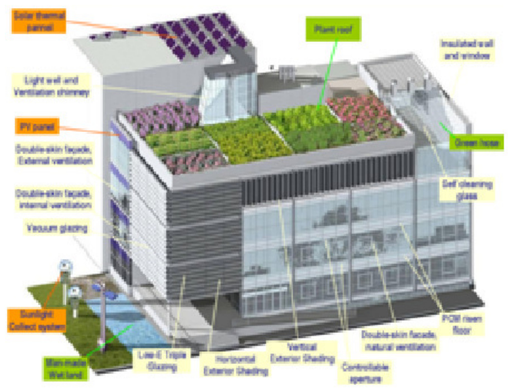


- Uncertainty Quantification
  - José Miguel Pasini (UTRC)
  - Amit Surana (UTRC)
  - Andrzej Banaszuk (UTRC)
  - Vladimir Fonoberov (Aimdyn Inc.)
  - Sophie Loire (UC Santa Barbara)



Complex Networks: Building Systems

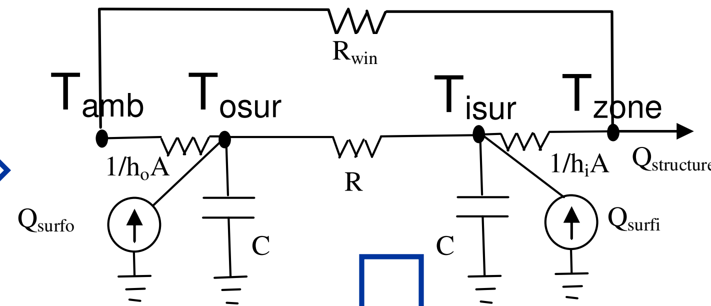
- 40% of produced energy in the US is consumed by buildings



Energy Efficient Integrated Buildings

Thermal models

CFD, Reduced Order Models

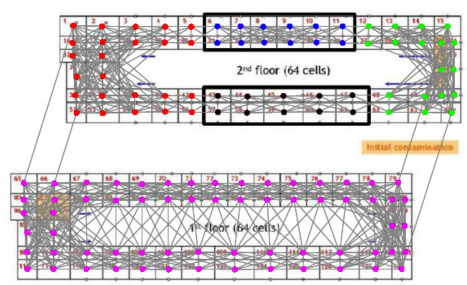


Analysis & Control

Large ODE Models

Distributed Computation
Uncertainty Quantification

Large Sensor Networks



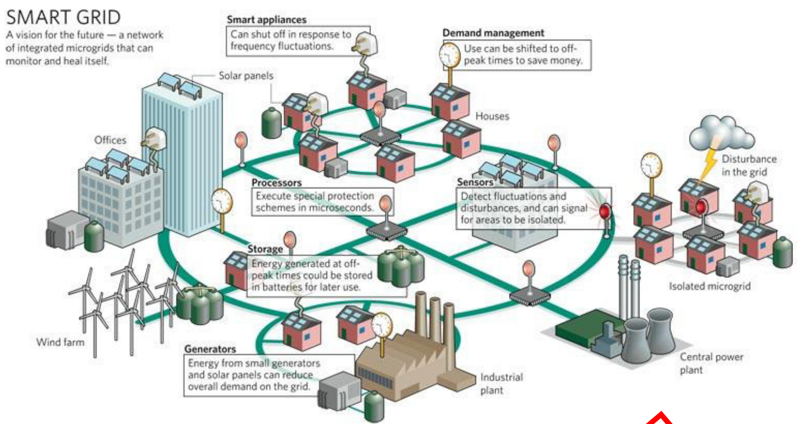
Monitoring, Security, Occupancy Estimation etc.

Distributed Estimation
Distributed Optimization

Need Scalable & Decentralized Solutions

Complex Networks: Smart Grids & Aircraft Systems

- Renewable energy sources: DOE's 2030 goal – 20% of all energy to have renewable sources



Aircraft Electrical Systems

No Central Control

Heterogeneous Switching systems

FAA Certification

Decentralized Control

Stability Analysis

- **Uncertainty Quantification**

This page contains no technical data subject to the EAR or the ITAR.

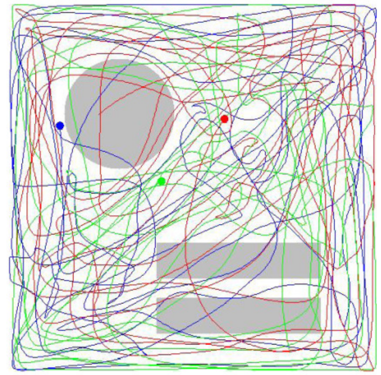
Complex Networks: UAV Swarms/Sensor Nets

- Networks of mobile sensors needed for various tasks

UAV Swarm

Path planning
for search & rescue

[Mathew and Mezic, ’09]

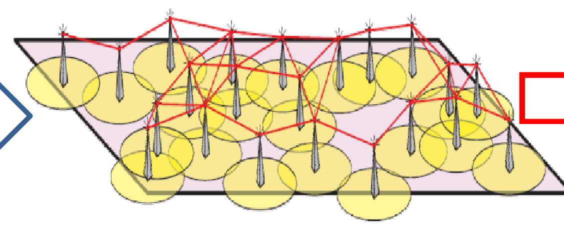


Sample Prior Distribution

Centralized:
Changing
Environment?

Sensor Coverage

[Ghrist and Jadbabaie]



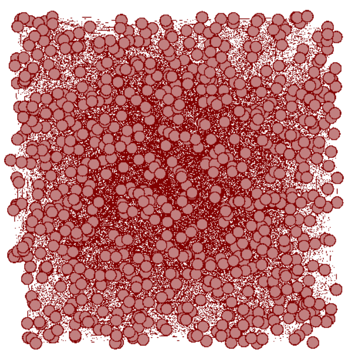
Slow:
Random Walk/
Consensus

- Novel algorithms needed for deployment in applications
- Mobile sensors need to self-organize using local information
- Role of uncertainty quantification for random parameters

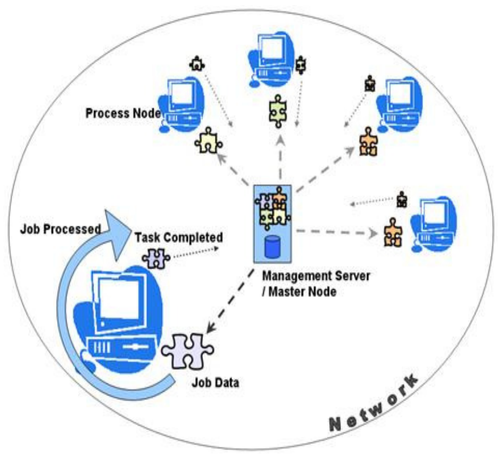
Complex Networks: Social Graphs

- Analysis of large social networks: time-varying and multi- attributed interactions

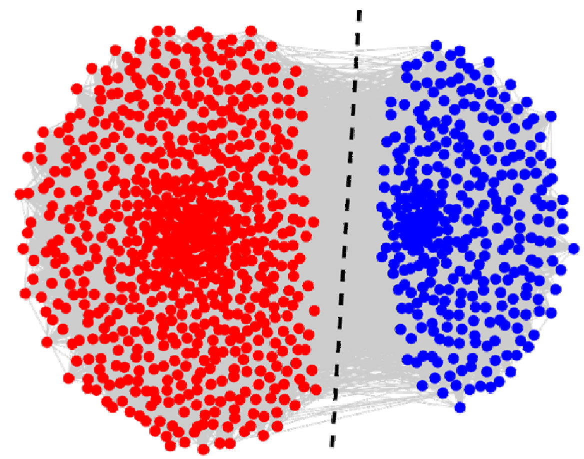
GUARD-DOG Program



Social Network



Parallel/Cloud Computation



Community detection

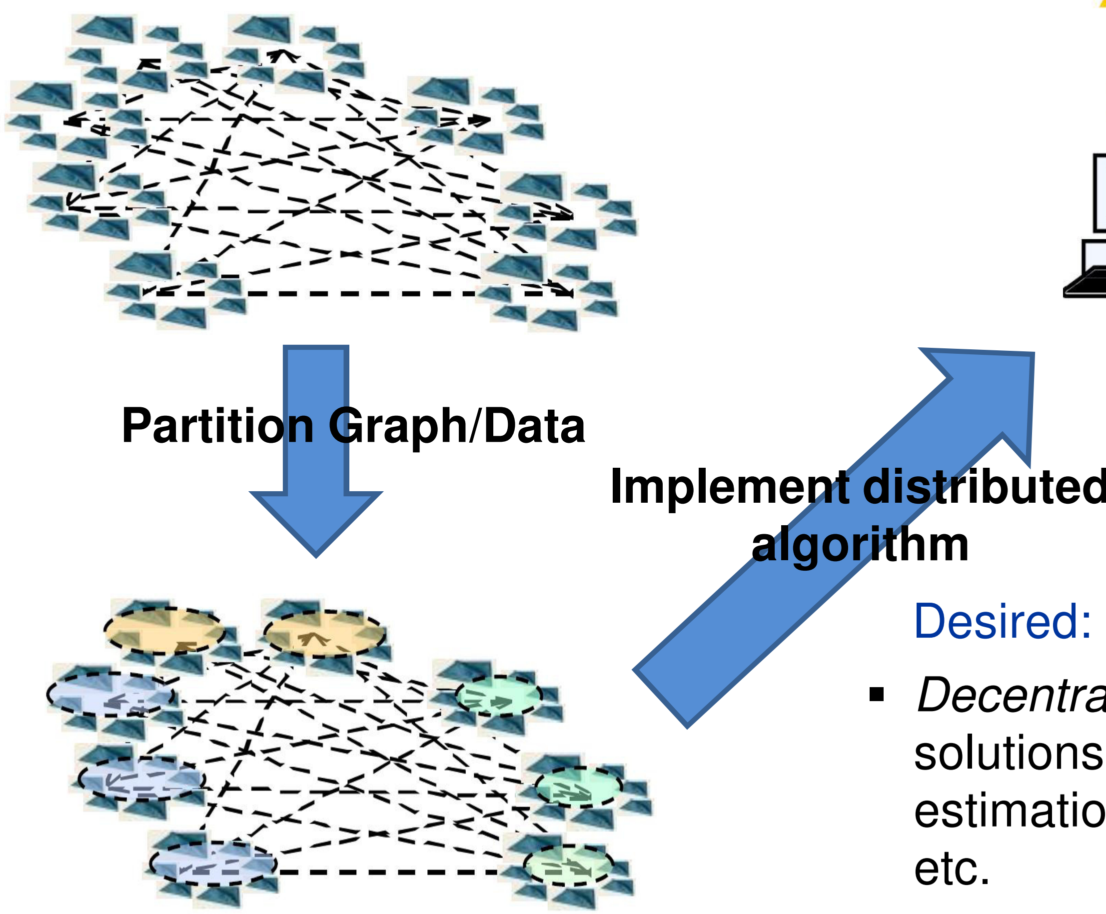
Scalable algorithms

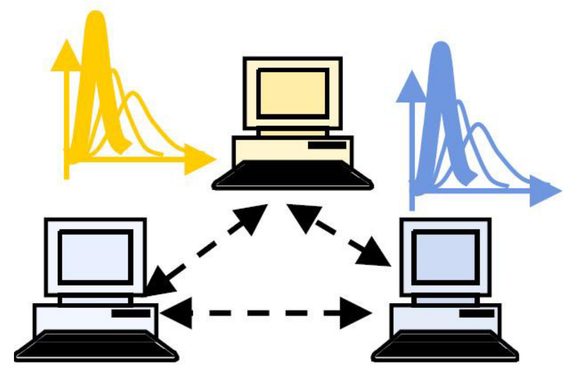
- Games on social networks

- Other important complex networks: MEMS oscillator networks, biological systems etc.

Scalable Solutions for Complex Networks

- “Divide and Conquer” - help build scalable solutions for various problems



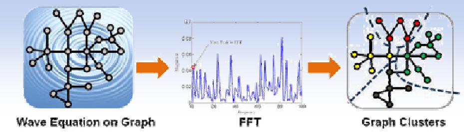


Desired:

- *Decentralized, scalable* and *fast* solutions for clustering, computation, estimation, uncertainty quantification etc.

- Clustering enables *scalability*

“Shape” from Harmonic Analysis



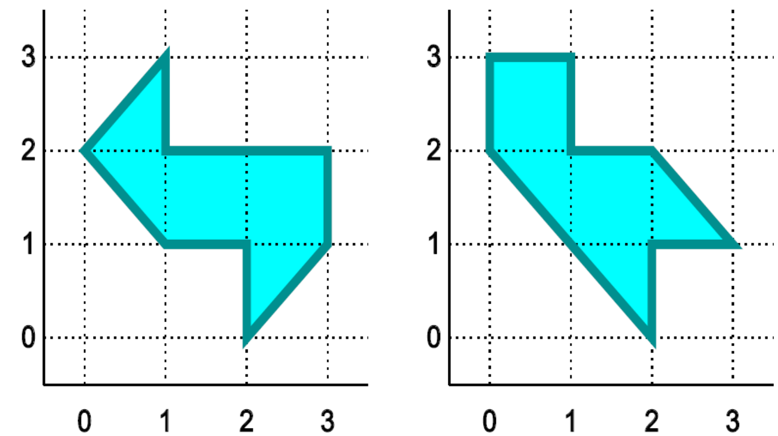
- “*Can one hear the shape of a drum?*” – an article by Mark Kac: do frequencies of vibrations determine the shape uniquely?

- Sparked decades of mathematical activity!

- *Answer*: Yes (if convexity assumed)

- Similarly we ask: Can one hear the properties of a graph (such as cluster locations)?

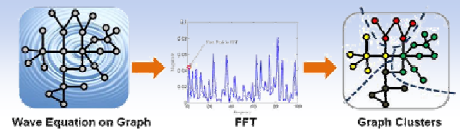
- *Answer*: Yes (if no symmetries)



Non-convex drums with same frequencies

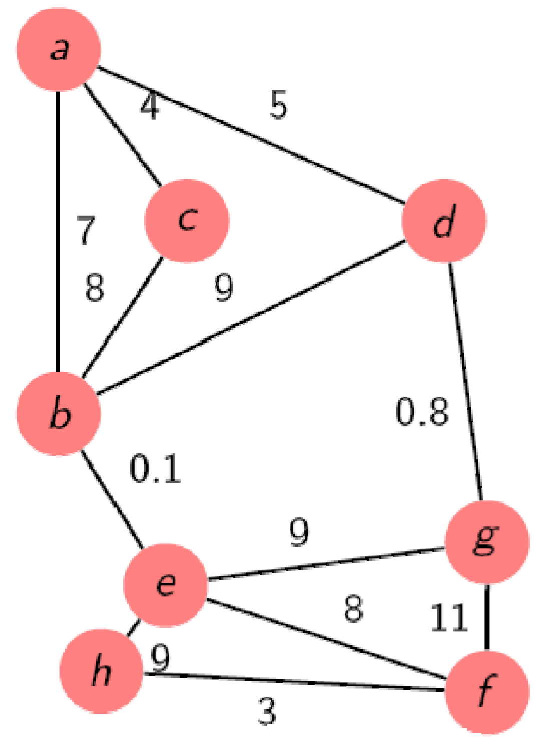
M. Kac, Can One Hear the Shape of a Drum, American Mathematical Monthly, 1966

Graph Analysis



- Matrix approach for representing graphs:

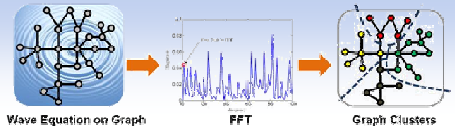
$$A = \begin{pmatrix} 0 & 7 & 4 & 5 & 0 & 0 & 0 & 0 \\ 7 & 0 & 8 & 9 & 0.1 & 0 & 0 & 0 \\ 4 & 8 & 0 & 0 & 0 & 0 & 0 & 0 \\ 5 & 9 & 0 & 0 & 0 & 0 & 0.8 & 0 \\ 0 & 0.1 & 0 & 0 & 0 & 8 & 9 & 9 \\ 0 & 0 & 0 & 0 & 8 & 0 & 11 & 3 \\ 0 & 0 & 0 & 0.8 & 9 & 11 & 0 & 0 \\ 0 & 0 & 0 & 0 & 9 & 3 & 0 & 0 \end{pmatrix} \qquad D = \begin{pmatrix} 16 & 0 & \cdots & & 0 \\ 0 & 24.1 & & & \vdots \\ \vdots & & 12 & & \\ & & & \ddots & 0 \\ 0 & \cdots & & 0 & 12 \end{pmatrix}$$



$$L = I - D^{-1}A$$

- Laplacian has nice properties
  - Eigenvalues: $0 = \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_N \leq 2$
  - Eigenvectors: $v_1 = \mathbf{1}, v_2, \ldots, v_N$
  - Eigenvalues: Connected components, diameter etc
  - Spectral Gap: Number of clusters
  - Eigenvectors: Clustering, Localization, Sensor Coverage, PageRank etc F. Chung, Spectral Graph Theory, 1997

Spectral Clustering (Centralized)



Clusters: set of nodes strongly connected to each other

- Partition graphs based on:

$$\min \sum_{i=1}^{k} \frac{\text{cut}(A_i, \overline{A}_i)}{|A_i|}$$

Strength of connection

NP-Complete

Size of clusters

where,

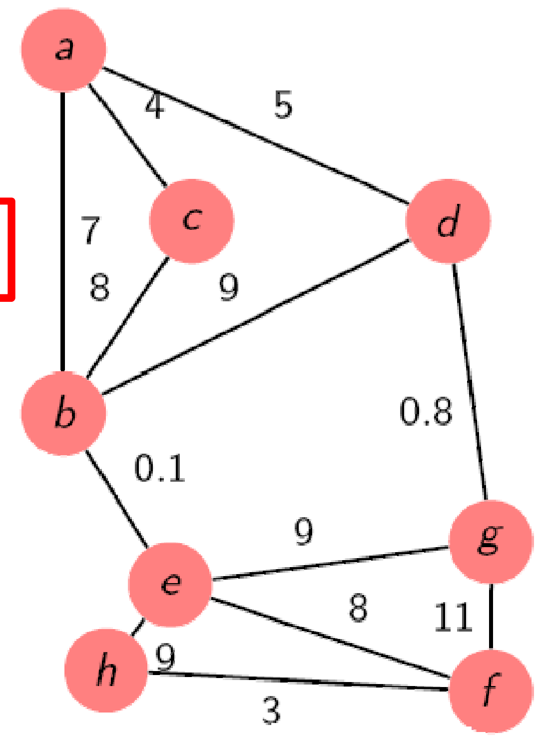
$$\text{cut}(A, \overline{A}) = \sum_{i \in A, j \in \overline{A}} w_{ij}$$

Relax problem

$$\begin{aligned} \min_{z} \quad & z^T L z \\ \text{s.t.} \quad & z^T \mathbf{1} = 0 \\ & z^T z = N \end{aligned}$$

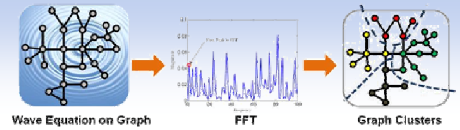
Rayleigh-Ritz

$$z = v_2$$

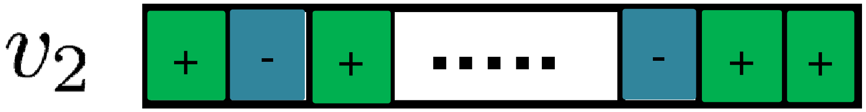


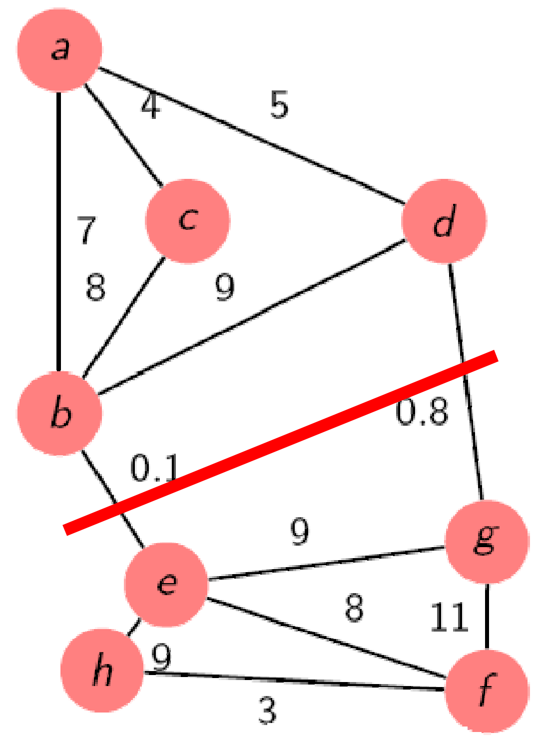
U. v. Luxburg, A Tutorial on Spectral Clustering, Statistics and Computing, 17(4), 2007

Spectral Clustering (Centralized)



Eigenvector of Laplacian holds the clustering information





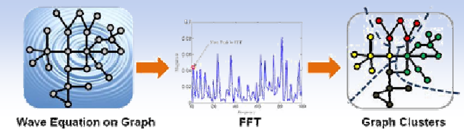
It is extended to more than 2 clusters by using higher eigenvectors (combinations of signs)

Very popular approach

Drawbacks:
- Solution is expensive for large graphs
- Central computer needed

M. Fiedler, Czechoslovak Mathematical Journal, 1975.

Distributed Spectral Clustering



- Orthogonal Iterations on a graph

Algorithm 1 Orthogonal Iteration (A)

Choose a random $n \times k$ matrix Q.
loop
    Let $V = AQ$.
    Let $Q = \text{Orthonormalize}(V)$.
end loop
Return Q as the eigenvectors.

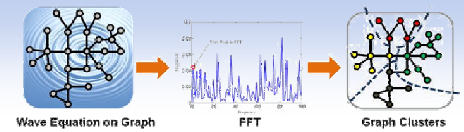
Random Walk on the graph

- Random walks can be slow – equivalent to evolving the heat equation on the graph. Convergence: $O(\tau \log^2 N)$

τ : **Measure of time for random walk to get to steady state**

D. Kempe and F. McSherry, A Decentralized Algorithm for Spectral Analysis, 2008.

Insight



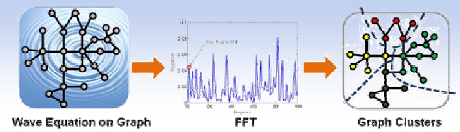
Traditional distributed clustering uses random walks

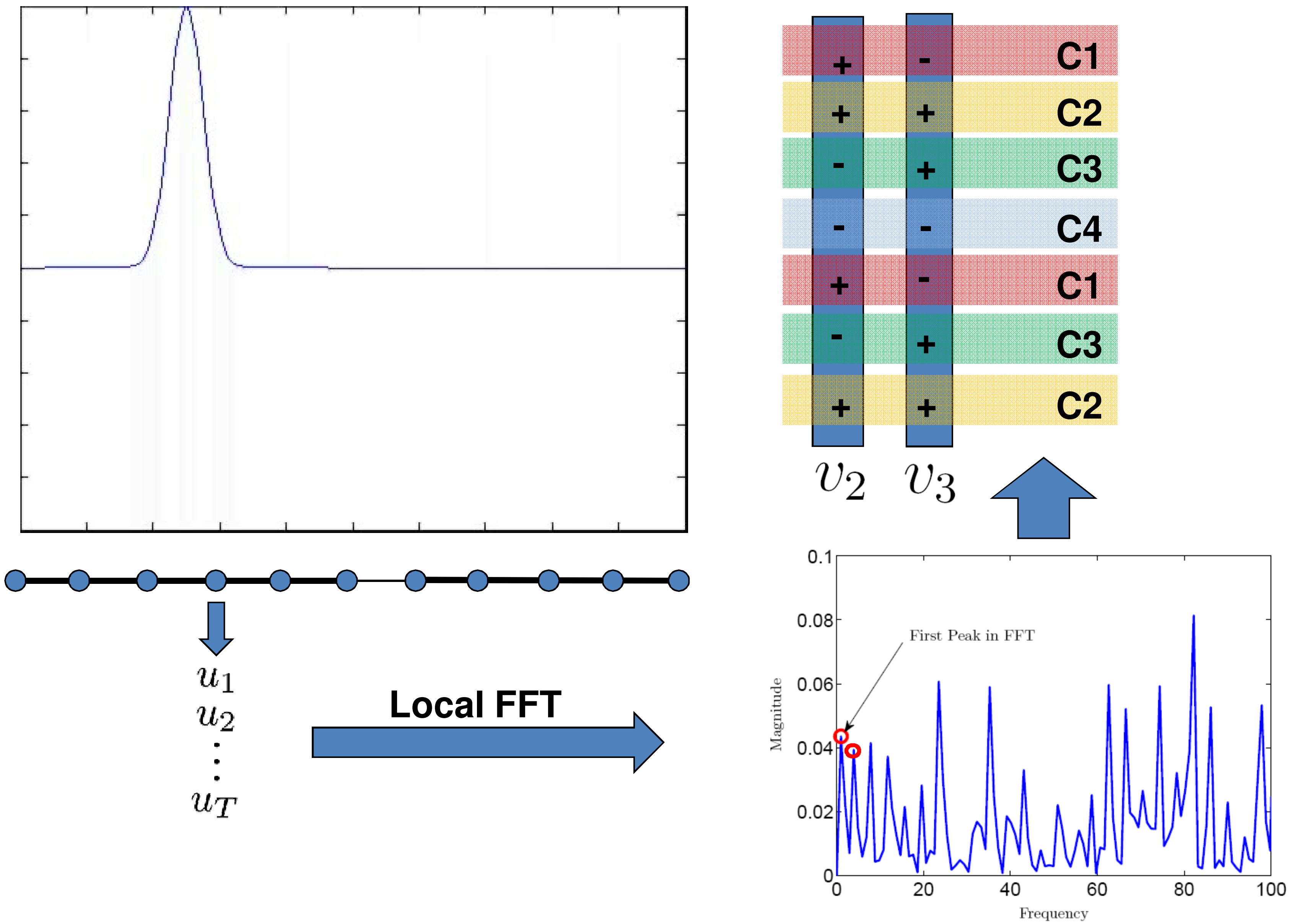
- Heat Equation (random walks): $u_t = \Delta u$
  - On the graph: $\mathbf{u}_i(t+1) = \mathbf{u}_i(t) - \sum_{j \in \mathcal{N}(i)} \mathbf{L}_{ij}\mathbf{u}_j(t)$
- Wave equation: $u_{tt} = \Delta u$
  - On the graph: $\mathbf{u}_i(t) = 2\mathbf{u}_i(t-1) - \mathbf{u}_i(t-2) - c^2 \sum_{j \in \mathcal{N}(i)} \mathbf{L}_{ij}\mathbf{u}_j(t-1)$

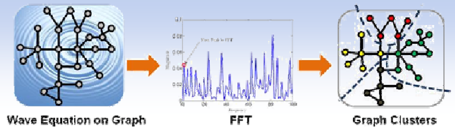
The solution of the heat equation dies out, but the wave equation does not

- M. Hein, J.-Y. Audibert and U. V. Luxburg, From Graphs to Manifolds - Weak and Strong Pointwise Consistency of Graph Laplacians, 2005.
- T. Sahai, A. Speranzon and A. Banaszuk, Hearing the Clusters in a Graph: A Distributed Algorithm, Automatica, 2012.

Example of the Algorithm







Main Result

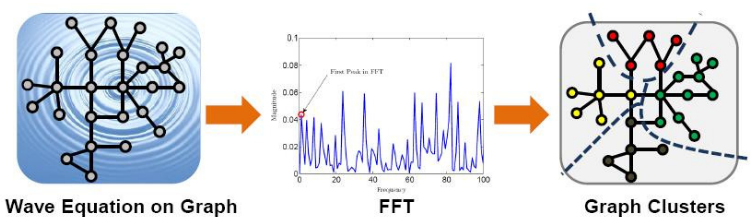
Proposition

The eigenvalues and eigenvectors of the graph Laplacian L can be computed using the frequencies and coefficients of the Fast Fourier Transform of $(u_i(1), \ldots, u_i(T))$*, for any* i*, where* u_i *is governed by the wave equation on the graph*

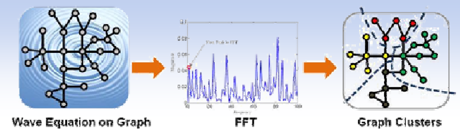
$$u_i(t) = 2u_i(t-1) - u_i(t-2) - c^2 \sum_{j \in \mathcal{N}(i)} L_{ij} u_j(t-1),$$

with the initial condition $u(-1) = u(0)$.

Evolving the wave equation combined with local frequency estimation is equivalent to computing the eigenvectors of the graph Laplacian



Proof Outline



Proof.

Discretize wave equation at each node,

$$u_i(t) = 2u_i(t-1) - u_i(t-2) - c^2 \sum_{j \in \mathcal{N}(i)} L_{ij} u_j(t-1)\,.$$

In matrix form,

$$\begin{pmatrix} u_t \\ u_{t-1} \end{pmatrix} = \begin{pmatrix} 2I - c^2 L & -I \\ I & 0 \end{pmatrix} \begin{pmatrix} u_{t-1} \\ u_{t-2} \end{pmatrix}.$$

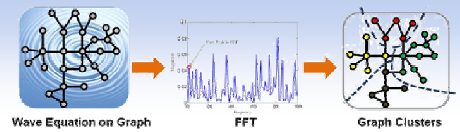
The solution of the above equation,

$$u(t) = \sum_j \left[C_{j1} v^{(j)} \cos(t\omega_j) + C_{j2} v^{(j)} \sin(t\omega_j) \right]$$

FFT coefficients at node i depend on v_i^j

Need to start with a random initial condition

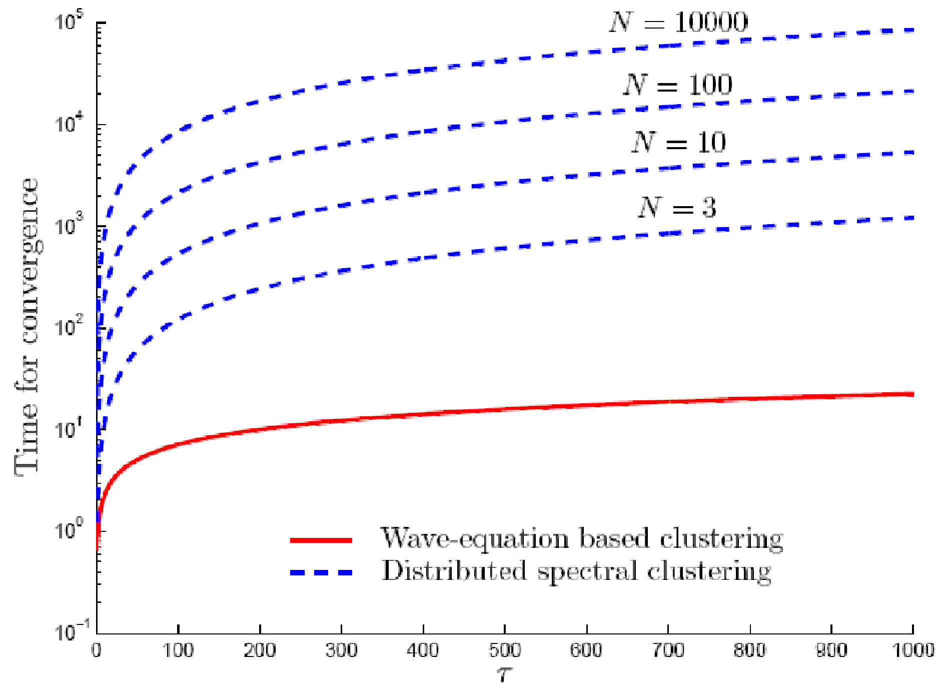
Performance vs. Random Walk



- It can be shown that:

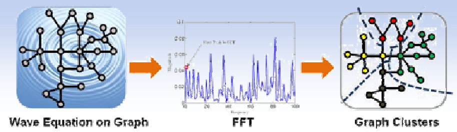
$$\boxed{T_{max} = O\left(\frac{\sqrt{\tau}}{c}\right)}$$

- Time needed to resolve the lowest frequency in the FFT



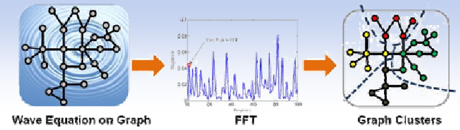
- Compare to $O(\tau \log^2 N)$ [Kempe & McSherry, 2008]
- Orders of magnitude faster than random walks
- τ is a function of N

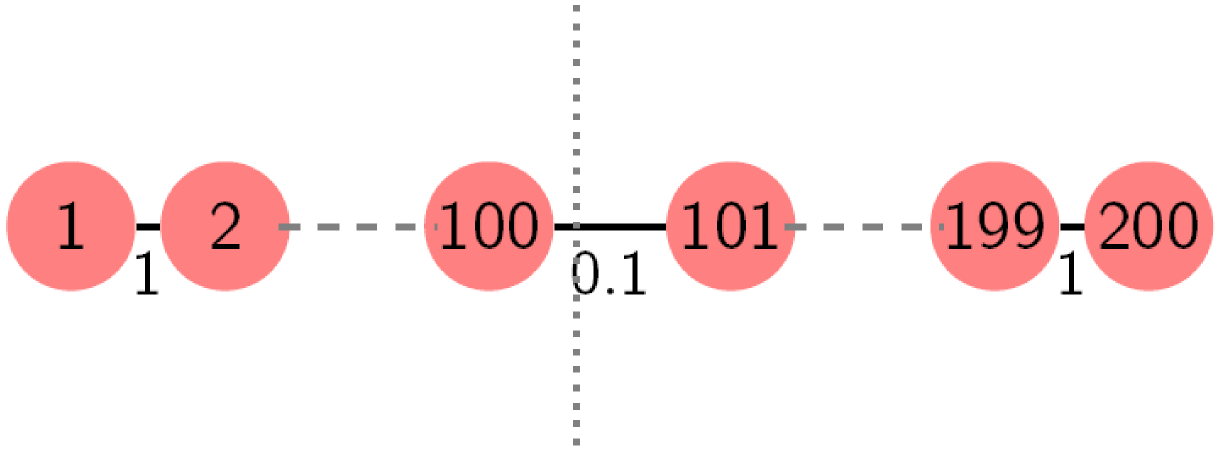
Comments about the Wave Equation

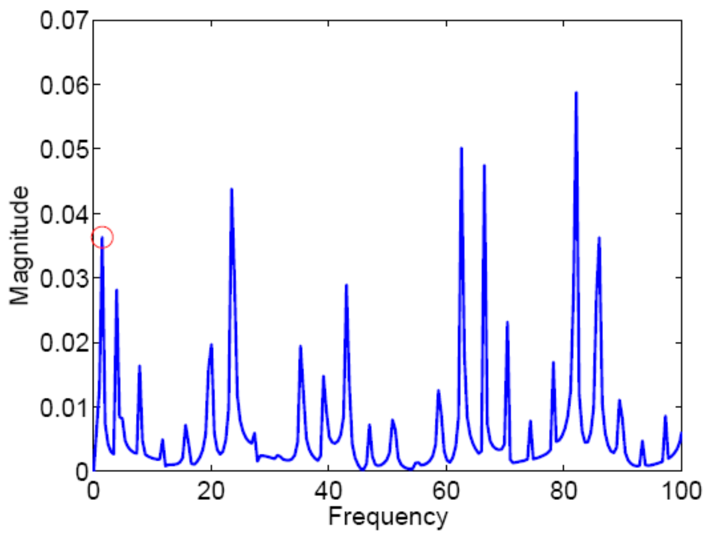


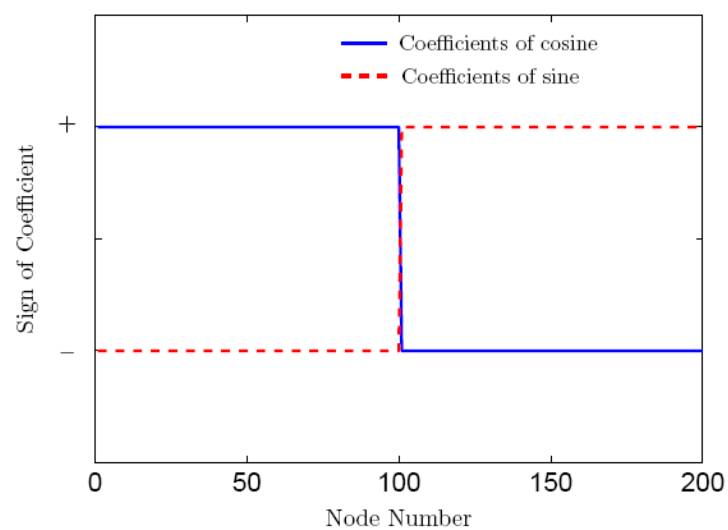
- Computes all eigenvectors and eigenvalues of the Laplacian
- Useful for distributed graph analysis and self-organizing networks
- Algorithm is decentralized (no central node required)
- Fast Convergence (faster than random-walk based methods)
- Only scalar quantities exchanged (low communication cost)

Results: Line Graph

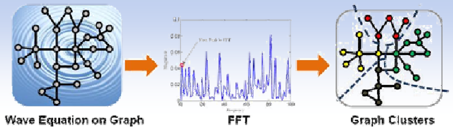




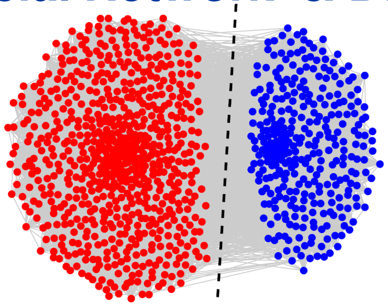




Applications & Results



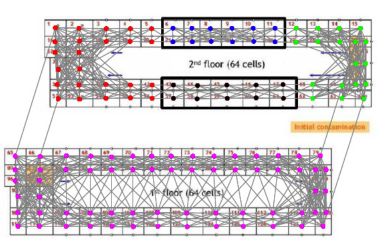
Social Network & Data Analysis

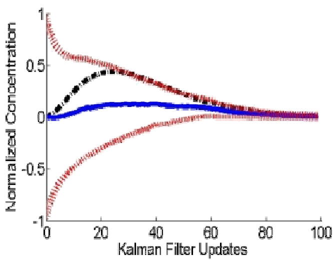


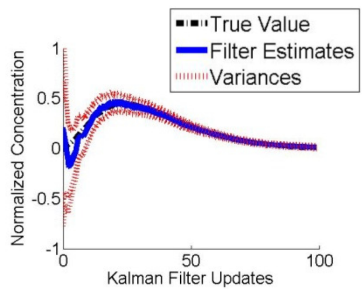
- Community detection in MapReduce (DARPA)
- Algorithm to detect disconnected components and isolated nodes
- Scalable analysis of large quantities of sensor data

Fortunato Benchmark Example

Distributed Estimation



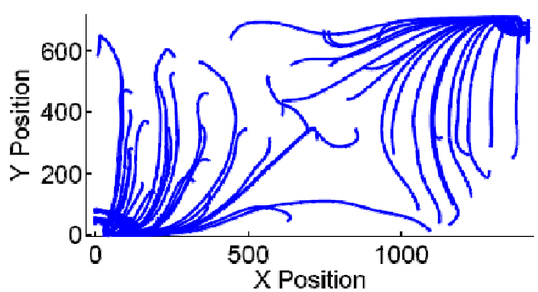




- Sensors self-organize into groups
- Accelerates distributed estimation by orders of magnitude

Contaminant Flow in Buildings

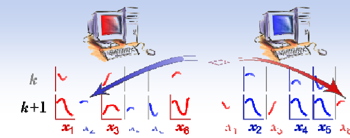
Decentralized Path Planning



- Extends work by Mathew and Mezic 2009
- Mobile agents self-organize into groups
- Decentralized computation of trajectories

T. Sahai, A. Speranzon and A. Banaszuk, Hearing the Clusters in a Graph: A Distributed Algorithm

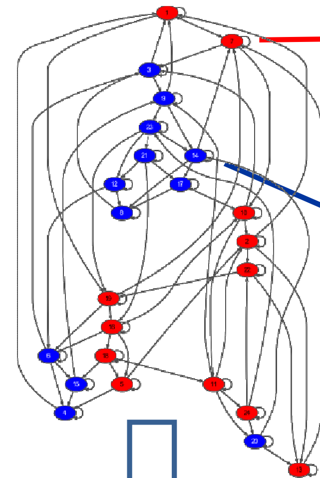
Distributed Computation



- **Waveform Relaxation:** Algorithm for distributed simulation of differential equations on parallel computers

$$
\begin{aligned}
\dot{x}_1 &= f_1(x_1, x_2, \ldots, x_n),\\
\dot{x}_2 &= f_2(x_1, x_2, \ldots, x_n),\\
&\vdots\\
\dot{x}_n &= f_n(x_1, x_2, \ldots, x_n).
\end{aligned}
$$

Split

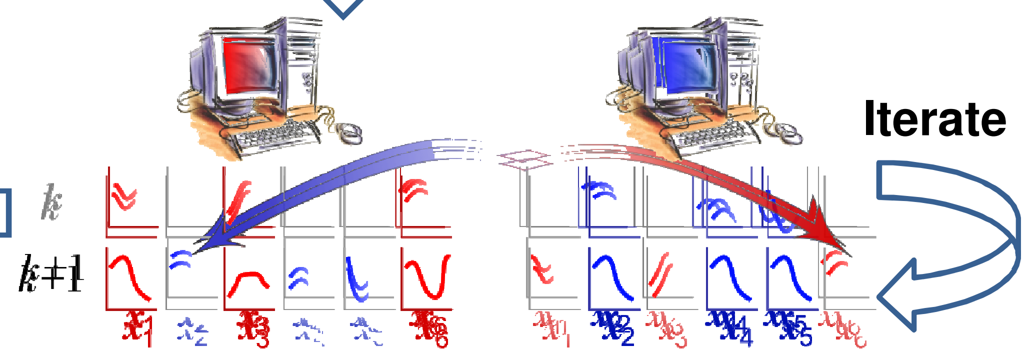


Variables in CPU #1

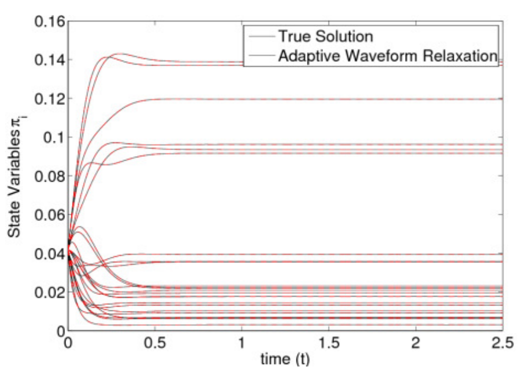
Variables in CPU #2

Goal: Simulate for the interval $[0, T]$

Compute

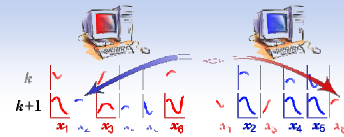


Iterate



J. K. White and A. Sangiovanni-Vincentelli, Relaxation Techniques for the Simulation of VLSI Circuits, 1986.

Distributed Computation



Theorem

The error at the k-th iterate $|E_k|$ for standard waveform relaxation, on the interval $[0, T]$, is bounded by

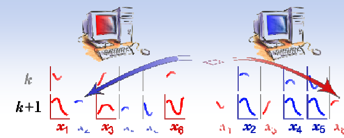
$$|E_k(t)| \leq \frac{C^k \eta^k T^k}{k!} |E_0(t)|,$$

under the assumption that E_k is C^1 $\forall$ k. Here $C = e^{\mu T}$, where μ and η are Lipschitz constants.

Waveform Relaxation always converges as long as the original function is Lipschitz continuous

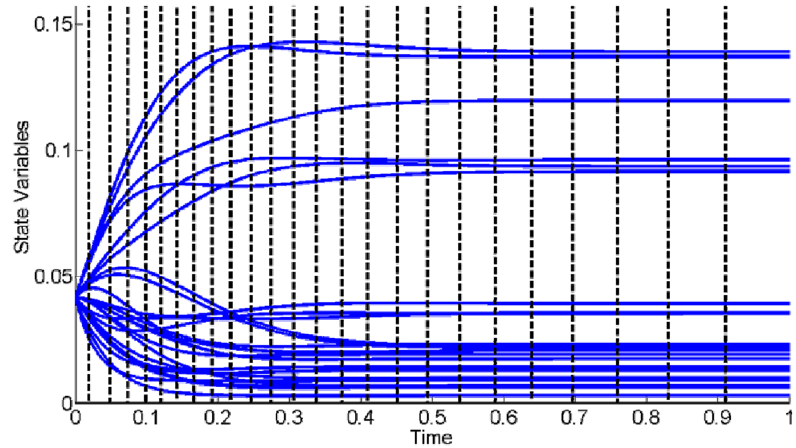
S. Klus, T. Sahai, C. Liu and M. Dellnitz (2011), An Efficient Algorithm for the Parallel Solution of High-Dimensional Differential Equations, Journal of Computational and Applied Mathematics.

Adaptive Waveform Relaxation



- Split $[0, T]$ into parts

- Reduce $|E_k(t)|$ by reducing ΔT when estimate for $|E_0(t)|$ is large



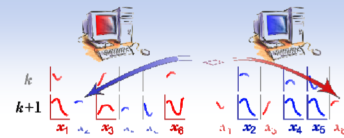
Take large steps when function changes slowly and small steps when function changes fast

Waveform relaxation can be represented as: $\dot{x}^{k+1} = \phi(x^{k+1}, x^k)$

Adaptive Waveform Relaxation

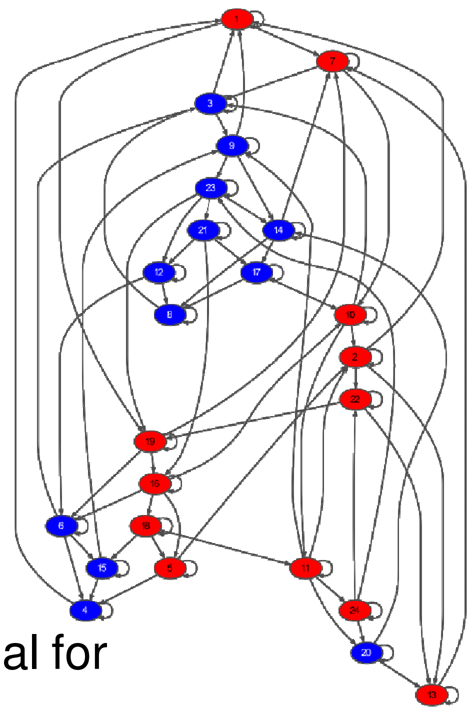
- Start with an initial time interval of say: $\frac{T}{40}$
- Compute the solution using waveform relaxation
- Let p be the number of desired iterations and ϵ the desired tolerance
- Estimate the next time interval: $\Delta T_k = \frac{1}{C\eta}\sqrt[p]{\frac{\epsilon p!}{|E_0|}}.$
- Where $E_0(t)$ is estimated using a standard extrapolation formula

Optimal Splitting for AWR



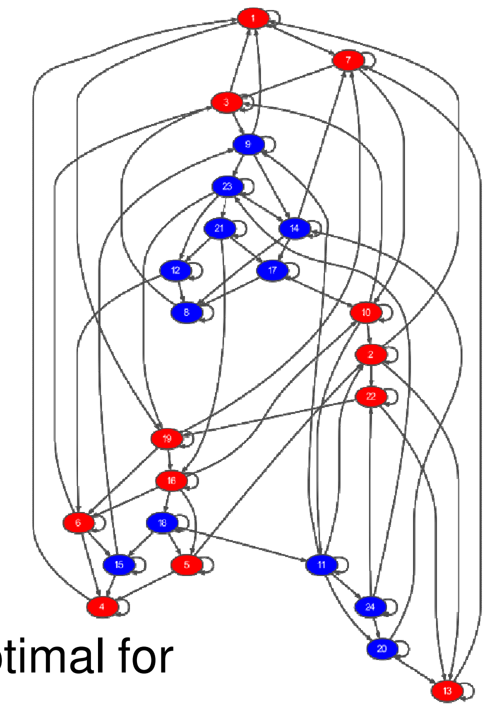
- Optimal splitting changes with the numerical scheme and step size: NP complete

Spectral Clustering is a good heuristic for splitting (symmetrized Jacobian)



Optimal for

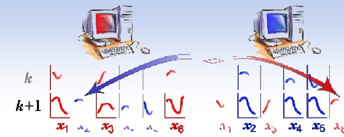
- Implicit Euler: $h = 10^{-1}$
- Implicit Euler: $h = 0.05$



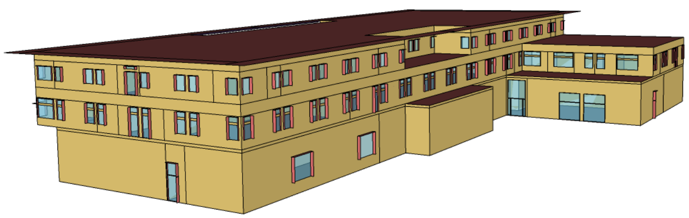
Optimal for

- Implicit Euler: $h = 10^{-2}$
- Trapezoidal Rule: $h = 0.05$

Thermal Model of a Building

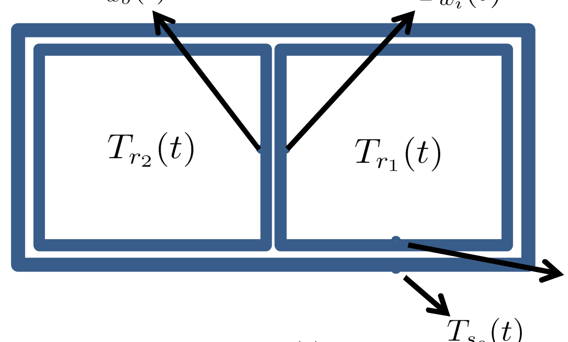


Energy consumption in Buildings



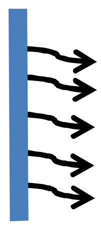
Equations for Rooms & Walls

$$C_w \dot{T}_{wall} = \dot{Q}_{wall}$$
$$m_{air} C_p \dot{T}_r = \dot{Q}_{room}$$



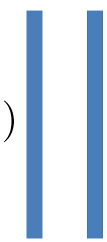
$\dot{Q}_{wall}$

Newton's Law of Cooling

$T_{wall}(t)$  $T_r(t)$

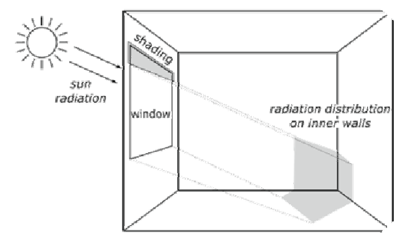
$$\dot{Q}_n(t) = hA(T_r(t) - T_{wall}(t))$$

Conduction

$T_1(t)$  $T_2(t)$

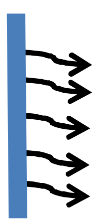
$$\dot{Q}_c(t) = \frac{T_2(t) - T_1(t)}{R}$$

Solar



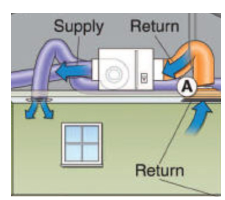
$\dot{Q}_{ext}(t)$

$\dot{Q}_{room}$

$T_{wall}(t)$  $T_r(t)$

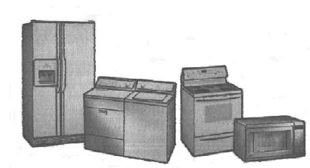
$$\dot{Q}_n(t) = hA(T_{wall}(t) - T_r(t))$$

HVAC



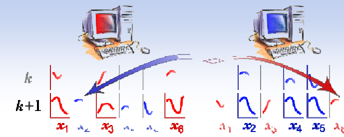
$$\dot{Q}_h = \dot{m}_h(t)(T_h(t) - T_r(t))$$

Appliances/People

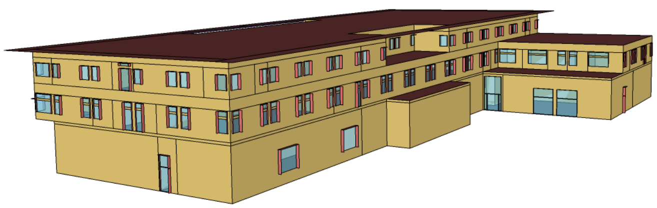


$\dot{Q}_{int}(t)$

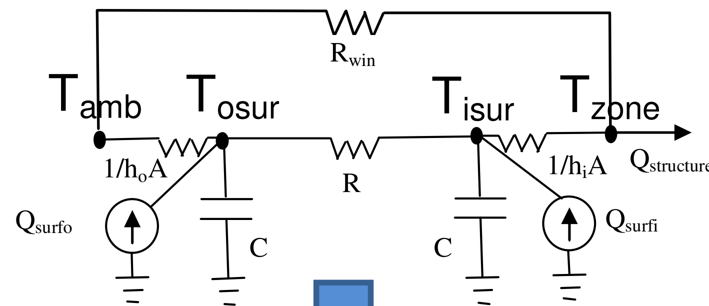
Thermal Model of a Building



Energy consumption in Buildings



Models

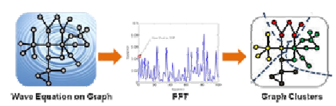


ODEs

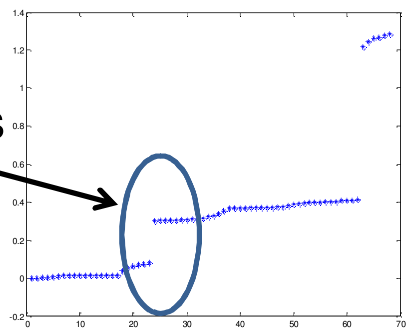
$$\frac{dT}{dt} = AT + BQ$$

1 floor gives 140 states

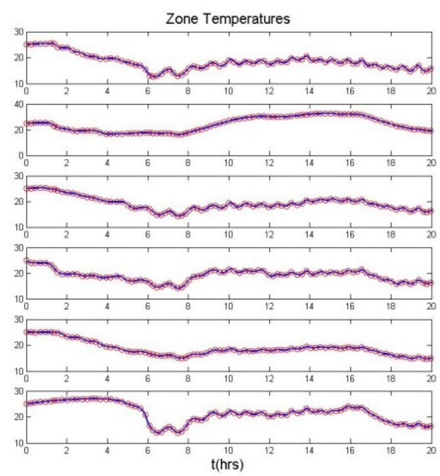
Clustering



23 Clusters



AWR & WR

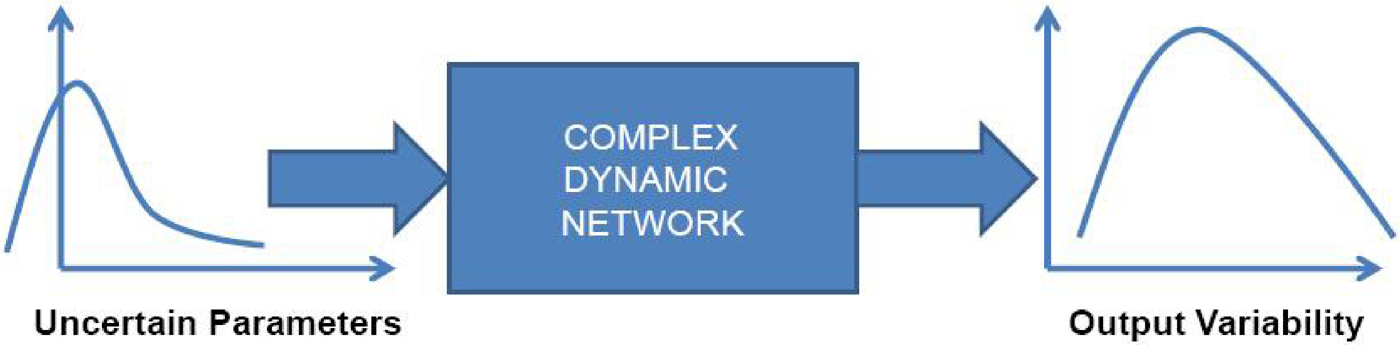


AWR 10 times faster than WR

This page contains no technical data subject to the EAR or the ITAR.

Propagating Uncertainty through Complex Networks

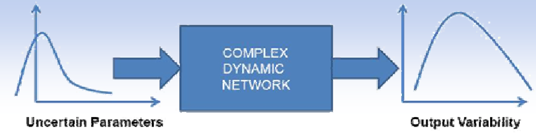
Given input (parametric) uncertainty, quantify the output uncertainty:



- Monte Carlo
- Quasi-Monte Carlo
- Response Surface Methods
- Polynomial Chaos/Probabilistic Collocation Methods
- Combination of the above

T. Sahai, V. Fonoberov and S. Loire (2010), Uncertainty as a stabilizer of the head-tail ordered phase in carbon-monoxide monolayers on graphite, Physical Review B.

Polynomial Chaos



Starting with the system: $\dot{x}(t,\lambda) \;=\; f(x(t,\lambda),\lambda,t)$ $\quad w(\lambda)$: distribution on λ

Expand the state variables (or outputs) as:

$$x_i(t,\lambda) \;=\; \sum_{k=0}^{P} a_k^i(t) H_k(\lambda)$$

Orthogonal Polynomials

$$\int_\Gamma H_i(\lambda) H_j(\lambda) w(\lambda) d\lambda \;=\; \delta_{ij}$$

Performing a Galerkin Projection gives:

$$\dot{a}_k^i(t) \;=\; \int_\Gamma H_k(\lambda) f_i(x(t,\lambda),\lambda,t) w(\lambda) d\lambda$$

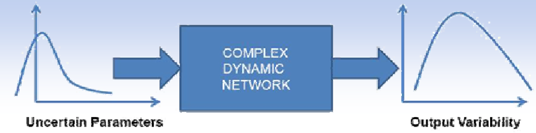
Determining $a_k(t)$ determines $x(t,\lambda)$

Exponential convergence rate for processes with finite variance.

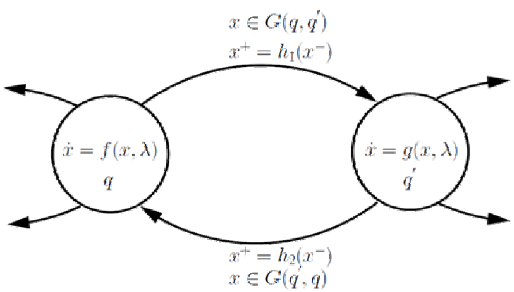
Curse of dimensionality: $n\dfrac{(N+P)!}{N!P!}$

X. Wan and G. E. Karniadakis (2008), Recent Advances in Polynomial Chaos Methods and Extensions

UQ for Hybrid Systems



- H-PC + Wavelet expansions + Boundary Layers help deal with hybrid dynamical systems – curse of dimensionality



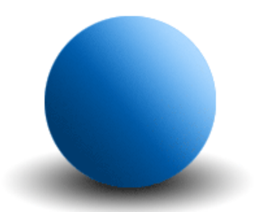
Hybrid dynamics
λ: Uncertain

H-PC: $\dot{x} = \mathbf{1}_{R_1}(x) f(x, \lambda) + \mathbf{1}_{R_2}(x) g(x, \lambda)$

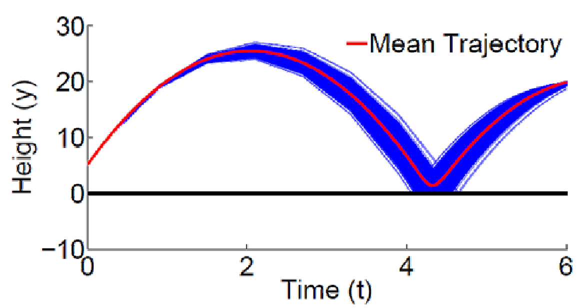
$+$

$$x(t; \lambda) = x_0(t) + \boxed{\sum_{j=0}^{P} \sum_{k=0}^{2^j - 1}} x_{jk}(t) \psi_{jk}(u(\lambda))$$

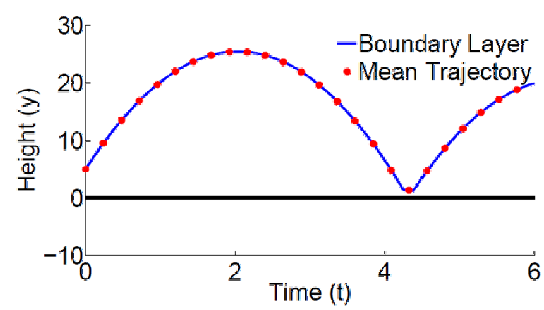
Wavelets



Bouncing Ball



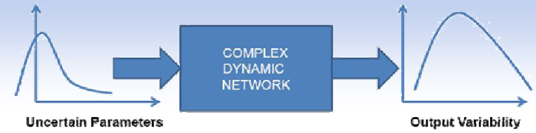
Monte Carlo



Boundary Layer

T. Sahai and J. M. Pasini, Uncertainty Quantification in Hybrid Dynamical Systems, 2013

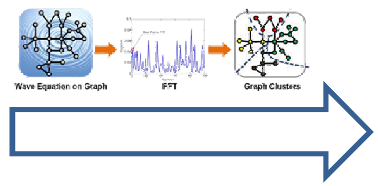
Scalable Uncertainty Quantification



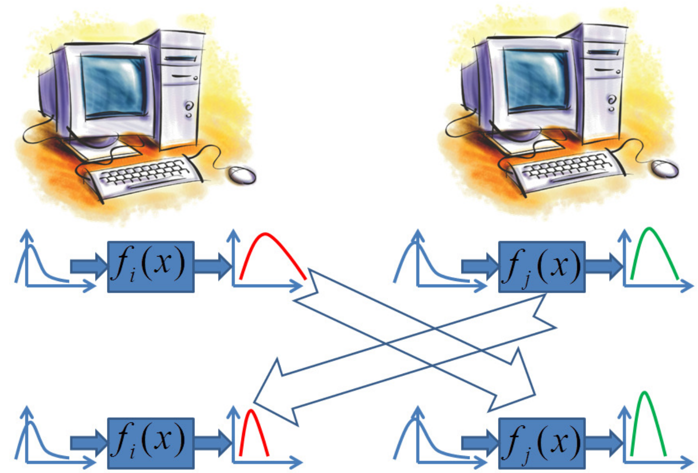
- Use decentralized clustering to partition the dynamical system into “weakly” interacting components

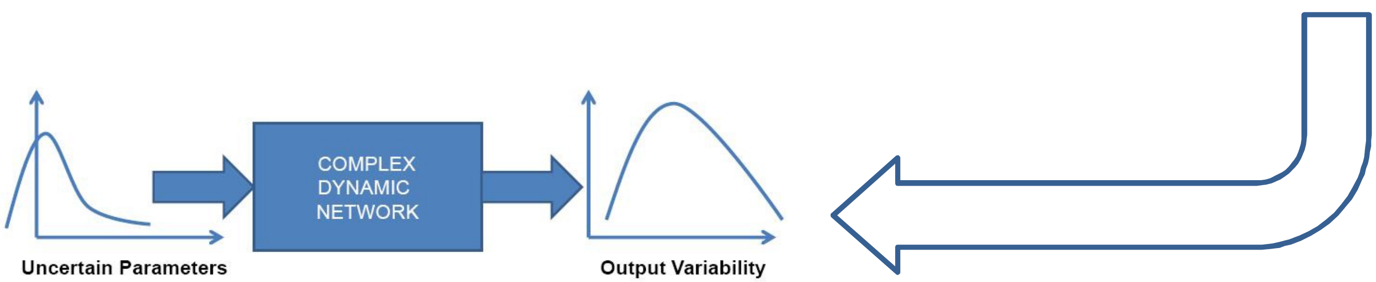
$$
\begin{aligned}
\dot{x}_1 &= f_1(x_1,\ldots,\mathbf{x}_n,\lambda_1,\ldots,\lambda_k),\\
\dot{x}_2 &= f_2(x_1,\ldots,\mathbf{x}_n,\lambda_1,\ldots,\lambda_k),\\
&\vdots\\
\dot{x}_n &= f_2(x_1,\ldots,\mathbf{x}_n,\lambda_1,\ldots,\lambda_k).
\end{aligned}
$$

Random Parameters



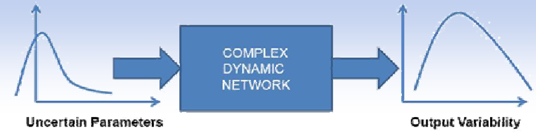
Weak interactions – lower order expansions



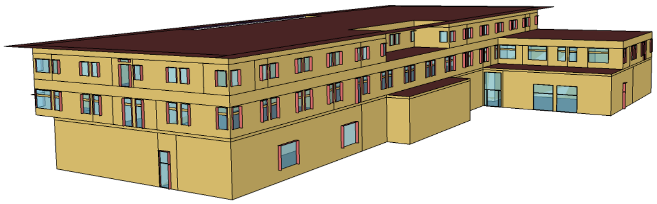


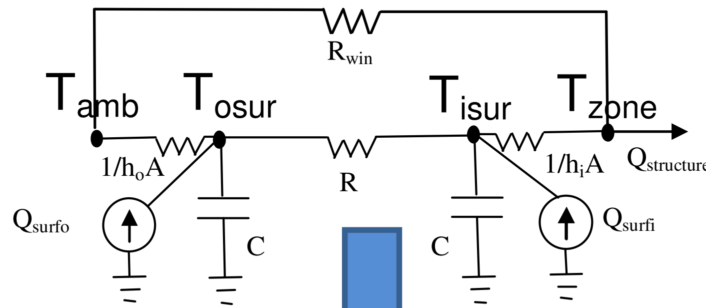
A. Surana, T. Sahai and A. Banaszuk (2012), Iterative Methods for Scalable Uncertainty Quantification in Complex Networks, International Journal of Uncertainty Quantification.

Results: Scalable UQ



- Consider a two room model with uncertain parameters





Random parameters – Gaussian distributions

$$T_i(t,\lambda) \approx \sum_{k=0}^{P} a_k^i(t) H_k(\lambda)$$

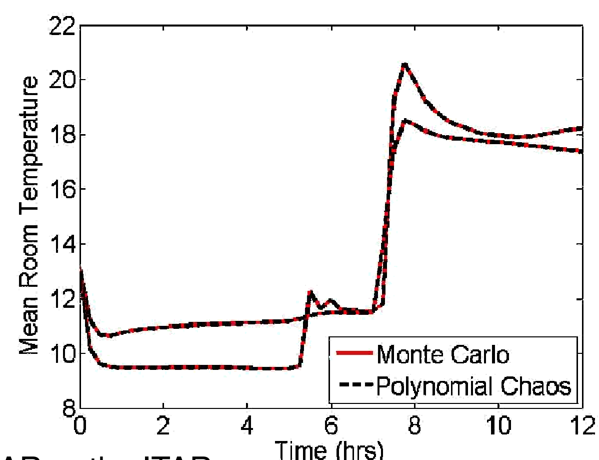
$$\frac{dT}{dt} = AT + BQ$$

2 rooms give a 10 state model

$$\frac{da}{dt} = Ca + D$$

60 state model

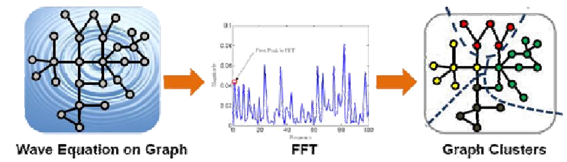
Clustering + AWR



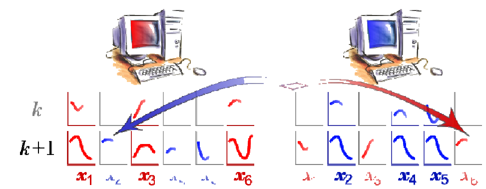
This page contains no technical data subject to the EAR or the ITAR.

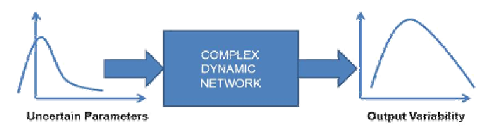
Conclusions/Future Work

- Motivated from continuous approaches one can construct efficient approaches for NP-hard problems such as graph partitioning



- Scalability is enabled by graph decomposition
  - Simulating high-dimensional dynamical systems
  - Polynomial chaos based uncertainty quantification in complex networks





- Future Directions include:
  - Distributed Computation: Index 2 DAEs, Equation-Free Methods
  - Clustering of time varying multi-attributed graphs for distributed computation, sensor coverage and localization
  - Machine Learning for “Big-Data” problems

Thank You!