University of Cagliari
Faculty of Engineering
Department of Electronic Engineering

Algorithms for the Stabilization of a Leader-Follower formation described via complex Laplacian

Candidate: Fabrizio Serpi

Advisors: Prof. Alessandro Giua
Prof. Zhiyun Lin

October 24, 2012

Table of Content

Part I

Introduction

Introduction

- Agents formation is represented by graphs.
- **Complex-valued Laplacian-based formation control**.
    - Weighted directed graphs.
    - Complex - valued Laplacian L.
    - Leader-follower formation with 2 co-leaders.
    - Single - integrator kinematics.
    - Double - integrator dynamics.

Introduction

- Agents can reach a planar formation if matrix $-L$ is stable.
- If matrix $-L$ is unstable then exist a complex diagonal matrix D such that $-DL$ is stable.
- Consider a group of m agents.
  - 2 **co-leaders** labelled 1 and 2.
  - $m-2$ **followers** labelled from 3 to m.
- Leader nodes (1 and 2) do not have incoming edges. The Laplacian matrix takes the following form

$$L = \left[\begin{array}{c|c} 0_{2\times 2} & 0_{2\times(m-2)} \\ \hline L_{lf} & L_{ff} \end{array}\right].$$

Part II

Algorithms and Simulations

Preliminaries

- (Ballantine , Friedland) Matrix A to be stabilized has to have all non-zero leading principal minors.
- Given a **group of agents**, find:
  1. a permutation matrix P such that $\hat{L}_{ff} = PL_{ff}P^T$ has all non-zero principal minors,
  2. a complex diagonal matrix M such that the matrix $-DL$ is stable, where D is:
     - **Single-Integrator Kinematics**

$$D = \begin{bmatrix} I_2 & 0 \\ 0 & M \end{bmatrix}.$$

     - **Double-Integrator Dynamics**

$$D = \begin{bmatrix} I_2 & 0 \\ 0 & \epsilon M \end{bmatrix}.$$

Backtracking Design Technique

- The solution must be expressible as an n-tuple $(x_1, \ldots, x_n)$ where $x_i \in S_i$.
  - S_i is a finite set.
- The problem calls for finding vectors satisfying a criterion function $\mathcal{P}(x_1, \ldots, x_n)$.
- The basic idea is to build up the same vector one component at a time.
- $T(x_1, x_2, \ldots, x_i)$ generates values for x_{i+1} such that $(x_1, x_2, \ldots, x_{i+1})$ is also a partial solution.
- Use **bounding functions** $\mathcal{B}_i$ to test whether the vector has any chance of success.

Backtracking Efficiency

- Algorithm has been implemented in a recursive way.
- Worst case time complexity for a backtracking algorithm.

1. number of possible solutions generated $\rightarrow \mathbf{2^n}$

$$O(q(\mathbf{n})\mathbf{2^n}).$$

2. number possible solutions generated $\rightarrow \mathbf{n}!$

$$O(p(\mathbf{n})\mathbf{n}!).$$

Modelling the Automorphism Problem

- The problem of finding P which solves equation $\hat{L}_{ff} = P L_{ff} P^T$, can be solved by a backtracking-based algorithm.
- Rewrite the automorphism equation:

$$\hat{L}_{ff} = \begin{bmatrix} e_i^T \\ e_j^T \\ e_h^T \\ \vdots \\ e_s^T \end{bmatrix} L_{ff} \begin{bmatrix} e_i & e_j & e_h & \cdots & e_s \end{bmatrix}.$$

Modelling the Automorphism Problem

- Find the right sequence of the vectors e_i-th such that, the permutation matrix P obtained, allows constraints over $\hat{L}_{ff}$ to hold.
- Sets S_i are sets of real numbers chosen through 1 to n, where n is the order of P:

$$S_i = \{1, \ldots, n\}, \;\; 1 \leq i \leq \text{max-steps}.$$

- For the current problem, the *criterion function* $\mathcal{P}(x_1, \ldots, x_n)$ is to obtain a matrix $\hat{L}_{ff}$ with all non-zero leading principal minors.

Modelling the Automorphism Problem

- Determinant - based algorithm.
  - Determinants of the leading principal minors are explicitly computed.
- Gauss - based algorithm.
  - Principal minors are tested via the LU factorization of the entire matrix.
- Algorithms have been design to search for the first occurrence of a solution.
- Different ways to generate the partial solutions have been used.

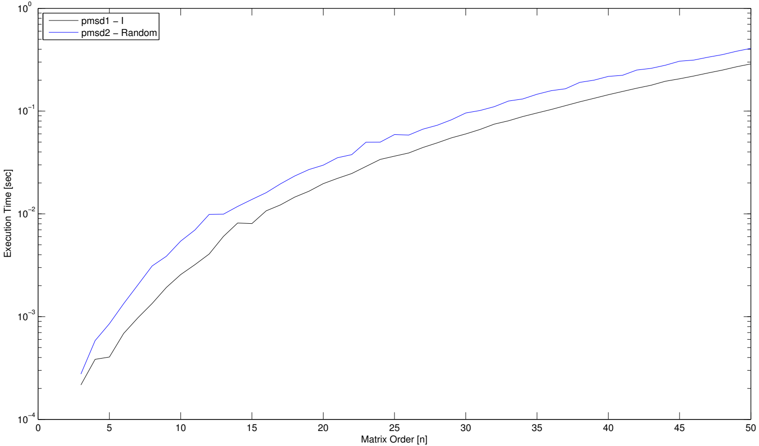
Determinant-based Algorithm

- The overall complexity of the algorithm is

$$O(p(n)n!) = O\left(\frac{1}{6}n^4 n!\right).$$

- Two different T function have been used:
    - T_{d1} supplies values x_i in order to search for the nearest solution to the identity matrix.
    - T_{d2} supplies values x_i randomly picking them from the set S_i until all values are tried.

Determinant-based Algorithm



Gauss-based Algorithm

- Automorphism equation.

$$\hat{L}_{ff} = PL_{ff}P^T.$$

- The Gauss elimination method with total pivoting and the correspondent *LU* factorization are

$$LU = PAQ.$$

- The automorphism problem can be seen as a *LU* factorization with total pivoting, where $Q = P^T$.

$$\hat{L}_{ff} = \hat{L}_g\hat{U}_g = PL_{ff}P^T.$$

Gauss-based Algorithm

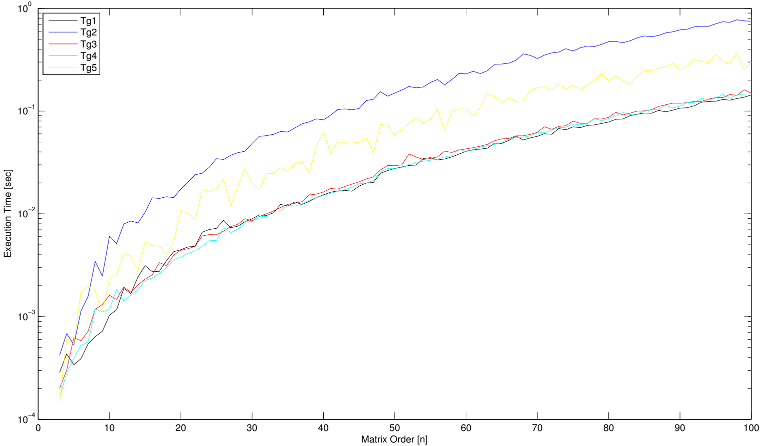
- The equivalence of the two problems is ensured by the existence theorem for the *LU* factorization.
  - *LU* factorization exists for a matrix if all its leading principal minors are non-null.
- The highest number of tuples that can be generated is $n!$.
- The overall worst case complexity for the algorithm results to be

$$\mathrm{O}\left(\mathrm{p}(n)n!\right) = \mathrm{O}\left(\frac{2}{3}n^3 n!\right).$$

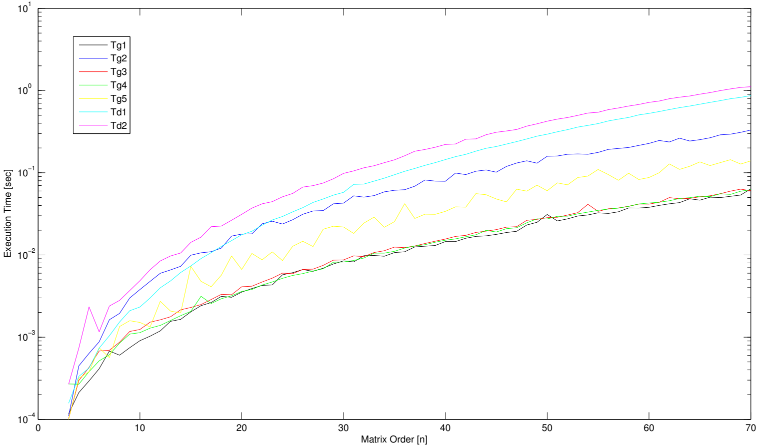
Gauss-based Algorithm

- From a Gauss elimination point of view, we chose four different Ts. At call k,
  - T_{g1} searches for the next non-null pivot element along the diagonal from position k to n;
  - T_{g2} searches randomly for a non-null pivot element along the sub-diagonal from position k to n;
  - T_{g3} searches for the pivot element with maximum modulus along the diagonal, from position k to n.
  - T_{g4} searches for the next non-null element along the diagonal from position n to k.
  - function T_{g5} behaves like one of the functions among T_{g1}, T_{g2} and T_{g3}. The choice is made randomly.

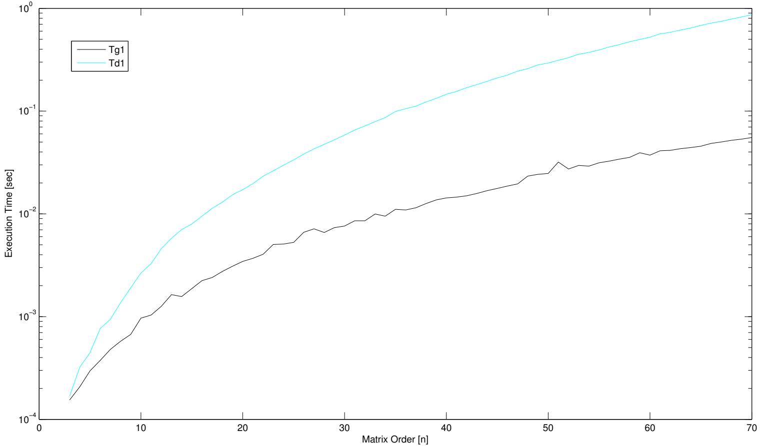
Gauss-based Algorithm



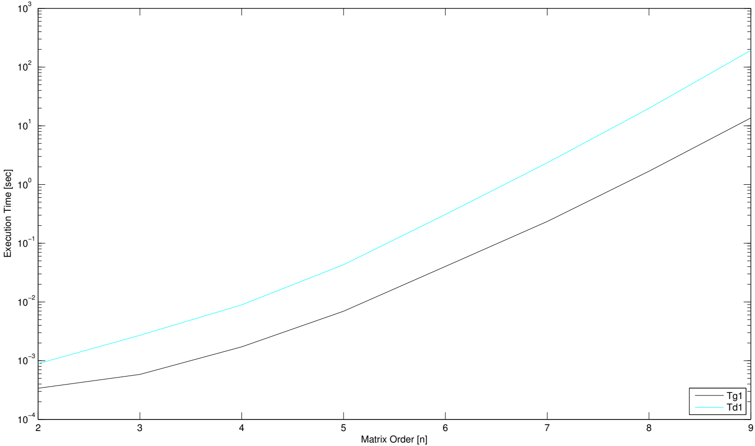
Comparing Algorithms - complex random L_{ff}



Comparing Algorithms - complex random L_{ff}



Comparing Algorithms - complex random singular matrices



Multiplicative IEP

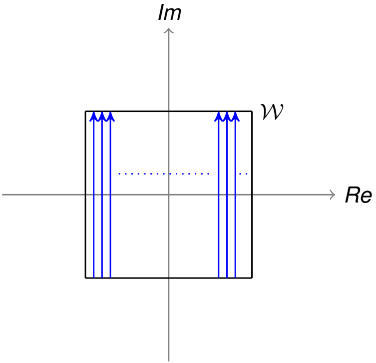
- The MAS is able to reach a planar formation if and only if the matrix $-\mathbf{DL}$ has all stable eigenvalues.
- The problem of finding D is known as Multiplicative Inverse Eigenvalue Problem.
- **Friedland (1975)** proved the solvability for the **MIEP**.
- **Ballantine (1970)** proved the existence of a complex diagonal matrix M such that complex matrix MA has eigenvalues with positive real parts.

Ballantine's Theorem

- The diagonal matrix can be computed one element at a time.
- Each diagonal element is chosen in order to modify the eigenvalues of a leading principal minor.
- the process is iterative, from the smallest to the biggest leading principal minor.
- (Complex case) d_i belongs to a sector which contains the positive real axis.
- Depending on where and how the diagonal elements are searched for, the algorithm supplies different solutions for the same problem.

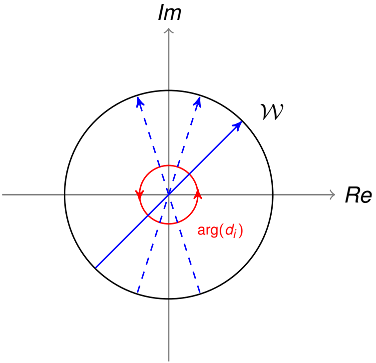
Ballantine's Theorem

- $\mathcal{W} = \{d_i : d_{r,i} \in [d_{r,min}, d_{r,max}], d_{m,i} \in [d_{m,min}, d_{m,max}]\}$



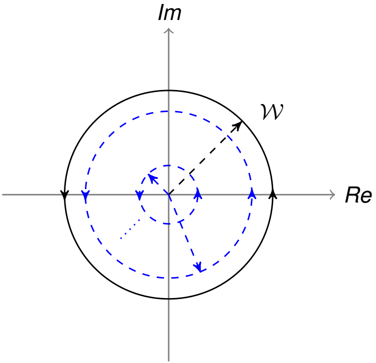
Ballantine's Theorem

- $d_i = d_{r,i} + \iota d_{m,i} = |d_i| \, e^{\iota \arg(d_i)}$



Ballantine's Theorem

- $d_i = d_{r,i} + \iota d_{m,i} = |d_i|\, e^{\iota\, \arg(d_i)}$



Part III

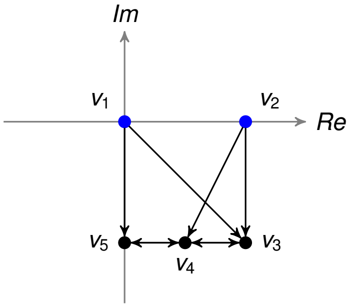
Application to a MAS formation

Simulations

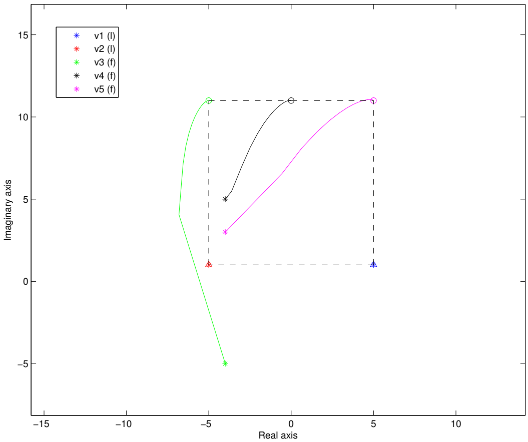
- Simulations for the **single-integrator kinematics** and the **double-integrator dynamics**.

  - Formation basis

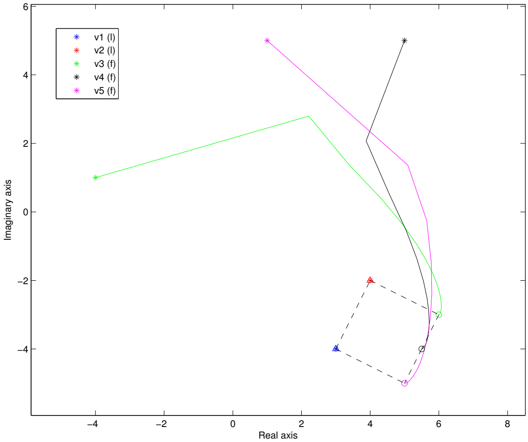
$$\xi = \begin{bmatrix} 0 \\ 4 \\ 4-4\iota \\ 2-4\iota \\ -4\iota \end{bmatrix}$$



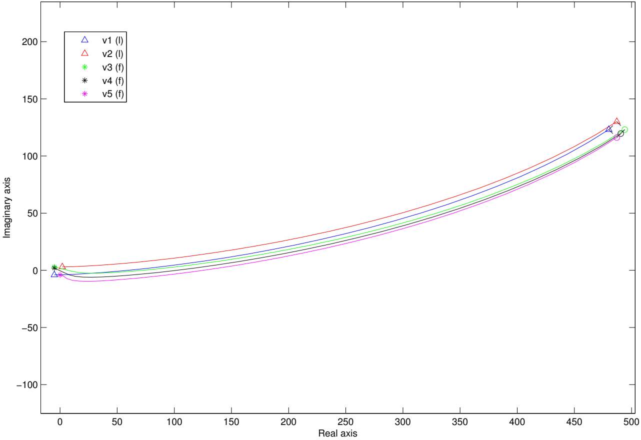
Simulations - SIK - unforced case



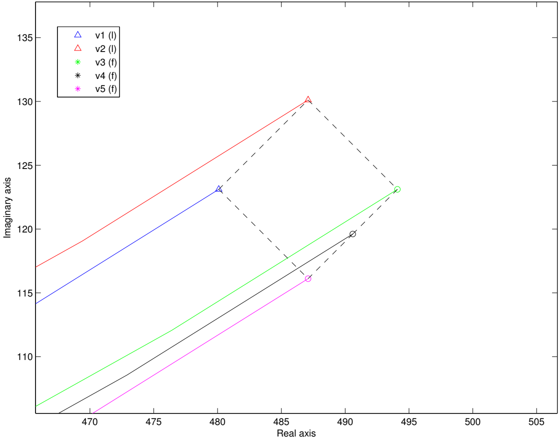
Simulations - SIK - unforced case



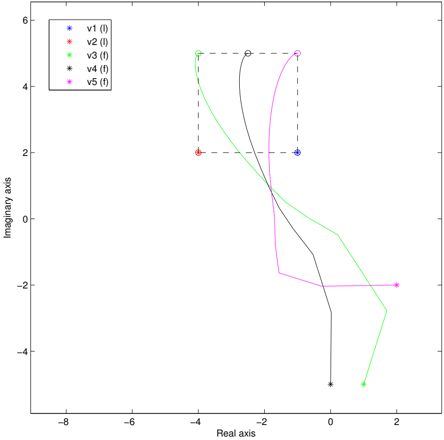
Simulations - SIK - forced case



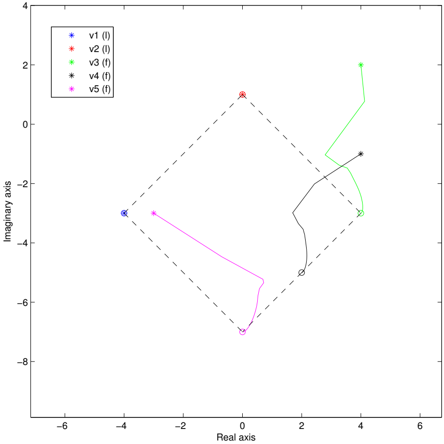
Simulations - SIK - forced case



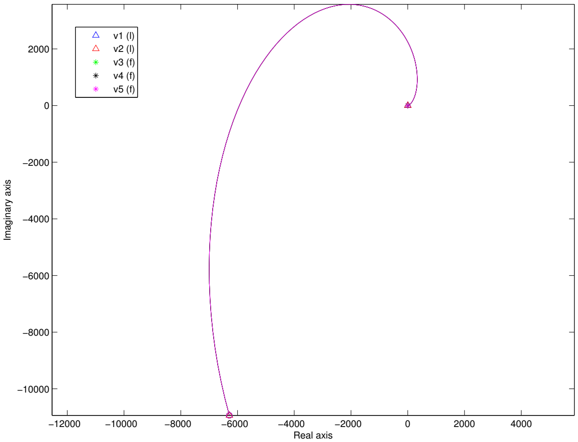
Simulations - DID - unforced case



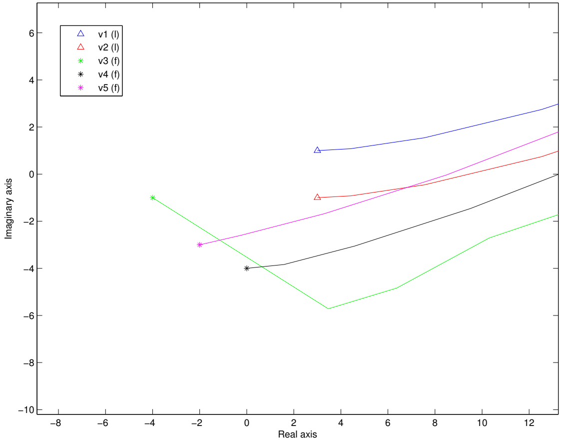
Simulations - DID - unforced case



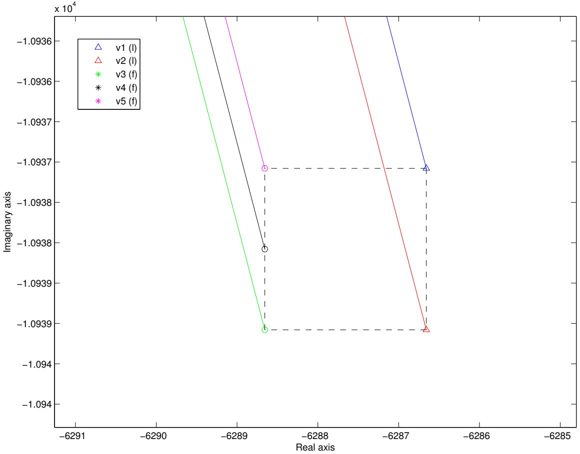
Simulations - DID - forced case



Simulations - DID - forced case



Simulations - DID - forced case



Conclusion

- The algorithms discussed have their limits.
  - High complexity for the worst case of the permutation matrix problem.
  - The impossibility of choosing directly the eigenvalues in the MIEP.
- Still they can be used for further research.
- Practical applications of interest could be
  - collision avoidance,
  - limited sensing capability.

THANK YOU!