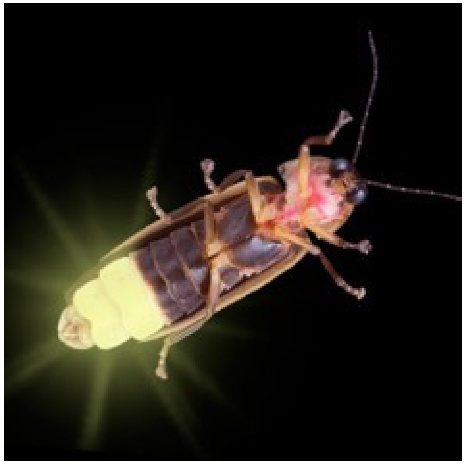
Computational Algebraic Geometry Methods with Applications to Synchronization and Power Flow Equations

Dhagash Mehta

United Technologies Research Center

PS: This document contains no technical data subject to the EAR or the ITAR.

Complex Systems: Synchronization



Fireflies at the Smoky Mountains (Gatlinburg, Tennessee, USA).
Courtesy: www.gatlinburgtnguide.com

Complex Systems: Synchronization

Fireflies at the Smoky Mountains (Gatlinburg, Tennessee, USA).
Courtesy: www.gatlinburgtnguide.com

Complex Systems: Synchronization

Rhythmic applause

Complex Systems: Synchronization

Power networks and electrical grids.

Complex Systems: Synchronization

Neural network synchronization

Complex Systems: Synchronization

The Kuramoto Model:

$$\frac{d\theta_i}{dt} = \omega_i + \frac{K}{N}\sum_{j=1}^{N}\sin(\theta_i - \theta_j), \text{ for } i = 1, \ldots, N$$

ω_i are normal frequencies

i.e. the frequency without the presence of the sine terms.

Complex Systems: Synchronization

The Kuramoto Model:

$$\frac{d\theta_i}{dt} = \omega_i + \frac{K}{N}\sum_{j=1}^{N}\sin(\theta_i - \theta_j), \ \text{for } i = 1, \dots, N$$

ω_i are normal frequencies

i.e. the frequency without the presence of the sine terms.

- Each oscillator (firefly) *knows* what all other oscillators are doing, called the complete graph.

- One can also have other more realistic graphs, e.g., random, cyclic, small-world, etc.

Complex Systems: Synchronization

The Kuramoto Model:

$$\frac{d\theta_i}{dt} = \omega_i + \frac{K}{N}\sum_{j=1}^{N}\sin(\theta_i - \theta_j), \text{ for } i = 1, \ldots, N$$

ω_i are normal frequencies

i.e. the frequency without the presence of the sine terms.

- 'K' is the *strength/amount of knowledge* about other oscillators.

- In this set up, each oscillator has the same amount of knowledge about others as all others.

- One can also have a setup with different values of K for each pair of oscillators and so on.

Complex Systems: Synchronization

The Kuramoto Model:

$$\frac{d\theta_i}{dt} = \omega_i + \frac{K}{N}\sum_{j=1}^{N}\sin(\theta_i - \theta_j), \text{ for } i = 1, \ldots, N$$

- when K=0, all oscillators oscillate with their natural frequencies

- increasing K from 0, the oscillators start working together

- but only at a particular value of K, they are synchronized

Complex Systems: Synchronization

The Kuramoto Model:

$$\frac{d\theta_i}{dt} = \omega_i + \frac{K}{N}\sum_{j=1}^{N}\sin(\theta_i - \theta_j), \text{ for } i = 1, \ldots, N$$

- when K=0, all oscillators oscillate with their natural frequencies

- increasing K from 0, the oscillators start working together

- but only at a particular value of K, they are synchronized

$$K < K_c, \text{ no synchronization}$$

$$K \geq K_c, \text{ synchronization}$$

Complex Systems: Synchronization

The Kuramoto Model:

$$\frac{d\theta_i}{dt} = \omega_i + \frac{K}{N}\sum_{j=1}^{N}\sin(\theta_i - \theta_j), \text{ for } i = 1, \ldots, N$$

- Many exact results available for $N \to \infty$

Complex Systems: Synchronization

The Kuramoto Model:

$$\frac{d\theta_i}{dt} = \omega_i + \frac{K}{N} \sum_{j=1}^{N} \sin(\theta_i - \theta_j), \;\; \text{for } i = 1, \ldots, N$$

- Many exact results available for $N \to \infty$

- But the finite size case has been very difficult so far, though it is more realistic: finite no. of fireflies, neurons, nodes in the power networks, etc.

Complex Systems: Synchronization

The Kuramoto Model:

$$\frac{d\theta_i}{dt} = \omega_i + \frac{K}{N}\sum_{j=1}^{N}\sin(\theta_i - \theta_j), \ \text{for } i = 1, \ldots, N$$

- Many exact results available for $N \rightarrow \infty$

- But the finite size case has been very difficult so far, though it is more realistic: finite no. of fireflies, neurons, nodes in the power networks, etc.
- For finite N: run the dynamical system (solve the ODEs from random initial conditions)

Complex Systems: Synchronization

The Kuramoto Model:

$$\frac{d\theta_i}{dt} = \omega_i + \frac{K}{N}\sum_{j=1}^{N}\sin(\theta_i - \theta_j), \text{ for } i = 1, \ldots, N$$

- Many exact results available for $N \to \infty$

- But the finite size case has been very difficult so far, though it is more realistic: finite no. of fireflies, neurons, nodes in the power networks, etc.
- For finite N: run the dynamical system (solve the ODEs from random initial conditions)

Problems: 1. Dependent on initial conditions
2. multiple stable steady states
3. Dependent on step size
4. No stable steady state?

Complex Systems: Synchronization

The Kuramoto Model:

$$\frac{d\theta_i}{dt} = \omega_i + \frac{K}{N}\sum_{j=1}^{N}\sin(\theta_i - \theta_j), \text{ for } i = 1, \ldots, N$$

A different mathematical set up of the problem:

- **Find the first instance of K, starting from K=0, for which the below system has at least one stable steady state.**

$$\omega_i + \frac{K}{N}\sum_{j=1}^{N}\sin(\theta_i - \theta_j) = 0, \text{ for } i = 1, \ldots, N$$

Complex Systems: Synchronization

The Kuramoto Model (power flow equations for lossless network with all nodes being PV nodes):

$$\omega_i + \frac{K}{N}\sum_{j=1}^{N}\sin(\theta_i - \theta_j) = 0, \text{ for } i = 1, \ldots, N$$

Complex Systems: Synchronization

The Kuramoto Model:

$$\omega_i + \frac{K}{N}\sum_{j=1}^{N}\sin(\theta_i - \theta_j) = 0, \text{ for } i = 1, \ldots, N$$

Complex Systems: Synchronization

The Kuramoto Model:

$$\omega_i + \frac{K}{N} \sum_{j=1}^{N} \sin(\theta_i - \theta_j) = 0, \text{ for } i = 1, \ldots, N$$

$$\sin\theta_i := s_i \qquad \cos\theta_i := c_i$$

Complex Systems: Synchronization

The Kuramoto Model:

$$\omega_i + \frac{K}{N}\sum_{j=1}^{N}\sin(\theta_i - \theta_j) = 0,\ \text{for } i = 1,\ldots,N$$

$$\sin\theta_i := s_i \qquad \cos\theta_i := c_i$$

$$\omega_i + \frac{K}{N}\sum_{j=1}^{N}(c_i s_j - s_i c_j) = 0,\ \text{for } i = 1,\ldots,N$$

Complex Systems: Synchronization

The Kuramoto Model:

$$\omega_i + \frac{K}{N} \sum_{j=1}^{N} \sin(\theta_i - \theta_j) = 0, \text{ for } i = 1, \ldots, N$$

$$\sin \theta_i := s_i \qquad \cos \theta_i := c_i$$

$$\omega_i + \frac{K}{N} \sum_{j=1}^{N} (c_i s_j - s_i c_j) = 0, \text{ for } i = 1, \ldots, N$$

$$s_i^2 + c_i^2 - 1 = 0, \text{ for } i = 1, \ldots, N$$

Complex Systems: Synchronization

The Kuramoto Model:

$$\omega_i + \frac{K}{N}\sum_{j=1}^{N}\sin(\theta_i - \theta_j) = 0, \text{ for } i = 1, \ldots, N$$

$$\sin\theta_i := s_i \qquad \cos\theta_i := c_i$$

$$\omega_i + \frac{K}{N}\sum_{j=1}^{N}(c_i s_j - s_i c_j) = 0, \text{ for } i = 1, \ldots, N$$

$$s_i^2 + c_i^2 - 1 = 0, \text{ for } i = 1, \ldots, N$$

Polynomial equations → Algebraic Geometry → Computations

An Example

$$x\,z - 3\,y + 1 = 0$$

$$x^2 - 2\,y = 0$$

$$x\,y - 5 = 0$$

$$\text{Solve for } x, y, z.$$

Two methods:

1.Groebner Basis Method

2.Numerical Algebraic Geometry

Groebner Basis

- **Very roughly speaking, one can obtain another system of polynomial equations by performing a finite set of operations on the original system (the Buchberger algorithm with lexicographic monomial ordering)**

- **The new system is 'easier' to solve**

- **The new system has the same solutions as the original**

- **The new system is called the Groebner basis**

- **Packages like Singular, COCOA, MACAULAY2, MAGMA, Maple, Mathematica, etc.**

- **The first three are available for free !!**

How is it useful?

For the running example, Mathematica gives (lexicographic monomial ordering)

$$x^3 - 10 = 0$$

$$-x^2 + 2\,y = 0$$

$$x^2 - 15\,x + 10\,z = 0$$

How is it useful?

For the running example, Mathematica gives (lexicographic monomial ordering)

$$x^3 - 10 = 0$$

$$-x^2 + 2\,y = 0$$

$$x^2 - 15\,x + 10\,z = 0$$

Univariate equation. 3 solutions.

How is it useful?

For the running example, Mathematica gives (lexicographic monomial ordering)

$$x^3 - 10 = 0$$

$$-x^2 + 2\,y = 0$$

$$x^2 - 15\,x + 10\,z = 0$$

Univariate equation. 3 solutions.

Back-substitute the three solutions in the rest of the system

How is it useful?

For the running example, Mathematica gives (lexicographic monomial ordering)

$$x^3 - 10 = 0$$

$$-x^2 + 2\,y = 0$$

$$x^2 - 15\,x + 10\,z = 0$$

Univariate equation. 3 solutions.

Back-substitute the three solutions in the rest of the system

There are 3 solutions: 1 real + 2 complex

Numerical Algebraic Geometry/ Homotopy Continuation Method

1. **Estimate an upper bound of the number of solutions of the system to be solved.**

   e.g.,

   Bezout bound = product of degrees of all the polynomials in the system.

   = 2x2x2 = 8, for our running example

$$\vec{f}(x,y,z) = (x\,z - 3\,y + 1, x^2 - 2\,y, x\,y - 5)^T$$

Numerical Algebraic Geometry/ Homotopy Continuation Method

1. **Estimate an upper bound of the number of solutions of the system to be solved.**

   e.g.,

   Bezout bound = product of degrees of all the polynomials in the system.

   = 2x2x2 = 8, for our running example

$$\vec{f}(x,y,z) = (x\,z - 3\,y + 1, x^2 - 2\,y, x\,y - 5)^T$$

2. **Construct a new system in the same variables**

   (a) which has the same no. of solutions as the estimated upper bound,

   (b) easy to solve

   e.g.,

$$\vec{g}(x,y,z) = (x^2 - 1, y^2 - 1, z^2 - 1)^T$$

Numerical Algebraic Geometry/ Homotopy Continuation Method

1. **Estimate an upper bound of the number of solutions of the system to be solved.**

   e.g.,

   Bezout bound = product of degrees of all the polynomials in the system.

   = 2x2x2 = 8, for our running example

$$\vec{f}(x,y,z) = (x\,z - 3\,y + 1, x^2 - 2\,y, x\,y - 5)^T$$

2. **Construct a new system in the same variables**

   (a) which has the same no. of solutions as the estimated upper bound,

   (b) easy to solve

   e.g.,

$$\vec{g}(x,y,z) = (x^2 - 1, y^2 - 1, z^2 - 1)^T$$

3. **Track each solution of the new system using**

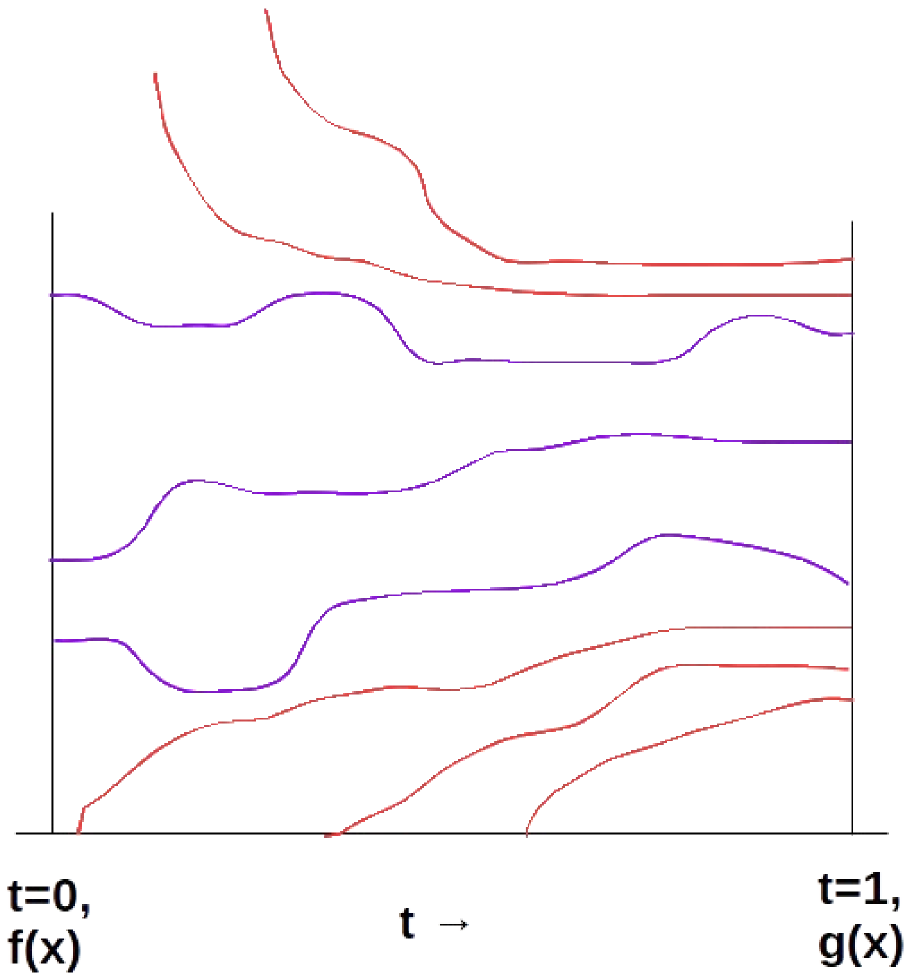
$$\vec{H}((x,y,z),t) = (1-t)\vec{f}(x,y,z) + e^{i\gamma}t\vec{g}(x,y,z) = 0$$

   from t=1 to t=0, using predictor-corrector or any other method.

   If a solution of the new system converges to the original one at t=0, then it is a solution, otherwise not.

   Note that 'gamma' is a generic real number, and is important here.

Numerical Algebraic Geometry/ Homotopy Continuation Method



Numerical Algebraic Geometry/ Homotopy Continuation Method

There are well-written packages available for free:

Bertini, HOM4PS2, PHCPack.

Groebner Basis	Numerical Algebraic Geometry
1. Exact solutions	Numerical, but ALL solutions/extrema
2. Exponential space complexity	No such scaling problems
3. Highly sequential	‘Embarrassingly’ parallelizable
4. Non-integer coefficients a problem	Floating point coefficients are fine

Groebner Basis	Numerical Algebraic Geometry
1. Exact solutions	Numerical, but ALL solutions/extrema
2. Exponential space complexity	No such scaling problems
3. Highly sequential	‘Embarrassingly’ parallelizable
4. Non-integer coefficients a problem	Floating point coefficients are fine

NAG: Framework rather than a method

Groebner Basis	Numerical Algebraic Geometry
1. Exact solutions	Numerical, but ALL solutions/extrema
2. Exponential space complexity	No such scaling problems
3. Highly sequential	‘Embarrassingly’ parallelizable
4. Non-integer coefficients a problem	Floating point coefficients are fine

NAG: Framework rather than a method

Caution: The Groebner basis methods can work exceptionally well in many cases (e.g., Sudoku) ...

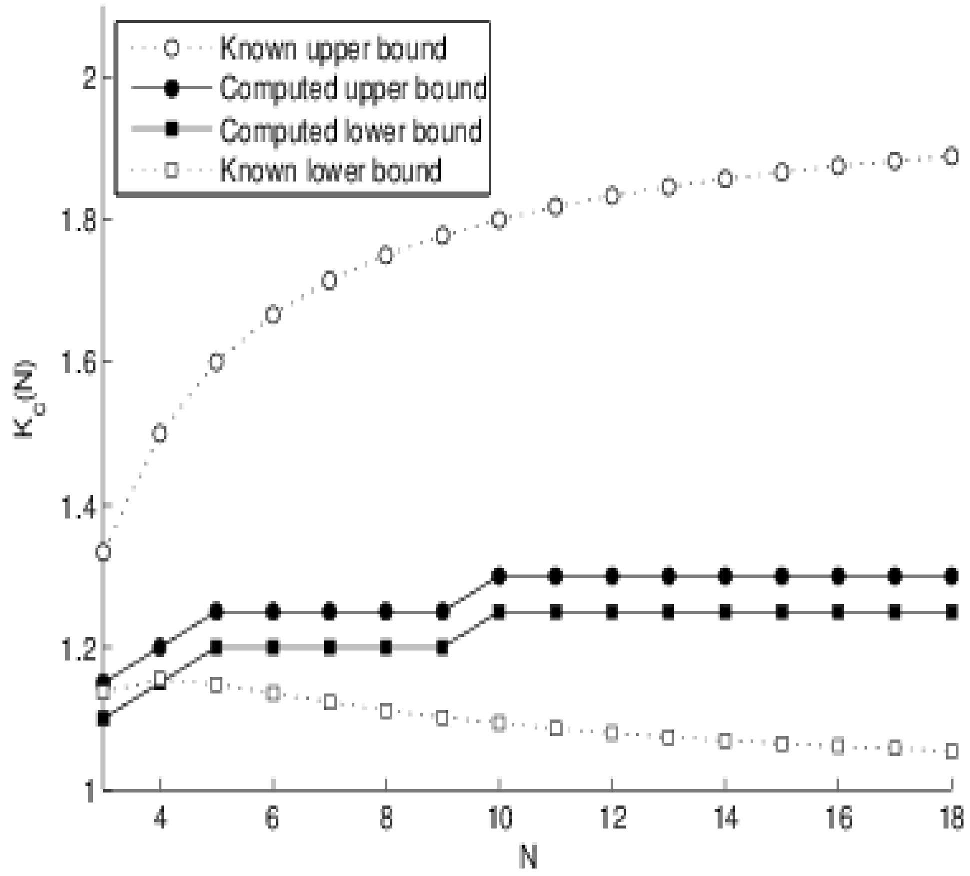
Complex Systems: Synchronization

Solve (DM, Noah Daleo, Jonathan D Hauenstein, Florian Doerfler):

$$\omega_i + \frac{K}{N}\sum_{j=1}^{N}(c_i s_j - s_i c_j) = 0, \text{ for } i = 1, \ldots, N$$

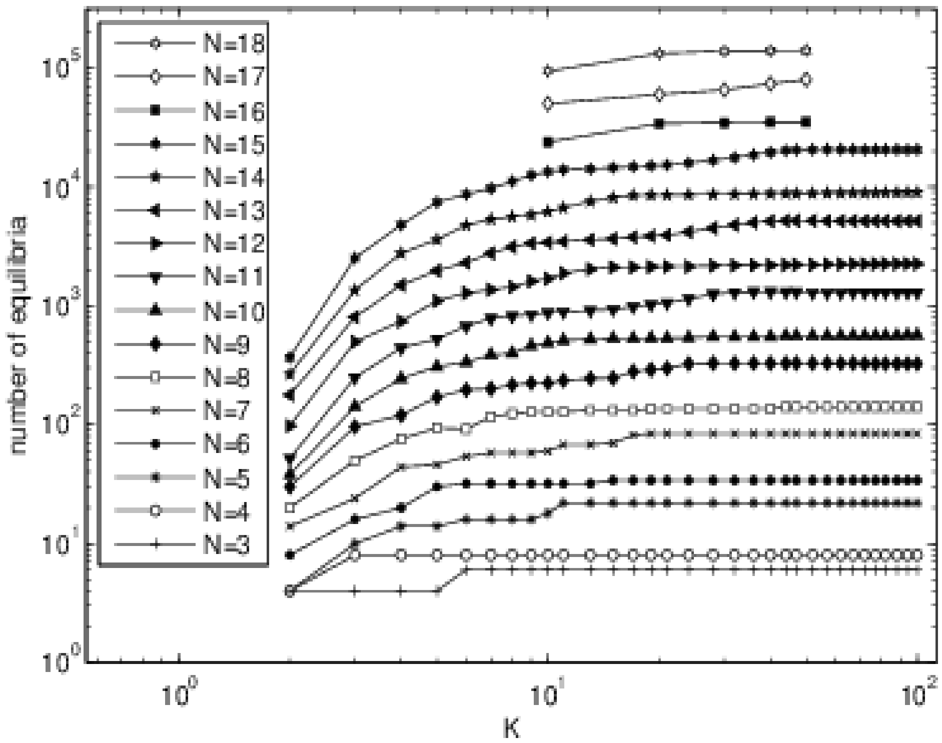
$$s_i^2 + c_i^2 - 1 = 0, \text{ for } i = 1, \ldots, N$$

Complex Systems: Synchronization (For equidistant frequencies)



DM, Noah Daleo, Jonathan D Hauenstein, Florian Doerfler. 2015.

Complex Systems: Synchronization (For equidistant frequencies)



DM, Noah Daleo, Jonathan D Hauenstein, Florian Doerfler. 2015.

Complex Systems: Synchronization (For equidistant frequencies)

Conjecture [Araposthatis et al., 1981]: if there is a stable equilibrium, then there is a stable equilibrium with

$$|\theta_i - \theta_j| < \frac{\pi}{2}, \text{ for all neighbours } i, j.$$

Complex Systems: Synchronization (For equidistant frequencies)

Conjecture [Araposthatis et al., 1981]: if there is a stable equilibrium, then there is a stable equilibrium with

$$|\theta_i - \theta_j| < \frac{\pi}{2}, \text{ for all neighbours } i, j.$$

Disproved by counter-example.

Complex Systems: Synchronization
(For equidistant frequencies)

Conjecture [Araposthatis et al., 1981]: if there is a stable equilibrium, then there is a stable equilibrium with

$$|\theta_i - \theta_j| < \frac{\pi}{2}, \text{ for all neighbours } i, j.$$

Disproved by counter-example.

- **Other results for the complete graph**
- **Results for other graphs (cyclic, Erdos-Renyi random graph, etc.)**
- **In future, other graphs, eigenvalue analysis, stochastic versions of the Kuramoto model**

Power Flow Problem

- In addition to the Kuramoto model

- Power flow equations: flow of power in an interconnected system

$$0 = -P_i + \sum_{k=1}^{n} G_{ik}(V_{iRe}V_{kRe} + V_{iIm}V_{kIm}) + \sum_{k=1}^{n} B_{ik}(V_{kRe}V_{iIm} - V_{iRe}V_{kIm});$$

$$0 = \quad Q_i + \sum_{k=1}^{n} G_{ik}(V_{kRe}V_{iIm} \quad V_{iRe}V_{kIm}) \quad \sum_{k=1}^{n} B_{ik}(V_{iRe}V_{kRe} + V_{iIm}V_{kIm})$$

- B (real part of the bus admittance matrix), G (imaginary part of the bus admittance matrix), P (net power injected) and Q (net reactive power injected) are parameters.

- Solutions are important for determining the best operation of the system as well as planning future expansions on the system, etc.

Power Flow Problem

- Found all the steady states for up to ***n <=14*** bus system (IEEE test systems) DM, K Turitsyn and H Nguyen, 2014. The first ever complete database of equilibria.

- Multistability in wind energy systems. S Chandra, DM, A Chakrabortty. 2014, 2015, 2016.

- Network topology dependent upper bound on the number of power flow equilibria. DM, T Chen, D Molzahn, M Niemerg. 2015, 2016.

Current and Future Works

- Further work on *graph topology dependent* upper bounds on both complex and real power flow and Kuramoto equilibria

- Novel computational algebraic geometry methods such as discriminant variety to identify all the solution-boundaries (where the Jacobian is singular)

- Novel (non-polynomial) and tailor-made homotopy continuation methods to find stable and type-1 solutions

- Dynamical systems on graphs

- Optimal Power Flow Problem with polynomial homotopy continuation

- Machine learning (artificial neural networks and deep learning)

- Belief Propagation (probabilistic graphical models), etc.

- Computer vision.