Computing with distance geometry

Leo Liberti, *Maël Kupperschmitt*
LIX CNRS Ecole Polytechnique, Institut Polytechnique de Paris, France
`liberti@lix.polytechnique.fr`

ICERM Workshop on Rigidity, 260706-10

Introduction

Summary

- Humans always model problems as SAT, Mathematical Programming (MP), constraint programming, logic programming, and so on
- All of these languages have explicit variable symbols
- Advanced SAT/MP/... solvers work very well, but they require sizable investments in terms of development
- Every **NP**-hard problem could serve as "modelling language"
- Consider using the Distance Geometry Problem (DGP)
  a mixed-discrete **NP**-hard problem
- Advantages:
  - raise dimensions $\Rightarrow$ relaxation
  - can use local solvers in continuous space to produce good solutions of combinatorial problems

Background: problems, instances, and the class NP

- *Problem*: formal description of input and output

  e.g. MAX STABLE:

  - input: graph $G = (V, E)$, $n = |V|$, $m = |E|$
  - output: largest $U \subseteq V$ s.t. $\forall u, v \in U \; \{u, v\} \notin E(G[U])$
  - P is a *decision* problem if output is in $\{\text{YES}, \text{NO}\}$

    e.g. k-STABLE: YES if $|U| \geq k$, NO othw

- *Instance* of a problem P: input data

  e.g. the graph $(V = \{1, 2, 3\}, E = \{\{1, 3\}, \{2, 3\}\})$

- A problem P corresponds to the subset of all of its instances

  we only consider problems with infinitely many instances

- We also consider *problem classes*

  e.g. **NP**: decision problems s.t.

  every YES instance comes with a polytime-checkable certificate

Background: reduction, hardness, and modeling

- *Reduction*: a polytime-computable map $\rho : \text{NP} \to \text{NP}$
  $(P, Q) \in \rho$ iff $\forall p \in P, q \in Q$ we have $\rho : p \mapsto q$ s.t. p is YES iff q is YES
- If $\exists$ reduction $\rho : P \to Q$, P cannot be harder to solve than Q
  othw for all $p \in P$ solve $\rho(p) \in Q$ instead $\Rightarrow$ P as hard as Q
- By [Cook 1971], SAT is **NP**-hard, i.e. hardest in **NP**
  if $\exists$ reduction $\rho : \text{SAT} \to Q$, Q is **NP**-hard [Karp 1972]
- A reduction $\rho : P \to Q$ transforms input of P to that of Q
  consider parsers $\mathscr{P}$ for P, $\mathscr{Q}$ for Q: ρ describes P using language of Q
  $\Rightarrow$ *ρ models P as Q*
- *Examples*:
  - [Cook 1971] models a certificate verification TM as a SAT
  - [Karp 1972] models SAT in terms of 3SAT
  - reduction ρ s.t. $\bigwedge_i (\bigvee_j \ell_j) \mapsto \forall i \, (\sum_j \ell_j), \ell_j \equiv x_j \mapsto x_j, \ \ell_j \equiv \neg x_j \mapsto 1 - x_j$
    models SAT as BINARY LINEAR PROGRAMMING (BLP)
  - $\forall \{u, v\} \in E \; x_u + x_v \leq 1 \wedge \sum_{v \in V} x_v \geq k$ models k-STABLE as BLP

Reduction and modelling are equivalent on **NP**

The human modeling bias

- By [Cook 1971], [Karp 1972], reductions between **NP**-hard problems form a connected graph
- But humans always model with a handful of target problems: SAT, BLP, MATHEMATICAL PROGRAMMING (MP), constraint programming (CP), rule-based programming, ProLog, LISP...
- What these problems/languages have in common: *unknown output/certificate is explicitly represented by variables*

  imagine modelling all problems with k-STABLE instead!
- This shows that we have a human modeling bias

  does this bias incur a computational cost?

  in other words, would we be better off with universal solvers for k-STABLE rather than for BLP, MP, CP?
- The computational cost might also concern the amount of work for creating and maintaining Gurobi, CPLEX, FICO Xpress, Hexaly, Mosek...

We consider the Distance Geometry Problem (DGP) as a universal problem for **NP**

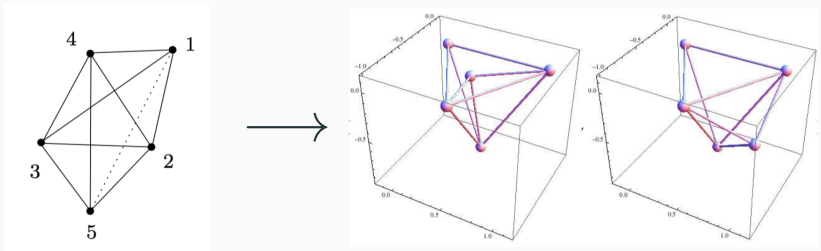
Distance Geometry Problem

Introduction to the Distance Geometry Problem

- The **Distance Geometry Problem** (DGP):
  find point positions in $\mathbb{R}^K$ from distances
- input: simple graph $G = (V, E, d)$ and integer $K > 0$
- output: realization (rlz) matrix $x \in \mathbb{R}^{nK}$ so that

$$\forall \{u, v\} \in E \; \|x_u - x_v\|_2^2 = d_{uv}^2 \quad \text{(DGP)}$$

  system of quadratic equations in nK variables
- Norms and distances may vary; Euclidean norm most common
- Equivalent definitions:
  draw graph in $\mathbb{R}^K$ s.t. edges represented by segments of length equal to weight
  inverse problem to "given n points in $\mathbb{R}^K$ find some of the inter-point distances"



- The DGP is **mixed-discrete**:
  - based on a (discrete) graph, with decisions in (continuous) $\mathbb{R}^K$
  - as edge density increases, flexible (continuous) rlz become rigid (discrete)

Some native applications of the DGP

- clock synchronization ($K = 1$)
- sensor network localization ($K = 2$)
- molecular structure from distance data ($K = 3$)
- autonomous underwater vehicles ($K = 3$)
- EDM completion (whatever K)
- finding graph embeddings (whatever K)
- *The DGP with fixed dimension K is denoted* DGP_K

Complexity of the DGP

- The DGP_1 with input in $\mathbb{Z}$ is in **NP**

  Proof:

  - if instance YES $\exists$ realization $x \in \mathbb{R}^{n\times 1}$
  - if some component $x_i \notin \mathbb{Q}$ translate x so $x_i \in \mathbb{Q}$
  - consider any other $x_j \notin \mathbb{Q}$
  - $\exists s_{uv} \in \{0,1\}$ s.t. $\ell = |\text{sh. path } p : i \to j| = \sum_{\{u,v\}\in p} (-1)^{s_{uv}} d_{uv} \in \mathbb{Q}$
  - then $x_j = x_i \pm \ell \to x_j \in \mathbb{Q}$
  - $\Rightarrow \exists$ polytime verification algorithm of $\forall \{i,j\} \in E \quad |x_i - x_j| = d_{ij}$

- DGP_K may not be in **NP** for $K > 1$

  don't know how to polytime check $\|x_i - x_j\|_2 = d_{ij}$ for $x \notin \mathbb{Q}^{nK}$ [Beeker &al. 2013]

- The DGP_1 is *weakly* **NP**-hard by reduction from PARTITION

  does not prevent pseudo-polynomial algorithms

- The DGP_1 is *strongly* **NP**-hard by reduction from 3SAT

  prevents pseudo-polynomial algorithms

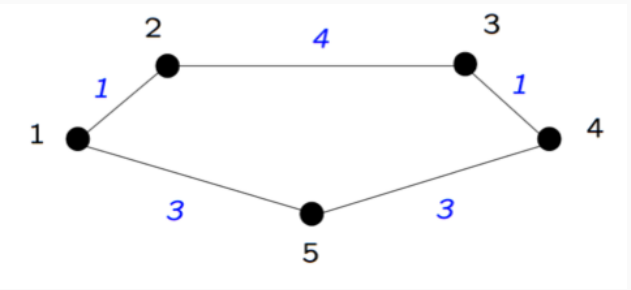
[Saxe 1979]

Weak NP-hardness

PARTITION *is (weakly)* NP*-hard*

Given $a = (a_1, \ldots, a_n) \in \mathbb{N}^n$, $\exists\ I \subseteq \{1, \ldots, n\}$ s.t. $\sum_{i \in I} a_i = \sum_{i \notin I} a_i$?

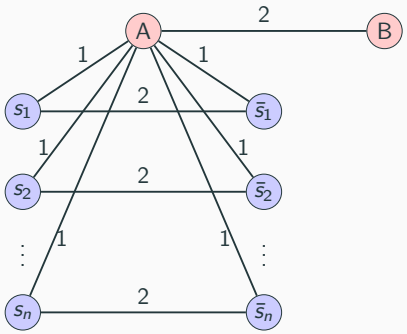
- Reduce PARTITION $\rightarrow$ DGP_1 on single-cycle graphs [Saxe 1979]
- $a \longrightarrow$ cycle C, with $V(C) = \{1, \ldots, n\}$, $E(C) = \{\{1, 2\}, \ldots, \{n, 1\}\}$
- For $i < n$ let $d_{i,i+1} = a_i$
  For $i = n$, let $d_{n,n+1} = d_{n1} = a_n$
- *E.g. for* $a = (1, 4, 1, 3, 3)$*, get cycle graph as* DGP_1 *instance*:



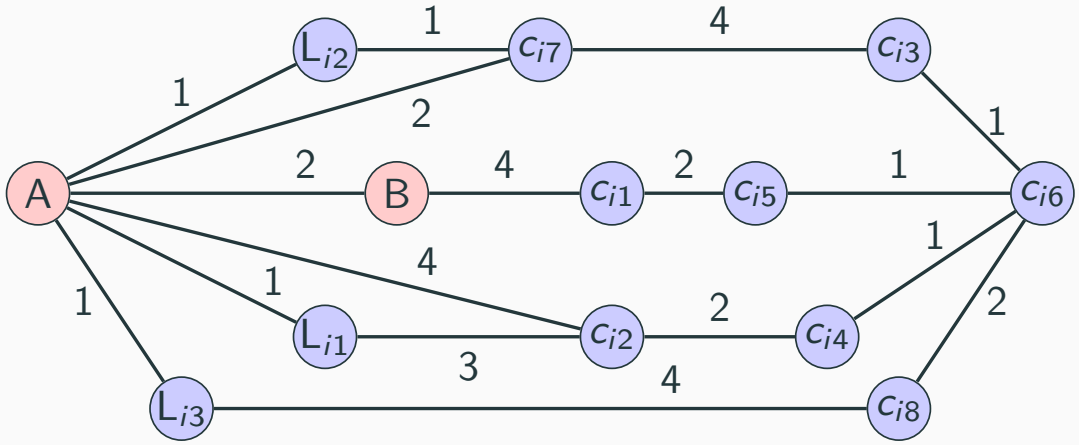
- Relization: i-th vertex position at $x_i = x_{i-1} \pm d_{i-1,i}$
  $+d_{i-1,i}$: right of x_{i-1} $\quad$ $-d_{i-1,i}$: left of x_{i-1}
- **Lemma:** $d_{n+1} = d_1$ **iff** DGP_1 **instance is YES**
- *This reduction is a bijection* PARTITION $\longleftrightarrow$ DGP_1 on simple cycles

Strong NP-hardness

- Reduce $3\text{SAT} \rightarrow \text{DGP}_1$ using two types of gadget graphs



for literals $s_j, \bar{s}_j$



for clauses $\text{L}_{i1} \vee \text{L}_{i2} \vee \text{L}_{i3}$

joined by the A, B anchors

- Proof that 3SAT is YES iff DGP_1 is YES

  by enumeration of path lengths over left/right vertex positions

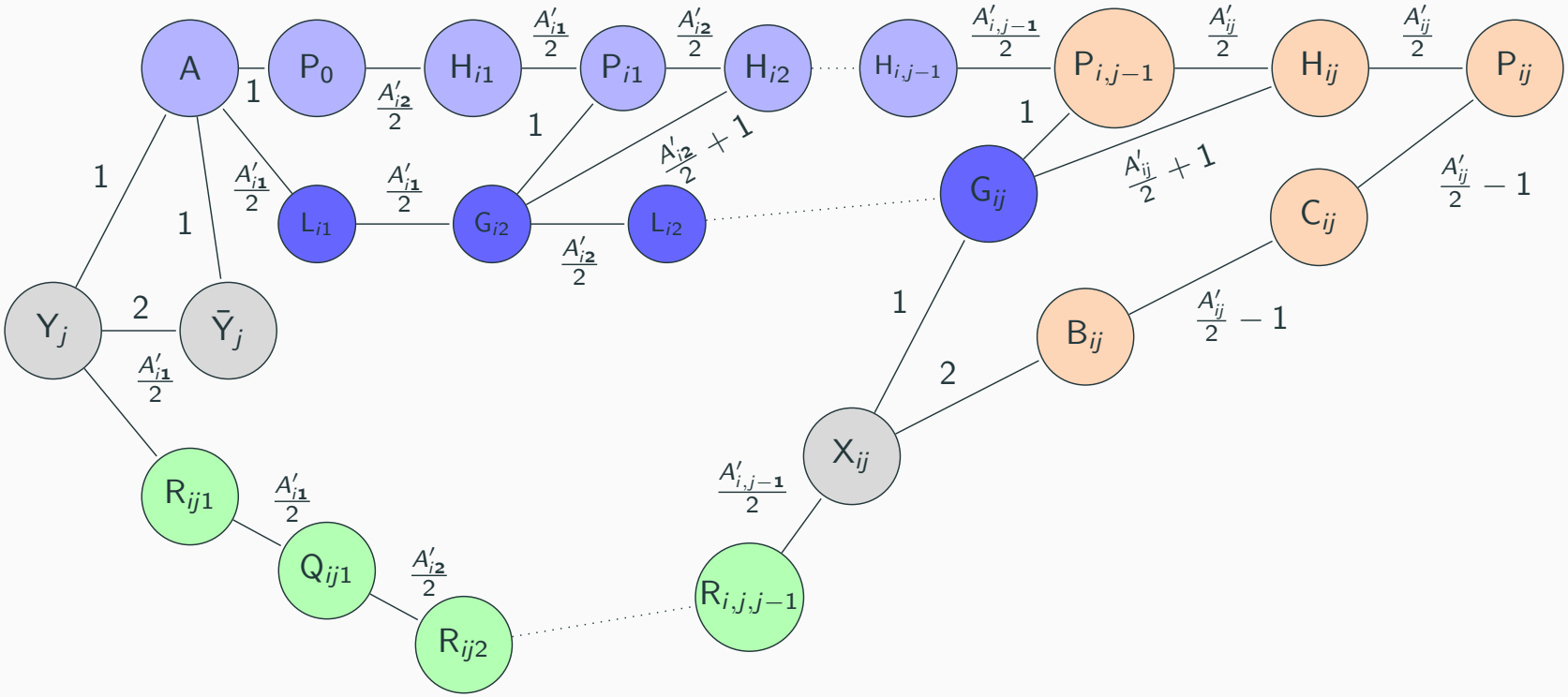
[Saxe 1979]

Modelling with the DGP

Human modeling sneaks back in

- Human bias may be costly
  but modelling without variables is painful
- Can use reductions to automate modelling without variables
- **E.g.** SAT has variables s
  use reductions $\rho_1 : \text{SAT} \to 3\text{SAT}$, $\rho_2 : 3\text{SAT} \to \text{DGP}_1$
  $\forall S \in \text{SAT} \quad (\rho_2 \circ \rho_1)(S) \in \text{DGP}_1$
- **E.g.** SUBSET-SUM is the BLP constraint $\sum_{j \leq n} a_j y_j = b \wedge y \in \{0,1\}^n$
  use reductions $\rho_3 : \text{SUBSET-SUM} \to \text{PARTITION}$, $\rho_4 : \text{PARTITION} \to \text{DGP}_1$
  $\forall \sigma \in \text{SUBSET-SUM} \quad (\rho_4 \circ \rho_3)(\sigma) \in \text{DGP}_1$
- To model in BLP $Ay = b \wedge y \in \{0,1\}$, we need $\rho : \text{BLP} \to \text{DGP}_1$
  1. Rewrite $-A_{ij}y_j$ as $+A_{ij}\bar{y}_j$ [Abío & al. JMLR 2012]
  2. Transform additions to adder circuits [Held & al. TALG 2017]
  3. Turn adder circuits to SAT [Eén & al. JSBMC 2006]
  4. Every clause with > 3 literals is "linearized" [Karp 1972]
     e.g. $y_1 \vee y_2 \vee y_3 \vee y_4 \longrightarrow (y_1 \vee y_2 \vee w_{34}) \wedge (\bar{w}_{34} \vee y_3 \vee y_4)$
     $\Rightarrow$ yields reduction to 3SAT
  5. Finally, apply reduction $3\text{SAT} \to \text{DGP}_1$ [Saxe 1979]
- **Issue**: every reduction increases instance size superlinearly

A direct BLP $\rightarrow$ DGP$_1$ reduction



- Gadget for i-th constraint of a BLP
- **Fundamental mechanism**:
  H_{ij} can reflect $P_{i,j+1}$ back to P_{ij} if $y_j = 0$ or at distance A_{ij} right of P_{ij} if $y_j = 1$
- The rest guarantees that no other vertex positions are possible
- Target DGP$_1$ instance size is linear in BLP size
- Coefficient is large and detailed proof is 15 pages long 😫

A BLP $\rightarrow$ DGP$_1$ "pseudo"-reduction

- Consider the PSEUDO-BOOLEAN SYSTEM problem:

$$\forall i \leq m \quad \sum_{j \leq n} A_{ij} y_j = b_i \quad \wedge \quad y \in \{0,1\}^n$$

where A, b integer — a.k.a. BLP with equality

- Reformulate by a quadratic penalty function

$$\sum_{i \leq m} \Big(\sum_{j \leq n} A_{ij} y_j - b_i \Big)^2 = 0$$

$$\forall j \leq n \; (y_j^2 = y_j) \Rightarrow \sum_{i \leq m} \Big(\sum_{j \leq n} (A_{ij}^2 - 2 b_i A_{ij}) y_j + \sum_{j \neq \ell} A_{ij} A_{i\ell} y_j y_\ell \Big) = \sum_{i \leq m} b_i^2$$

- Linearize $y_j y_\ell$ to $z_{j\ell}$, get PSEUDO-BOOLEAN EQUATION
- Reduce (a relaxation of) PSEUDO-BOOLEAN SYSTEM to DGP$_1$

linearizing constraints $z_{j\ell} = y_j y_\ell$ can be enforced by some solvers

- Gives small DGP$_1$ instances

DGP solvers

Branch-and-Prune (BP) for the DGP

Algorithm for all incongruent rlz x of a DGP instance

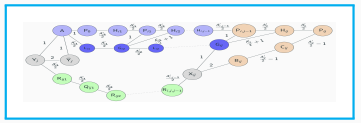
- Introduce a **vertex order** $(V, <)$ defining adjacent predecessors:
  $\forall v \in V$ let $\bar{U}_v = \{u \in V \mid u < v \wedge \{u, v\} \in E\}$
- $\forall \{u, v\} \in E$ let $\mathbb{S}^{K-1}(u, v)$: sphere centered at x_u with radius d_{uv}
- **Input**: $(K, G=(V, E, d))$, order $(V, <)$ s.t. $\forall v \in V \; |\bar{U}_v| \geq \min(K, v)$
  $\forall v \in V$ select $U_v \subseteq \bar{U}_v$ s.t. $|U_v| = \min(K, v)$ e.g. contiguous to $v \Rightarrow$ FPT
- **Output**: all incongruent realizations x of G in $\mathbb{R}^K$
- **BP algorithm** $\boxed{\mathrm{BP}(v, \bar{x}, X)}$ *recursive formulation*, starts with $x_1 = 0$
  1. arguments: $v > 1$, partial rlz $\bar{x} = (x_1, \ldots, x_{v-1})$, rlz set X
  2. compute $S = \bigcap_{u \in U_v} \mathbb{S}^{K-1}(u, v)$ by K-lateration $\Rightarrow \mathbb{P}(|S| \in \{0, 1, 2\}) = 1$
  3. $\forall x_v \in S, \{u, v\} \in E \; (u < v \wedge u \notin U_v \wedge \|x_u - x_v\|_2 = d_{uv})$:
     3.1 let $x = (\bar{x}, x_v)$
     3.2 **if** $v = n$ append x to X, **else** call $\mathrm{BP}(v + 1, x, X)$
- Runs on a real RAM [Blum, Shub & Smale 1989]

Branch-and-Prune for the DGP$_1$ (BP1)

Algorithm for all incongruent rlz x of a DGP$_1$ instance

- Introduce a **vertex order** $(V, <)$ defining adjacent predecessors:
  $\forall v \in V$ let $\bar{U}_v = \{u \in V \mid u < v \wedge \{u, v\} \in E\}$
- $\forall \{u, v\} \in E$ let $\mathbb{S}^{K-1}(u, v)$: sphere centered at x_u with radius d_{uv}
- **Input**: Connected $G = (V, E, d) \Rightarrow \exists$ order $(V, <)$ s.t. $\forall v \in V\ |\bar{U}_v| \geq 1$
  $\forall v \in V$ select a single adj. pred. $u_v \in \bar{U}_v$
- **Output**: all incongruent realizations x of G in $\mathbb{Q}^1$
- **BP1 algorithm** $\boxed{\text{BP1}(v, \bar{x}, X)}$ *recursive formulation*, starts with $x_1 = 0$
  1. **input**: $v > 1$, partial rlz $\bar{x} = (x_1, \ldots, x_{v-1})$, rlz set X
  2. let $S = \mathcal{S}(u_v, v) \cap_{u \in U_v} \mathbb{S}^{K-1}$ $|S| = 2$ in general; see below for $|S| = 1$
  3. $\forall x_v \in S, \{u, v\} \in E\ (u < v \wedge u \notin U_v \wedge |x_u - x_v| = d_{uv})$:
     3.1 let $x = (\bar{x}, x_v)$
     3.2 **if** $v = n$ append x to X, **else** call BP$(v + 1, x, X)$
- Runs on a TM [Turing 1937]

More on BP1

- BP1 is a deterministic branching algorithm running on a TM
  BP for $K > 1$ is probabilistic and prone to floating point errors
- BP1 only only requires a DGP_1 instance with a connected graph
  BP for $K > 1$ also requires a nontrivial vertex order which only exists for certain rigid graphs, and is generally hard to find
- Reductions from SAT, SUBSET-SUM, BLP:
  can be simplified by directing some edges $\{u, v\}$ in BP1
  - set $\mathcal{S}(u, v) = \{x_u + d_{uv}\}$ if (u, v) or $= \{x_u - d_{uv}\}$ if (v, u)
  - no branching at v since $|\mathcal{S}(u, v)| = 1$
  - SAT: the horrendous gadget  simplifies
  - SUBSET-SUM: for completing partial solutions
  - BLP: to reconstruct $z_{uv} = y_u y_v$ lost in relaxation
- Vertex orders can be constructed along with the BP1
  e.g. depth-first or breadth-first, but also others
- Combinatorial heuristics can be added to BP1
  borrowed from problems that reduce to DGP_1
  e.g. SAT: probing, arc-consistency, VSIDS

Mathematical Programming (MP) based solvers for the DGP

- MP is undecidable $\Rightarrow$ it can model any problem [Jeroslow 1973]
- (continuous) Polynomial Programming (PP): decidable [Tarski 1951]
  at least **NP**-hard; best alg. is doubly exponential
- Quadratically Constrained Programming (QCP): **NP**-hard
  $\Rightarrow$ model DGP as QCP $\{x \mid \forall \{u,v\} \in E \ \|x_u - x_v\|^2 = d_{uv}^2\}$
- **Reformulations:**
  - Slacked: $\min_{x,s} \{ \sum_{\{u,v\} \in E} s_{uv}^2 \mid \forall \{u,v\} \in E \quad \|x_u - x_v\|^2 = d_{uv}^2 + s_{uv} \}$
  - Quartic unconstrained: $\min_x \sum_{\{u,v\} \in E} (\|x_u - x_v\|^2 - d_{uv}^2)^2$
  - Push-&-Pull: $\max_x \{ \sum_{\{u,v\} \in E} \|x_u - x_v\|^2 \mid \forall \{u,v\} \in E \ \|x_u - x_v\|^2 \leq d_{uv}^2 \}$
    note: concave objective but convex constraints
- **Variable symbols are back!** But only for solving purposes
  fast local solvers: gradient descent (GD), Interior Point Methods (IPM)
- $\Rightarrow$ Obtain candidate approximate rlz in $K > 1$
  for relaxation purposes only, see later
- DGP_1 can be formulated as BLP $\Rightarrow \exists$ reduction $\mathrm{DGP}_1 \to$ BLP
  can solve DGP_1/BLP by BP1 or MP solvers: useful for comparisons

Dimensional relaxations and heuristics for the DGP_1

The spectrum of DGP hardness

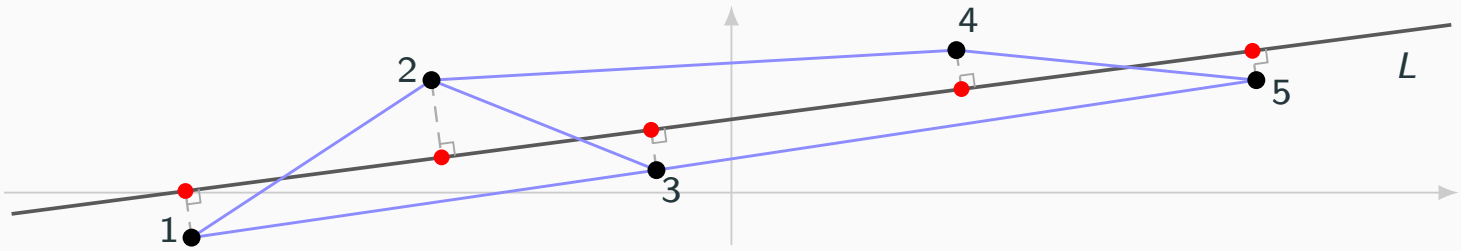
- The DGP_K is **NP**-hard for any $K < |V| - 1 = n - 1$ [Saxe 1979]
- The DGP_{n-1} can be solved in polytime for any given $\epsilon > 0$ precision
  precision issue: because DGP_K may not be in **NP** for $K > 1$
- Use a polytime IPM on the following SDP formulation:

$$\forall \{u, v\} \in E \quad X_{uu} + X_{vv} - 2X_{uv} = d_{uv}^2 \quad \wedge \quad X \succeq 0 \qquad (1)$$

- Consider soln $\bar{X}$: PSD by construction $\Rightarrow \exists \bar{x} \in \mathbb{R}^{n \times n}$ s.t. $\bar{x}^{\mathsf{T}} \bar{x} = \bar{X}$
- $\bar{x}$ satisfies $\forall \{u, v\} \in E \; \|x_u - x_v\|_2^2 = d_{uv}^2$ by Eq. (1)
  $\bar{x}$ is a valid rlz for a DGP instance (V, E, d) with $K = n$
  centering yields lin dep $\Rightarrow K = n - 1$
- DGP_1 are hardest, DGP_{n-1} & DGP_n are in **P** up to $\epsilon > 0$
  The more space there is to realize a graph, the easier the DGP
- **Prop.** Consider DGP instance (K, G) with $K < n - 1$. (i) If feasible, then $(K + 1, G)$ is feasible. (ii) Conversely, for any $K > 1$ there exist DGP instances (K, G) such that $(K - 1, G)$ is infeasible.
  (i) $\mathbb{R}^K \subset \mathbb{R}^{K+1}$ (ii) consider K-dimensional equilateral simplices

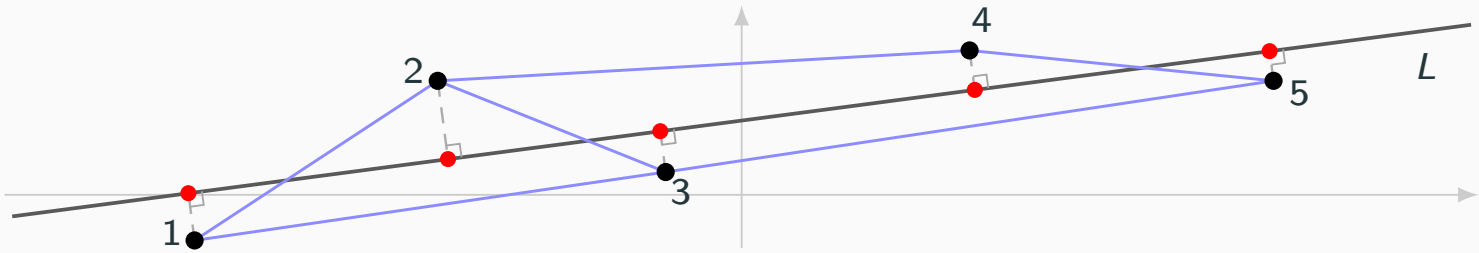
DGP on a thin slab or cylinder

- By previous **relaxation prop.** we realize DGP_1 instances in $\mathbb{R}^K$
- Want rlz close to a line $\Rightarrow$ restrict vertical dim to slab/cylinder
- Relaxation must be fast $\Rightarrow$ consider fast local solvers in $\mathbb{R}^K$
  e.g. DGP_2 on a slab $[-\sigma, \sigma]$: $\min\limits_{\substack{x_1 \in \mathbb{R} \\ x_2 \in [-\sigma,\sigma]}} \sum\limits_{\{u,v\} \in E} (\|x_u - x_v\|^2 - d_{uv}^2)^2$
- If soln $\bar{x}$ has error ≈ 0 accept, othw increase σ and/or K & repeat
  may also consider cylinder around $\mathbb{R}^1$ instead of slab
- Orthogonal regression of $\bar{x}_1, \ldots, \bar{x}_n$ identifies best line L
  project $\bar{x}$ to L, obtain 1-dimensional approximate rlz $\hat{x}$



- Valid rlz in $\mathbb{R}^1$ characterized by **order of points** of $\hat{x}$ on $\mathbb{R}^1$
  retrieve rlz x^* from $\hat{x}$: could fail, but failures $\Rightarrow$ approximate solutions
- **Find solns of combinatorial problems w/continuous methods**

Projection on PCA line (orthogonal regression)



- Fitted line is $L = \{c + tv \mid t \in \mathbb{R}\}$
  c a point in L, $v \in \mathbb{S}^{K-1}$ its direction vector
- Want to find L s.t. $\min\{\sum_{v \le n} \text{dist}(\bar{x}_i, L)^2 \mid c \in \mathbb{R}^K, v \in \mathbb{S}^{K-1}\}$
- For a line with direction v, orthog proj of $\bar{x}_i - c$ onto v is $vv^\mathsf{T}(\bar{x}_i - c)$
  so squared distance to L is $\|(I - vv^\mathsf{T})(\bar{x}_i - c)\|_2^2$
- $\Rightarrow$ solve $\min\{\sum_{i \le n} \|(I - vv^\mathsf{T})(\bar{x}_i - c)\|_2^2 \mid c \in \mathbb{R}^K \wedge \|v\|_2 = 1\}$ by PCA
  1. center pts by centroid $c = \frac{1}{n}\sum_{i \le n} \bar{x}_i$: $\forall i \le n$ let $z_i = \bar{x}_i - c$
  2. SVD of centered data matrix $Z = (z_i^\mathsf{T})_i = U\Sigma V^\mathsf{T}$
     where $\Sigma = \text{diag}(\sigma)$ lists singular values σ_i in decreasing order
  3. return $L = \{c + tv \mid t \in \mathbb{R}\}$ where $v = \text{col}(V, 1) \longleftrightarrow \sigma_1 = \max(\sigma)$
- Now project $\bar{x}$ on L: $\forall i \le n\ (t_i = v^\mathsf{T}(\bar{x}_i - c)) \wedge (\hat{x}_i = c + t_i v)$, **get** $\hat{x}$

Reconstruction of the realization from the order

- Assume a DGP_1 connected graph instance $G = (V, E, d)$ is YES

  rlz $x = (x_1, \ldots, x_n)$

  $\pi =$ **vertex order induced by x in $\mathbb{R}^1$**

- **Proposition** π is a YES certificate of G

  Proof. $\mathrm{DGP}_1 \equiv \forall \{u, v\} \in E \; |x_u - x_v| = d_{uv}$. Use the order π to determine if $x_u - x_v = +d_{uv}$ or $x_u - x_v = -d_{uv}$ $\forall \{u, v\} \in E$. Obtain linear system, can be solved in polytime: its solution x^* yields a YES rlz.

- **Exact algorithm** for any given vertex order π:
  1. Set $x^* = \bot$, $x^*_{v_1} = 0$, and traverse a spanning tree T of G rooted at v_1
  2. Traversal yields edges $\{u, w\}$ s.t. $x^*_u \neq \bot \wedge x^*_w = \bot$: let

$$x^*_w = \begin{cases} x^*_u + d_{uw} & \text{if} \quad u \prec_\pi w \\ x^*_u - d_{uw} & \text{if} \quad u \succ_\pi w \end{cases}$$

  3. For all $\{u, v\} \in E \setminus T$ check that

$$x^*_w - x^*_u = \begin{cases} d_{uw} & \text{if} \quad u \prec_\pi w \\ -d_{uw} & \text{if} \quad u \succ_\pi w \end{cases}$$

     order π is **a YES certificate** iff check succeeds

  - **Note: x^* may not be $= x$ nor correspond to π**

    local order variations are possible, e.g. partial reflections

- If check fails: x is a wrong rlz, π wrong order

Heuristic reconstruction

- What if projection+reconstruction ($\bar{x} \to \hat{x} \to \pi$) yields a wrong π?
  - rlz x s.t. $\boxed{\forall \{u,v\} \in E\ x_u - x_v = s_{uv}(\pi) d_{uv}}$ with $s_{uv} = \begin{cases} +1 \text{ if } u \prec_\pi v \\ -1 \text{ if } u \succ_\pi v \end{cases}$
  - π wrong $\Rightarrow$ above linear system $\boxed{Bx = b^\pi}$ is infeasible
- Best reconstruction: $\tilde{x} = \arg\min_x \|Bx - b^\pi\|_2^2$
  - DGP_1 instance is YES $\Rightarrow \exists b^*$ s.t. $Bx = b^*$ has soln x^*
  - let $\mathcal{B}(\pi) = \{e \in E \mid b^\pi_e \neq b^*_e\}$: $s \in \pm 1 \Rightarrow b^\pi_e = -b^*_e$ if e wrong
  - $\Rightarrow b^\pi_e - b^*_e = \pm 2 d_e \Rightarrow \mathrm{err}(\pi) \equiv \|b^* - b^\pi\|_2^2 \leq 4 \sum_{e \in \mathcal{B}(\pi)} d_e^2$
  - $\Rightarrow \mathbb{E}(\mathrm{err}(\pi)) \leq 4 \sum_{e \in E} d_e^2 \mathbb{P}(e \in \mathcal{B}(\pi)) \quad (*)$
- Edge errors as probabilities $\mathbb{P}(e \in \mathcal{B}(\pi))$
  - let $\rho_{uv} = \left| \|\bar{x}_u - \bar{x}_v\|_2 - d_{uv} \right|$ (2D rlz), $g_{uv} = |\tilde{x}_u - \tilde{x}_v|$ (reconstr 1D rlz)
  - *confidence*: $c_{uv} = \frac{g_{uv}}{\rho_{uv} + d_{uv} + \alpha}$ for some $\alpha > 0$, *risk*: $r_{uv} = 1 - c_{uv}$
  - use r_{uv} as $\mathbb{P}(e \in \mathcal{B}(\pi))$, obtain error estimation on π by $(*)$

$$\mathbb{E}(\mathrm{err}(\pi)) \leq 4 \sum_{\{u,v\} \in E} d_{uv}^2 r_{uv}$$

- Heuristic reconstruction $\tilde{x}$ minimizes the error

  wrong $\pi \Rightarrow$ error > 0

A double homotopy method

- Sequence of local solver runs applied to related problems
  problems vary in function of topological parameters ϑ
  $\vartheta = (k, \sigma)$ =(dimension of hosting space, cylinder radius about x_1 axis)
  problem sequence $\{P^i(\vartheta)\}_i \longrightarrow$ problem of interest P
- Soln $\hat{x}^{i-1}$ of previous call = starting point x_0^i for next call
- Start from $k =$ size estimate of max clique in DGP_1 instance G
  if $(k+1)$-cliques are realizable, they can be realized in $\mathbb{R}^k$
- Start from $\sigma = O(\max_{\{u,v\} \in E} d_{uv})$ where $G = (V, E, d)$
  allows an initial rotation of the longest edge
- Algorithm:
  1. descent on k
  2. for each k, descent on σ
  3. for each (k, σ) subproblem:
     soln $\bar{x} \to$ orthogonal regression $\to \hat{x}$ on $\mathbb{R}^1 \to$ order π on $V \to$ rlz $\tilde{x}$
  4. if small edge error on $\tilde{x}$ stop

Branch-and-Bound (BB) for the DGP_1

A DGP_1 formulation with binary variables in $\{+1, -1\}$

- **Lemma**: for $t \in \mathbb{R}$ and $d > 0$, $\big|\,|t| - d\big| = \min\limits_{s \in \pm 1} |t - sd|$

  Proof: by inspection
- Consider DGP formulation $\min\limits_{x \in \mathbb{R}^{nK}} \sum\limits_{\{u,v\} \in E} \big|\,\|x_u - x_v\|_2 - d_{uv}\big|$
  - specialize to $K = 1$, get: $\min\limits_{x \in \mathbb{R}^n} \sum\limits_{\{u,v\} \in E} \big|\,|x_u - x_v| - d_{uv}\big|$
  - apply Lemma, get: $\min\limits_{x \in \mathbb{R}^n} \sum\limits_{\{u,v\} \in E} \min\limits_{s_{uv} \in \pm 1} |x_u - x_v - s_{uv} d_{uv}|$
  - commute inner min, get: $\min\limits_{s \in (\pm 1)^m} \min\limits_{x \in \mathbb{R}^n} \sum\limits_{\{u,v\} \in E} |x_u - x_v - s_{uv} d_{uv}|$
  - let $D \in \mathbb{R}^{m \times n}$ s.t. (u, v)-th row of (Dx) is $x_v - x_u$ (*rigidity matrix*)
  - Hadamard product of s, d is $s \circ d = (s_{uv} d_{uv} | \{u, v\} \in E)$
- Obtain DGP_1 formulation

$$\min_{\substack{s \in (\pm 1)^m \\ x \in \mathbb{R}^n}} \|Dx - s \circ d\|_1 \tag{2}$$

A BB algorithm for the DGP_1

- BB based on Eq. (2): $\min_{\substack{s\in(\pm\mathbf{1})^m \\ x\in\mathbb{R}^n}} \|Dx - s\circ d\|_1$
- Compute initial upper bound (UB) using heuristic reconstruction
- Branching occurs on edge signs $s_{uv} \in \{1, -1\}$
- A single branch contains at most $m = |E|$ nodes
  - if $|\text{branch}| = m$ then s is known and Eq. (2) is an LP
  - solve LP, update UB if incumbent improves
- Compute relaxation from a fundamental cycle basis (FCB)
  yields a lower bound (LB), see later
- If LB $\geq$ UB, prune branch by bound
  othw branch on next unfixed edge $\{u, v\}$

Cycle-based relaxation

- Consider DGP_1 instance graph $G = (V, E, d)$
  for any $F \subseteq E$ and rlz of F define $\mathrm{err}(F, x) = \sum_{\{u,v\} \in F} \big| |x_u - x_v| - d_{uv} \big|$
- **Prop.** Let $\gamma_1, \ldots, \gamma_\ell \subset E$ be simple cycles of G, and x the best rlz for all of them. If they are pairwise disjoint, $\sum_{h \leq \ell} \mathrm{err}(\gamma_h, x) \leq \mathrm{err}(E, x)$

  Proof. Each single cycle clearly has $\mathrm{err}(\gamma_h, x) \leq \mathrm{err}(E, x)$ since $E \supseteq \gamma_h$.
- In other words, if γ are pairwise disjoint, the best rlz x can be optimal for each single cycle, while the best rlz for the whole graph can only be worse

Cycle-based relaxation algorithm

FCB: given any spanning tree (SPT) T, each non-tree edge determines a *fundamental cycle* (FC); these FCs are a basis of the cycle space

1. Compute a FCB by any SPT T, get a cycle basis $\mathcal{C}$
2. Realize each cycle $\gamma \in \mathcal{C}$ using Dynamic Programming (DP)
   - reduction PARTITION $\rightarrow$ DGP_1 on cycles is bijective
   - $\Rightarrow$ DGP_1 on cycles can be modelled by PARTITION
   - PARTITION modelled as SUBSET-SUM, which can be solved by DP
   - DP can solve NO instances with error $\min |\sum_j a_j y_j - b|$
3. Sort cycles in $\mathcal{C}$ by decreasing error
4. Select max error cycle $\gamma \in \mathcal{C}$
5. Add γ error to LB, remove all cycles η s.t. $\gamma \cap \eta \neq \varnothing$
6. Terminate when $\mathcal{C} = \varnothing$, return LB

Partial validation experiments

Branch-and-Prune

input params				*output params*		
type	$\|V\|$	$\|E\|$	cycles	$\|X\|$	cpu-all	cpu-1st
cycle	10	10	1	3	0.00	0.00
cycle	20	20	1	1024	0.02	0.00
cycle	30	30	1	595924	7.87	0.00
cycle	40	40	1	>3.5M	>120	0.00
cycle	50	50	1	same	same	0.00
cycle	60	60	1	same	same	20.00
cycle	70	70	1	same	same	2.36
cycle	80	80	1	same	same	0.41
cycle	90	90	1	?	>120	>120
fcb	100	109	10	>1e5	>120	0.35
fcb	100	111	12	519	4.13	1.33
fcb	100	1099	1000	1	0.00	0.00
fcb	100	2099	2000	1	0.00	0.00
fcb	100	3099	3000	1	0.01	0.01
fcb	500	599	100	?	>120	>120
fcb	500	3099	3000	1	0.01	0.01
fcb	500	6099	6000	1	0.01	0.01

long cycles are hardest (they encode PARTITION)

Double homotopy

input params						*output params*
type	$\|V\|$	$\|E\|$	cycles	error (MDE)	cpu	cpu-bp1
cycle	60	60	1	0.367	0.33	20.00
cycle	70	70	1	1.943	0.26	2.36
cycle	90	90	1	1.733	0.31	>120
fcb	100	111	12	15.67	0.59	1.33
fcb	500	599	100	104.21	66.02	>120

Whole pipeline validated on a single small random BLP instance solved as a DGP using BP1