# The Revolution in Graph Theoretic Optimization Problem

Gary L Miller

Carnegie Mellon University

Simons Open Lecture

Oct 6, 2014

# JOINT WORK

Guy Blelloch,
Hui Han Chin,
Michael Cohen,
Anupam Gupta,
Jonathan Kelner,
Yiannis Koutis,
Alexsander Madry,
Jakub Pachocki,
Richard Peng,
Kanat Tangwongsan,
Shen Chen Xu

# OUTLINE

- Linear system solvers
- Regression and Image Denoising.
- Simple formulation and connection with solving linear systems.
- Overview of SDD solvers.
- A better $L_1$ formulation of denoising.
- Maximum flow using solvers
- New for 2013-14

# SPECTRAL GRAPH THEORY
# LAPLACIAN PARADIGM

Use graph algorithms to solve linear algebra problems.

Use linear algebra to solve graph problems.

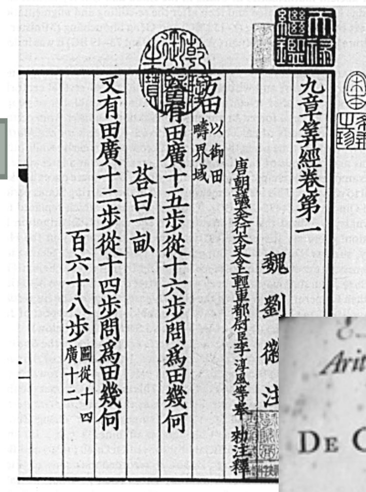
Use both to solve optimization problems

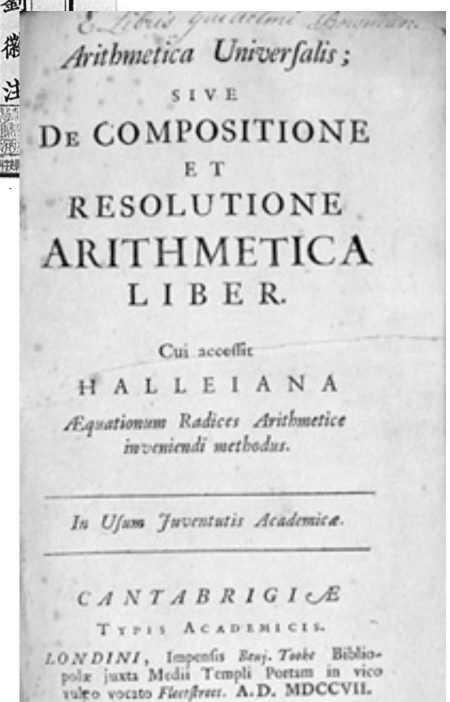
# OLDEST COMPUTATIONAL PROBLEM

$$\begin{pmatrix} 3 & 2 & -1 \\ 2 & -5 & 4 \\ -1 & 1/2 & 2 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} 3 \\ 7 \\ 2 \end{pmatrix}$$

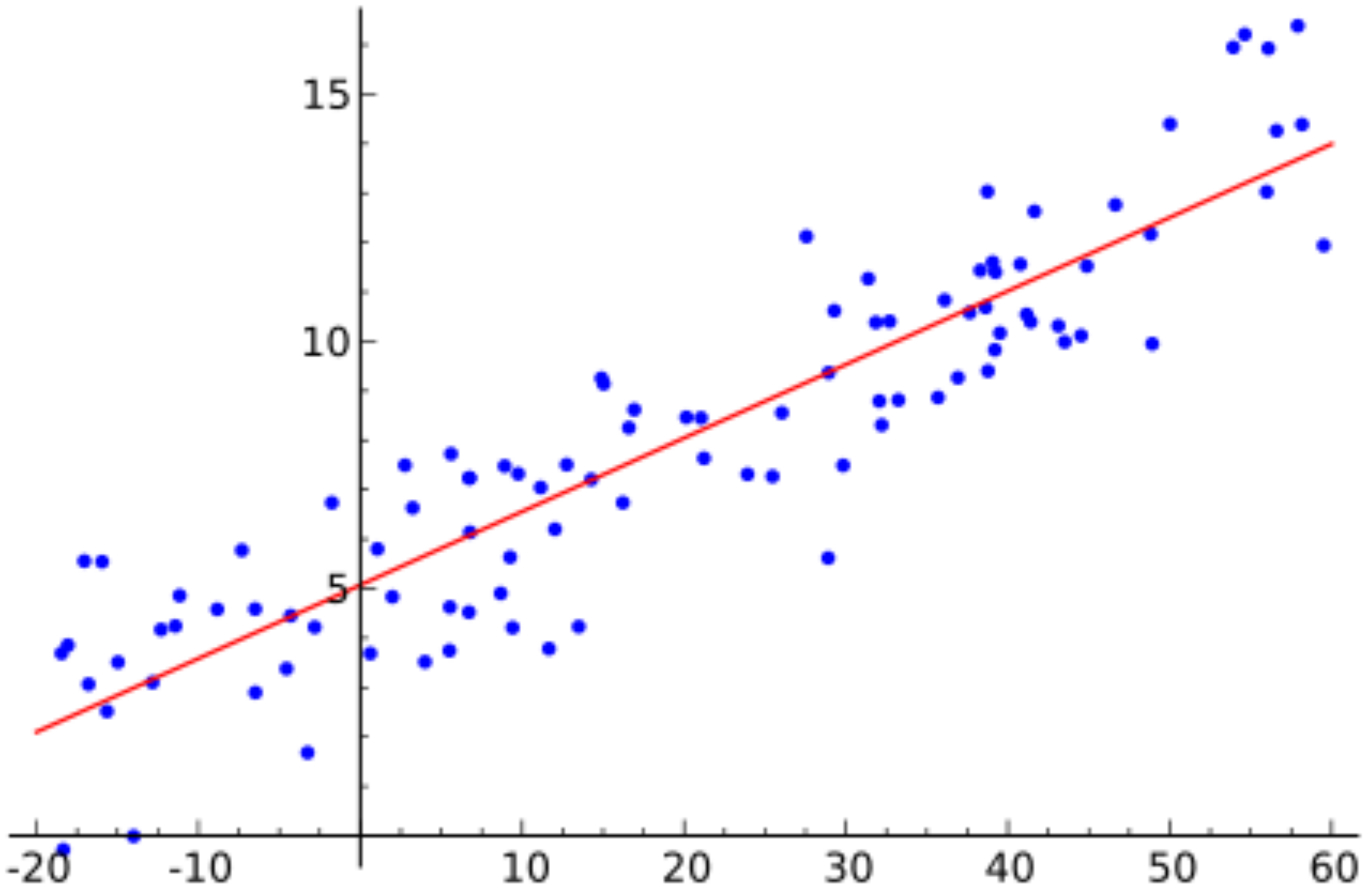
# DIRECT LINEAR SYSTEM SOLVES

- [1st century CE] Gaussian Elimination: $O(n^3)$
- [Strassen `69] $O(n^{2.8})$
- [Coppersmith-Winograd `90] $O(n^{2.3755})$
- [Stothers `10] $O(n^{2.3737})$
- [Vassilevska Williams`11] $O(n^{2.3727})$
- [George `73], [Lipton-Rose-Tarjan `80], [Alon-Yuster `10] Faster direct methods for special non-zero structures.


九章筭經卷第一
魏 劉 徽 注
唐朝議大夫行太史令上輕車都尉臣李淳風等奉 勅注釋
方田 以御田疇界域
今有田廣十五步從十六步問爲田幾何
荅曰一畝
又有田廣十二步從十四步問爲田幾何
荅曰一畝


Arithmetica Universalis;
SIVE
DE COMPOSITIONE
ET
RESOLUTIONE
ARITHMETICA
LIBER.
Cui accessit
HALLEIANA
Æquationum Radices Arithmetice inveniendi methodus.
In Usum Juventutis Academicæ.
CANTABRIGIÆ
TYPIS ACADEMICIS.
LONDINI, Impensis Benj. Tooke Bibliopolæ juxta Medii Templi Portam in vico vulgo vocato Fleetstreet. A.D. MDCCVII.

# REGRESSION



# OVER CONSTRAINED SYSTEMS

Over Constrained System: $A\ x = b$.

Solve system $A^T A\ x = A^T b$

- Matrix $A^T A$ is Symmetric Positive Semi-Definite (SPSD).
- Open Question:
  Find sub-quadratic time solvers for SPD systems?

We will need problems with an underlying graph.

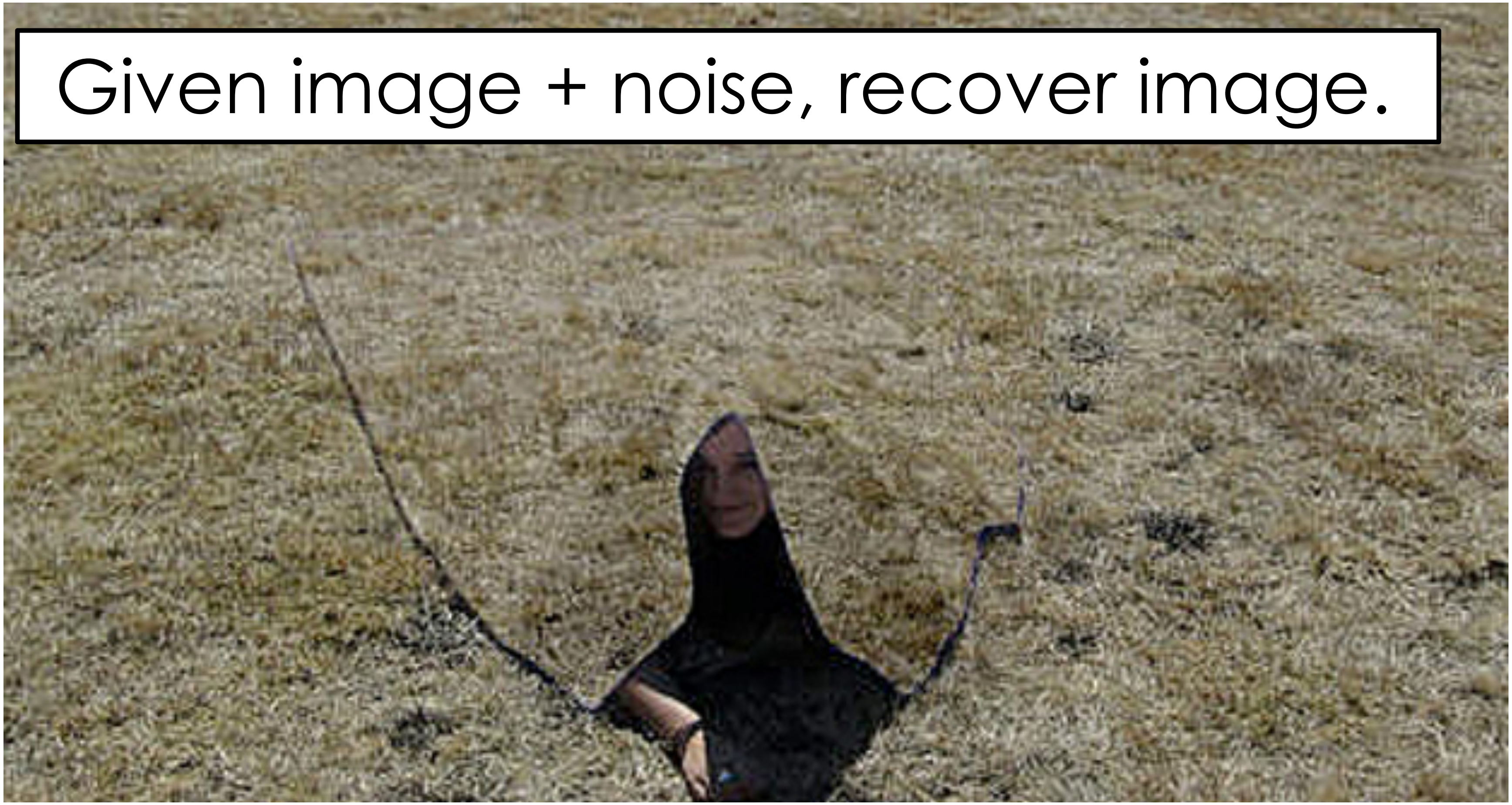
# APPROXIMATION ALGORITHMS

- Whole conferences NP-Approximation
- Same ideas and goals can be  applied problems in Polytime.
- Our goal is find  good approximation but much faster than known exact solutions.
- Maybe even faster exact solutions!

# CLASSIC REGRESSION PROBLEM

- Image Denoising
- Critical step in image segmentation and detection
- Good denoising makes the segmentation almost obvious.

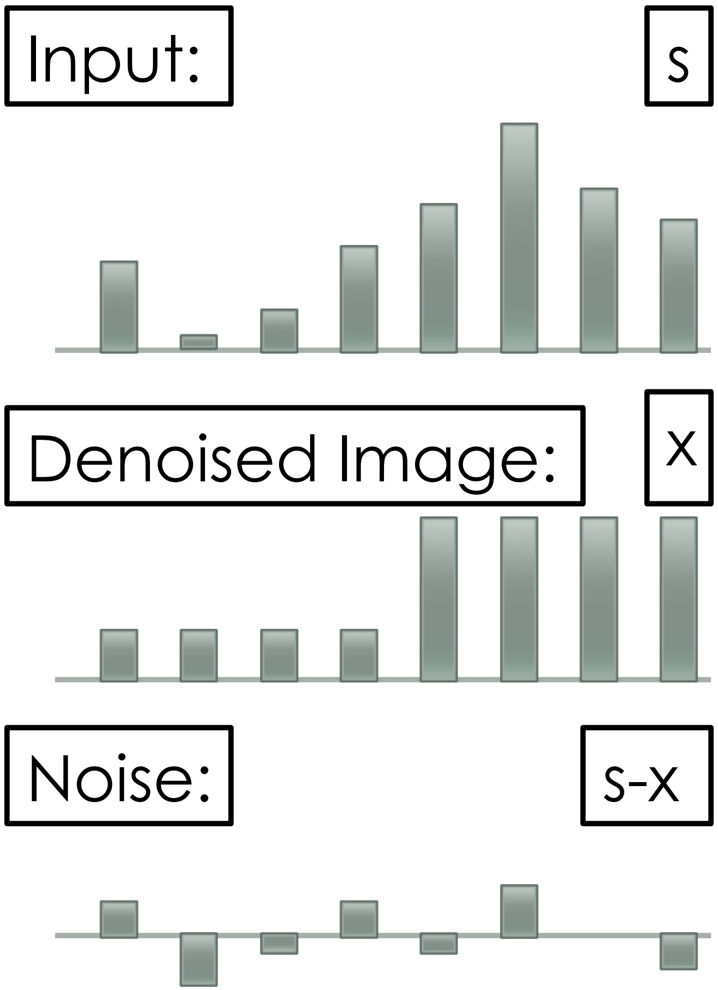
# CAMOUFLAGE DETECTION



# CAMOUFLAGE DETECTION

Hui-Han Chin

# IMAGE DENOISING: THE MODEL




- Assume there exist a 'original' noiseless image.
- Noise generated from some distribution.
- Input: original + noise.
- Goal: approx the original image.

# CONDITIONS ON X

- Noise is small: denoised image should still be close to image + noise.
- Real images are 'smooth' except at boundaries.

Function of (x-s)

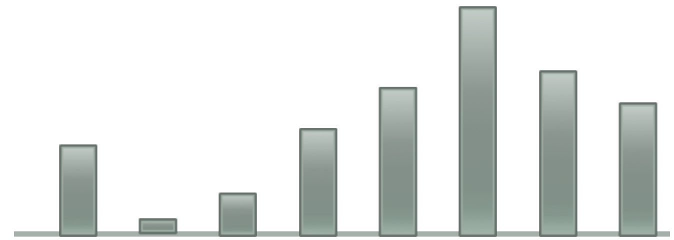
Fidelity(x, s)

Function computed on differences of neighboring pixels of x

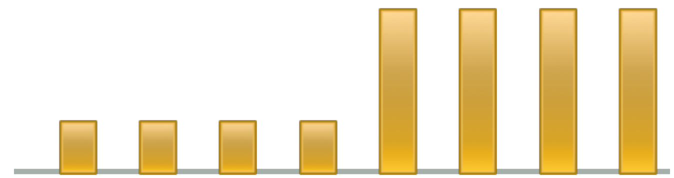
Smoothness(x)

# ENERGY FUNCTION

Noisy image: s



Candidate solution: x



Fidelity(x, s) + Smoothness(x)

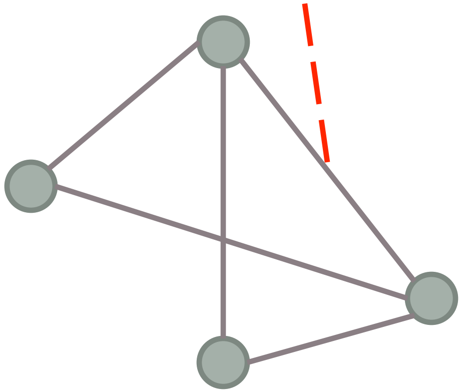
$(x_i - s_i)^2$ summed over all pixels i.

$(x_i - x_j)^2$ summed over all neighbor pixels i, j.

This is a toy example

# MATRICES ARISING FROM IMAGE PROBLEM HAVE NICE STRUCTURES

$$\begin{bmatrix} 3 & -1 & -1 & -1 \\ -1 & 2 & -1 & 0 \\ -1 & -1 & 3 & -1 \\ -1 & 0 & -1 & 2 \end{bmatrix}$$



A is Symmetric Diagonally Dominant (SDD)

- Symmetric.
- Diagonal entry ≥ sum of absolute values of all off diagonals.

# OPTIMIZATION PROBLEMS IN CS

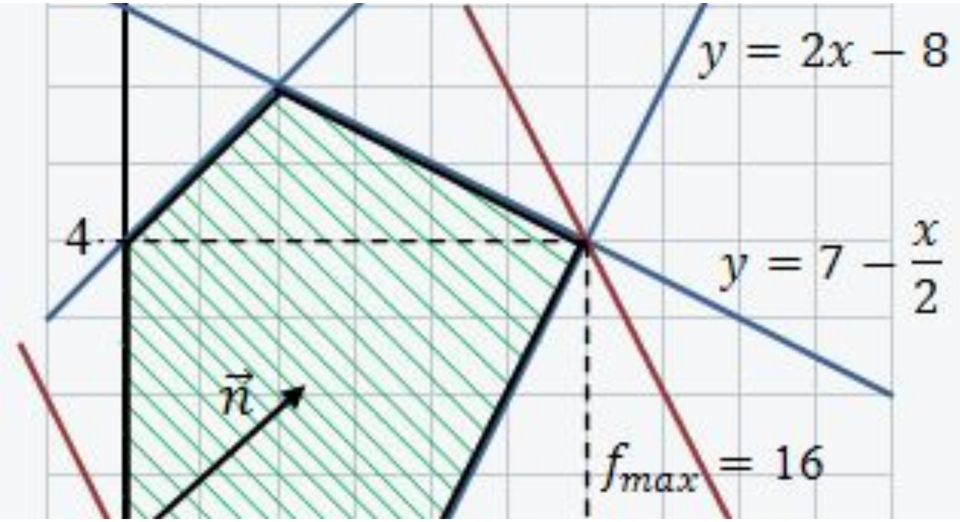
Many algorithm problems in CS are optimization problems with underlying graph.

- Maximum flow in a graph.
- Shortest path in a graph.
- Maximum Matching.
- Scheduling
- Minimum cut.

Some of these do not seem to have an underlying graph.

Longest common subsequence.

# LINEAR PROGRAMMING



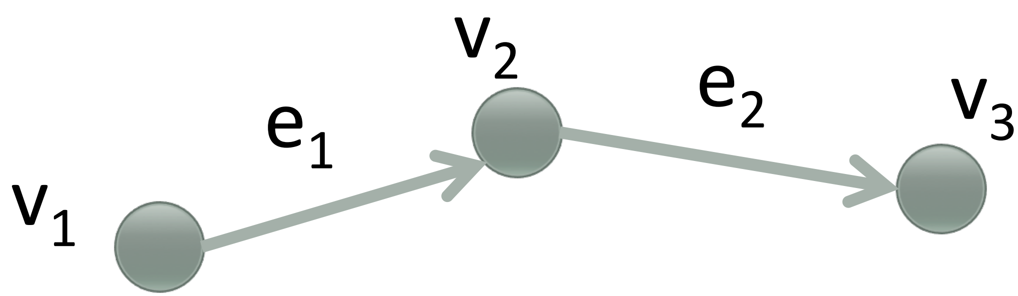

Many optimization problems can be written as an LP

EG: Single source shortest path.

Is this useful?

# LAPLACIAN PRIMER

- Matrix view of flows
- The Boundary Map B: Flow ➔ Residual Flow




$$
\begin{array}{c}
\\
v_1 \\
v_2 \\
v_3
\end{array}
\begin{array}{c}
\begin{array}{cc} e_1 & e_2 \end{array} \\
\begin{pmatrix} -1 & 0 \\ +1 & -1 \\ 0 & +1 \end{pmatrix}
\end{array}
\begin{pmatrix} 2 \\ -3 \end{pmatrix} = \begin{pmatrix} -2 \\ 5 \\ -3 \end{pmatrix}
$$

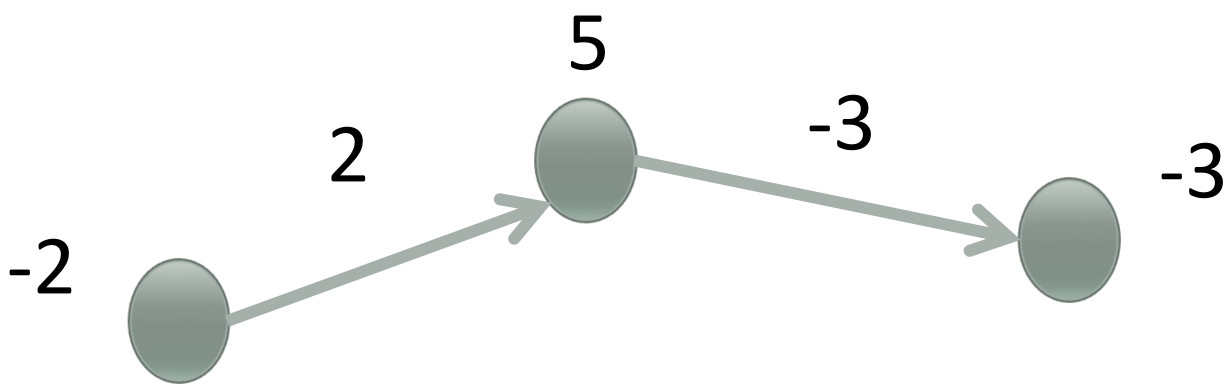
# THE BOUNDARY MAP B

Let $G = (V,E)$ be n vertex m oriented edges graph.

- Def: B is a Vertex by Edge matrix
  where $B_{ij} = +1$ if $v_i$ is head of $e_j$
  $-1$ if $v_i$ is tail of $e_j$
  $0$ otherwise

- Note: If f is a flow then Bf is residual vertex flow.

# BOUNDARY MATRIX




$$\begin{array}{cc} & \begin{array}{cc} \color{blue}{e_1} & \color{blue}{e_2} \end{array} \\ \begin{array}{c} \color{blue}{v_1} \\ \color{blue}{v_2} \\ \color{blue}{v_3} \end{array} & \begin{pmatrix} -1 & 0 \\ +1 & -1 \\ 0 & +1 \end{pmatrix} \end{array} \begin{pmatrix} 2 \\ -3 \end{pmatrix} = \begin{pmatrix} -2 \\ 5 \\ -3 \end{pmatrix}$$

# $B^T$ AND POTENTIAL DROPS

- Let v be a n-vector of potentials
- $B^Tv$ = vector of potential drops.
- $R^{-1}BTv$ = vector of potential drops.
  - R a diagonal matrix of resistive values
  - Ohms law:  Rule to go from potentials to flows.
- Today we set resistors all to one.
- Thus $B^Tv$ = vector of flows.

# GRAPH LAPLACIAN

- Def: $L := BB^T$ , Laplacian of G.
- Goal: Solve $Lv=b$
- Suppose v satisfies $BB^Tv=b$.

  thus $f := B^Tv$ is a flow s.t. $Bf=b$
- What can we say about f?

# CIRCULATIONS AND POTENTIAL FLOWS

- Def: $f_c$ in $R^m$ is a circulation if $Bf_c = 0$
- Def: $f_p$ in $R^m$ is a potential flow if $f_p = B^T v$
- Claim : $f_c$ is orthogonal to $f_p$
  then $f_c^T f_p = 0$

Pf: $f_c^T f_p = f_c^T (B^T v) = (Bf_c)^T v = 0^T v = 0$

# CIRCULATIONS AND POTENTIAL FLOWS

- Note: Dim(potential flows) = n-1

  (G connected)
- Claim : Dim(circulations) = m –n+1

  (The number of nontree edges.)

Pf: Given a spanning tree each nontree edge induces cyclic flow that is independent.

# POTENTIALS AND FLOWS

- Suppose $BB^Tv=b$ then $f=B^Tv$ is a flow s.t $Bf=b$
- Claim: min $f^Tf$ s.t. $Bf=b$ is a potential flow.

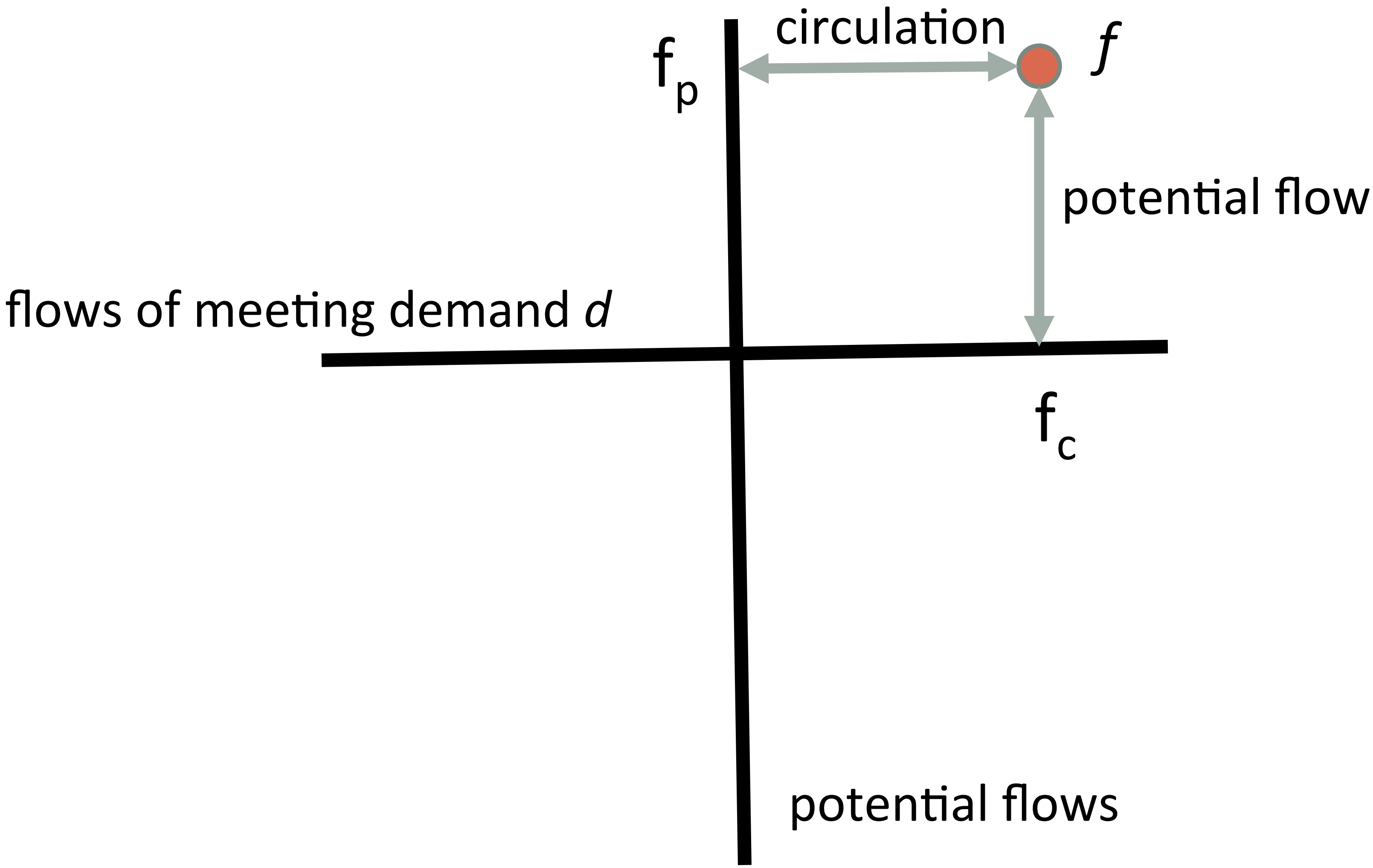
Pf: $f = f_p+f_c$

- $Bf_p = Bf_p + Bf_c = B(f_p+f_c) = b$
- $f^Tf = (f_p+f_c)^T(f_p+f_c)$
  $= f_p^2 + 2f_p^Tf_c + f_c^2$
  $= f_p^2 + f_c^2 \quad \geq f_p^2$

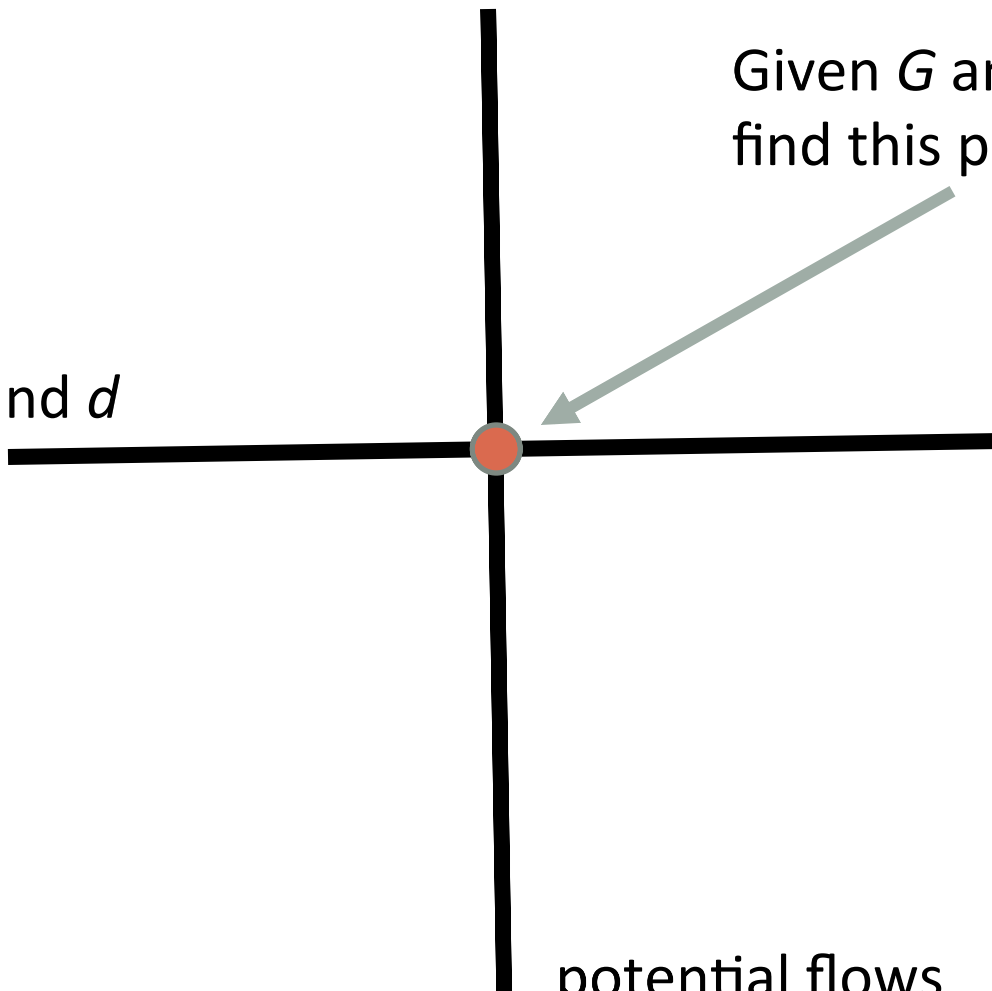
# GRAPH LAPLACIAN SOLVERS

- Def: $L := BB^T$, Laplacian of G.
- Two dual approaches to approximately solving $Lv = b$
- 1) Find a potential that minimizes $Lv-b$
- 2) find a minimum energy flow f s.t. $Bf=b$

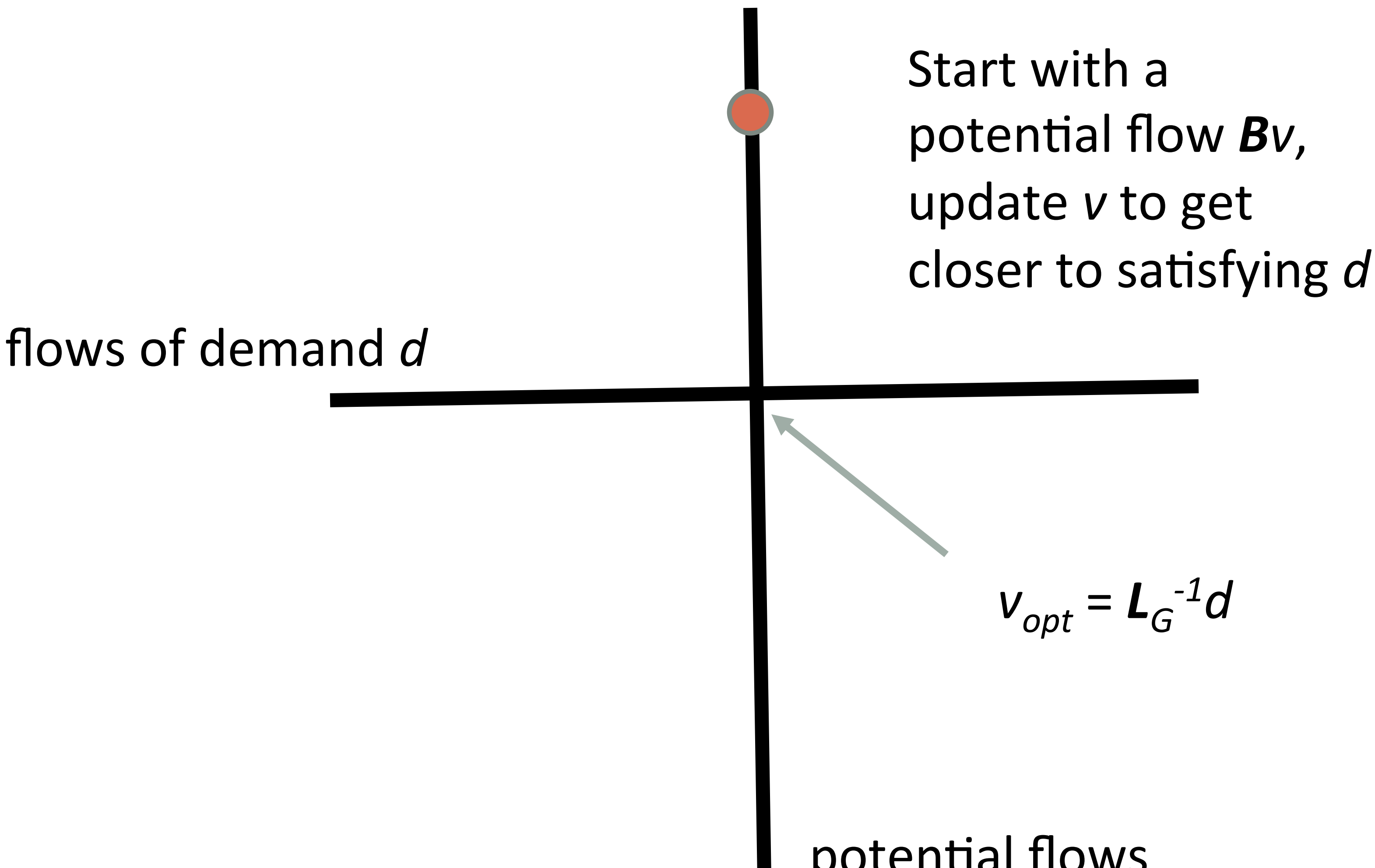
# THE SPACE OF FLOWS



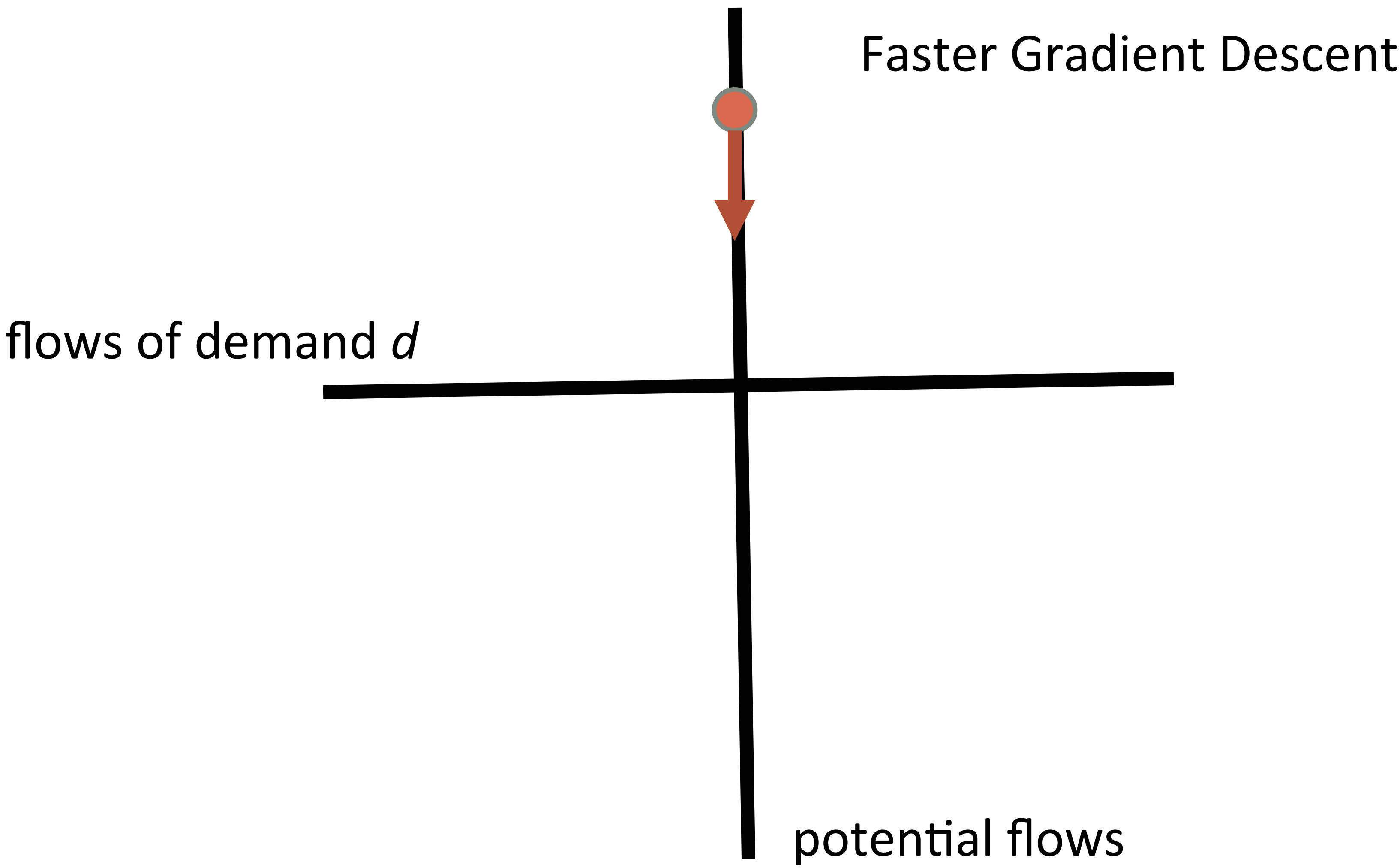
# SOLVING LAPLACIANS



# DUAL APPROACH: SOLVING A LINEAR SYSTEM



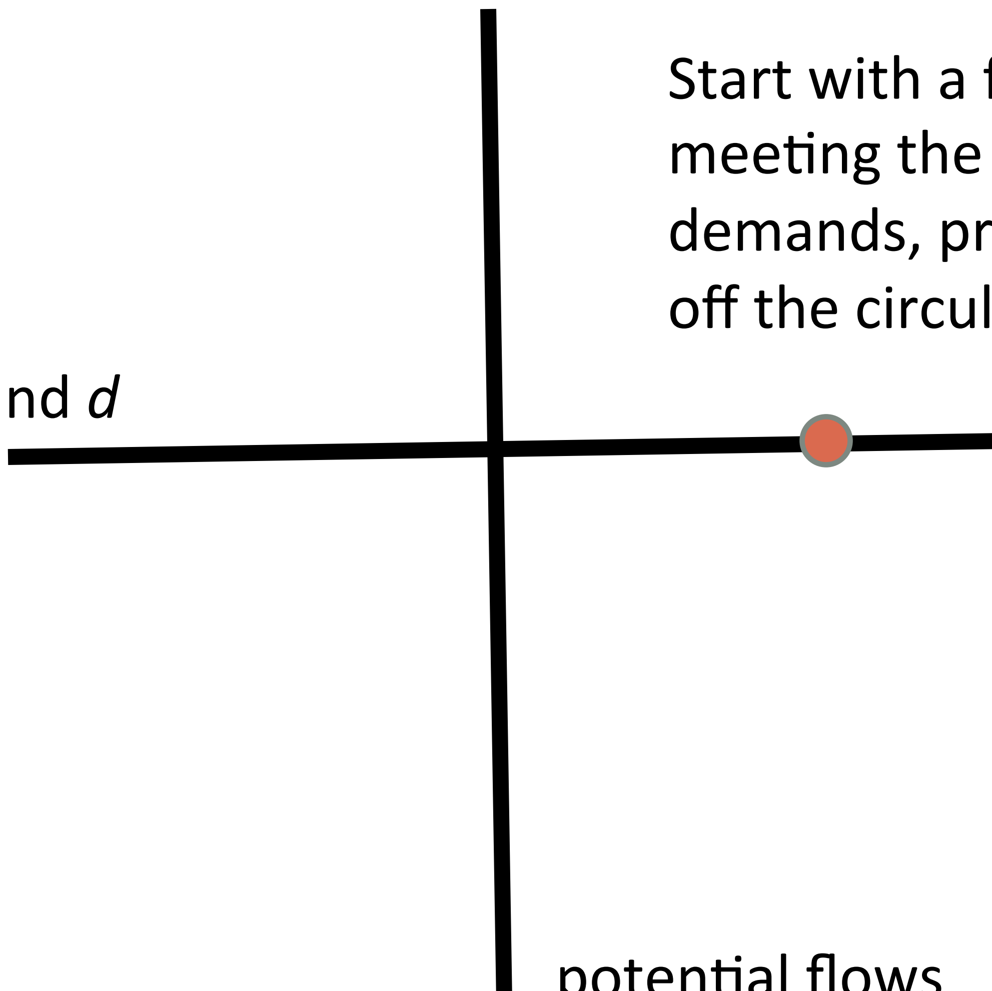
# DUAL APPROACH: SINGLE STEP (ST'04,KMP '10, '11)



# PRIMAL APPROACH: SOLVING A FLOW PROBLEM

Start with a flow meeting the demands, project off the circulation

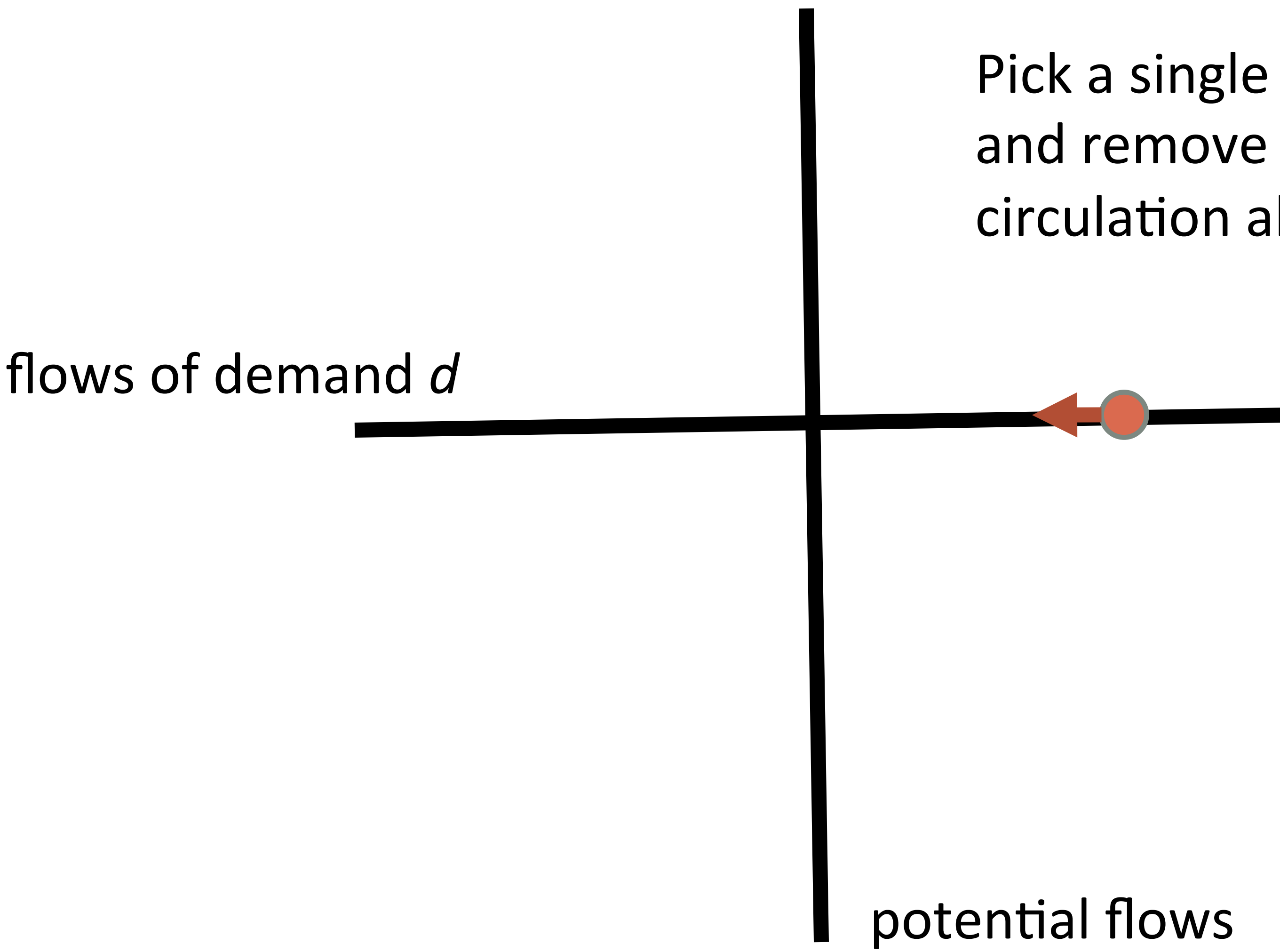
flows of demand *d*



potential flows

# PRIMAL APPROACH: SINGLE STEP (KOSZ '13)

Pick a single cycle and remove the circulation along it



# POTENTIAL BASED SOLVERS [SPIELMAN-TENG`04] [KOUTIS-M-PENG`10, `11]

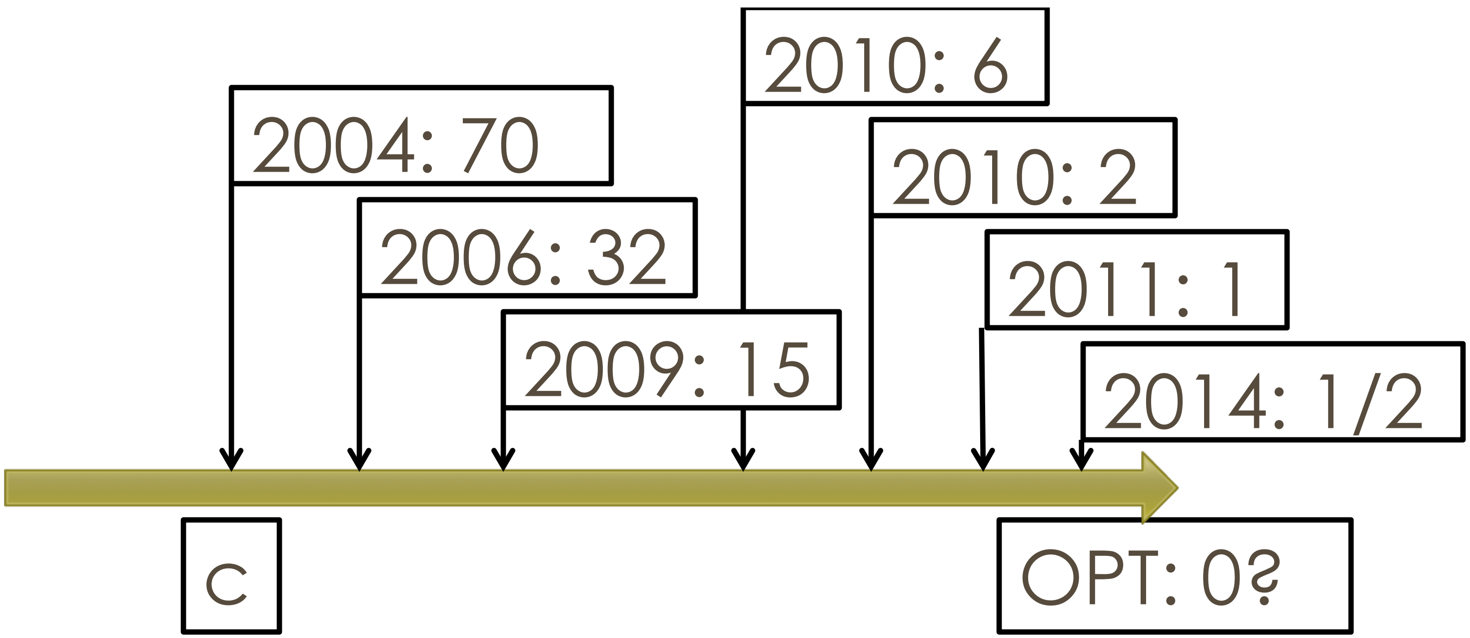
**Input**: n by n SDD matrix A with m non-zeros
vector b
**Output**: Approximate solution Ax = b
**Runtime**: O(m log n )

[Blelloch-Gupta-Koutis-M-Peng-Tangwongsan. `11]:
Parallel solver, $O(m^{1/3})$ depth and nearly-linear work

# ZENO'S DICHOTOMY PARADOX

$O(m\log^{\boxed{c}} n)$

Fundamental theorem of Laplacian solvers: improvements decrease c by factor between [2,3]



# FLOW BASED SOLVERS [KELNER-ORECCHIA-SIDFORD-ZHU `13] [LEE-SIDFORD `13]

**Input**: n by n SDD matrix A with m non-zeros, demand b
**Output**: Approximate minimum energy electrical flow
**Runtime**: $O(m \log^{1.5} n)$

# POTENTIAL BASED SOLVER AND ENERGY MINIMIZATION

- Suppose that A is SPD:

  Claim: minimizing $\frac{1}{2} x^T A x - x^T b$

  gives solution to $Ax = b$.

  Note: Gradient = $Ax-b$

Thus solving these systems are quadratic minimization problems!

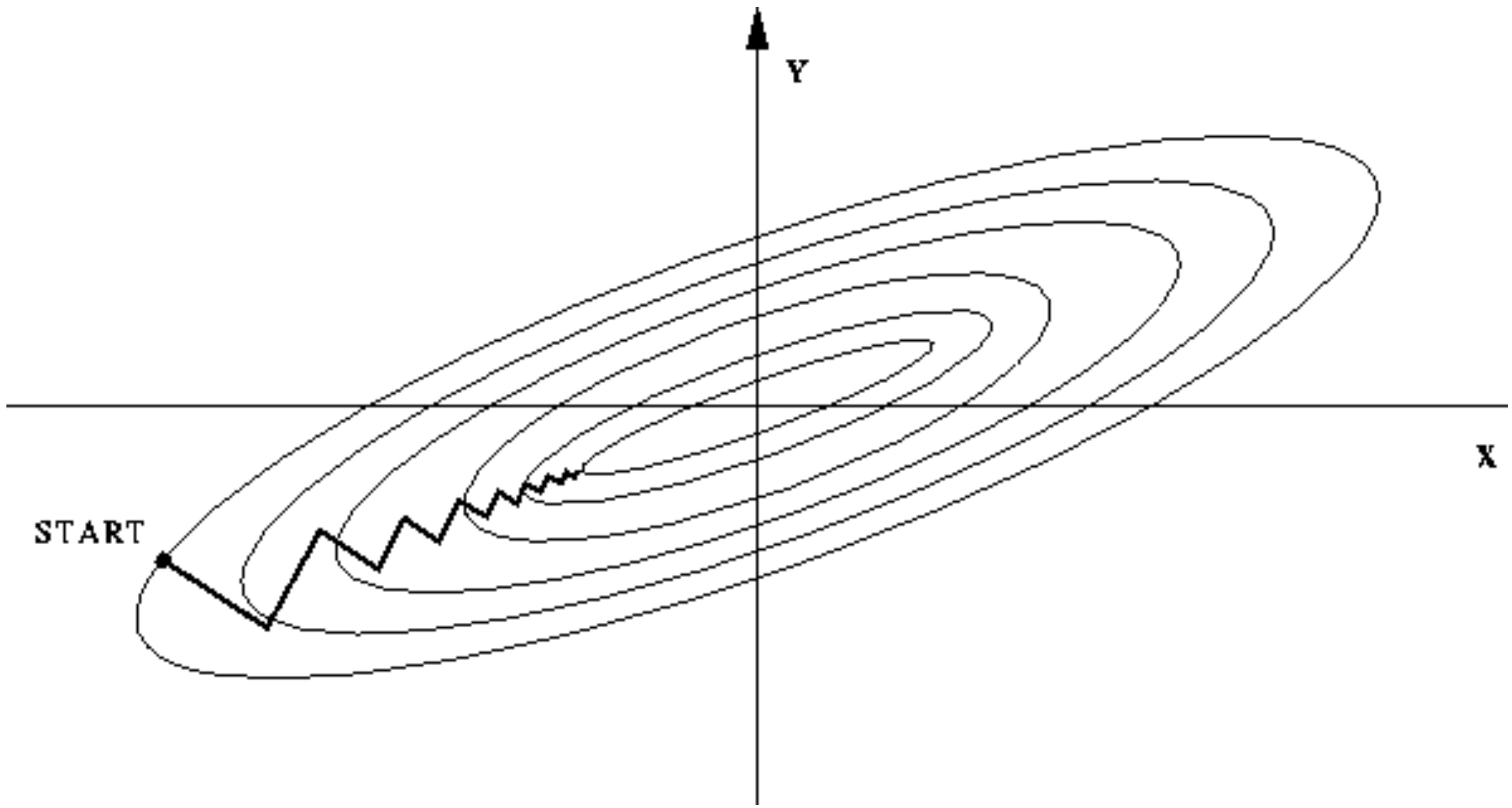
# ITERATIVE METHOD
# GRADIENT DESCENT

- Goal: approx solution to Ax = b
- Start with initial guess $u^0 = 0$
- Compute new guess

$$u^{(i+1)} = u^{(i)} + (b - Au^{(i)})$$

This maybe slow to converge or not converge at all!

# STEEPEST DESCENT



# PRECONDITIONED ITERATIVE METHOD

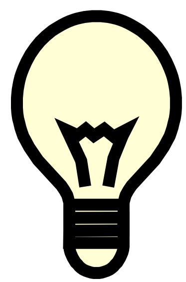
- Goal: approx solution to $B^{-1}Ax = B^{-1}b$
- Start with initial guess $u^0 = 0$
- Compute new guess

$$u^{(i+1)} = u^{(i)} - B^{-1}(b - Au^{(i)})$$

Recursive solve Bz=y where y=($b$-A$u^{(i)}$).

# PRECONDITIONING WITH A GRAPH

[Vaidya `91]: Since A is a graph, B should be as well.
Apply graph theoretic techniques!



And use Chebyshev acceleration

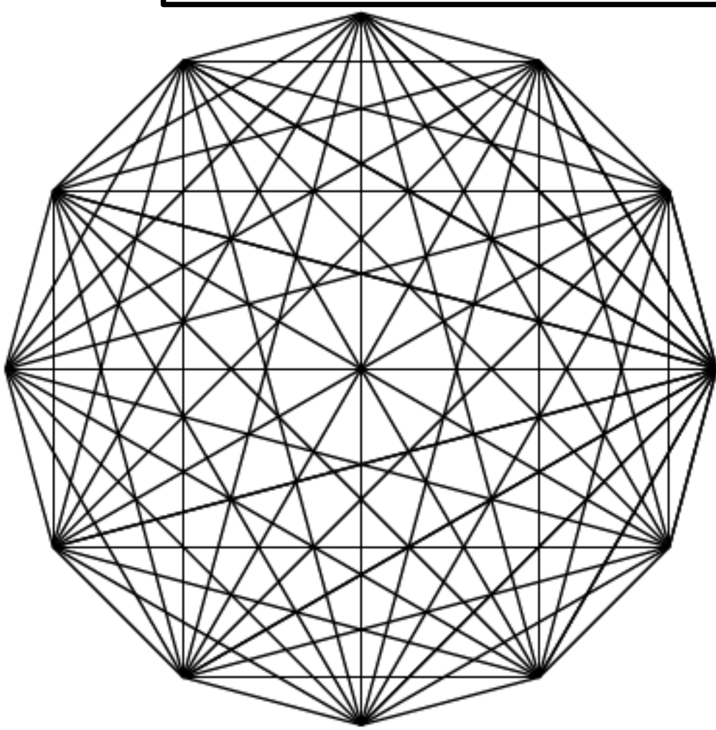
# GRAPH SPARSIFIERS

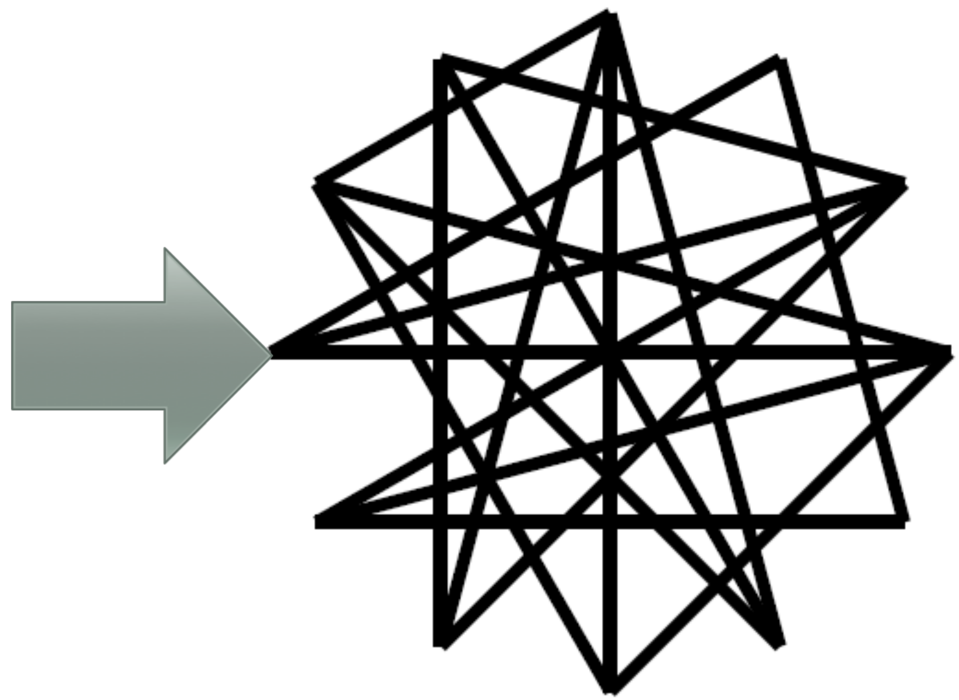
Sparse Equivalents of Dense Graphs that preserve some property

- Spanners: distance, diameter.
- [Benczur-Karger '96] Cut sparsifier: weight of all cuts.
- Spectral sparsifiers: eigenstructure

# EXAMPLE: COMPLETE GRAPH

O(n logn) sampling edges uniform suffice!





# SPECTRAL SPARSIFICATION BY EFFECTIVE RESISTANCE

What probability P(e) should we sample edge?

Answer: [Spielman-Srivastava `08]:

- Set P(e) = R(u,v) effective resistance from u to v.
- For each sample set edge set weight to 1/P(e)
- Sample O(n log n) times.

spectral sparsifier with O(nlogn) edges for any graph

Fact: $\sum_e R(e) = n-1$

# THE CHICKEN AND EGG PROBLEM

How To Calculate Sample Probablity?

[Spielman-Srivastava `08]: Use Solver

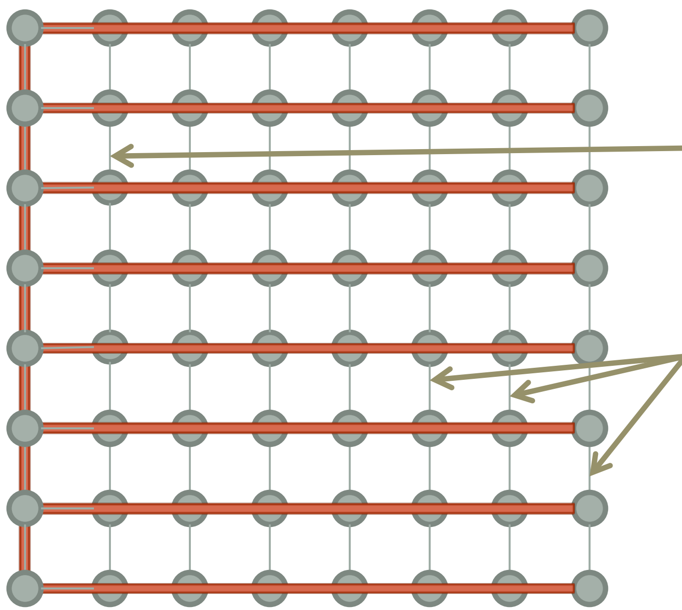
[Spielman-Teng `04]: Need Sparsifier

Workaround: upper bound using Low Stretch Spanning Trees

# CHOICE OF TREES MATTER

$n^{1/2}$-by-$n^{1/2}$ unit weighted mesh

'haircomb' tree is both shortest path tree and max weight spanning tree



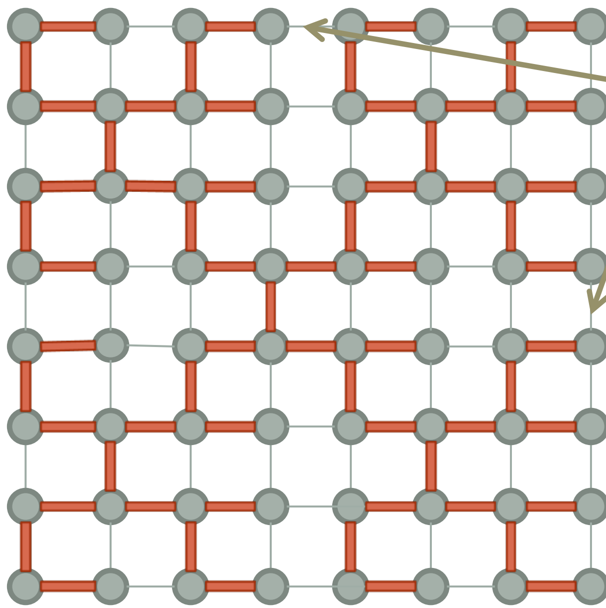
stretch(e)= O(1) 🙂

stretch(e)=O( $n^{1/2}$) ☹

total stretch = O($n^{3/2}$) ☹

# AN O(N LOG N) STRETCH TREE

Recursive 'C' Construction



stretch(e) still $O(n^{1/2})$ ☹

But only $O(n^{1/2})$ such edges ☺

logn levels, total = O(nlogn) ☺

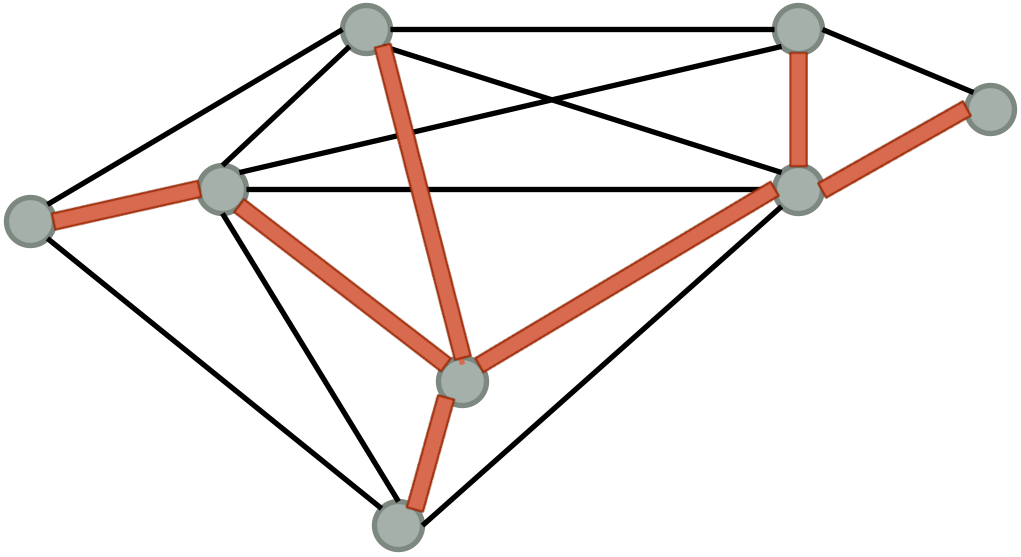
Able to obtain good trees for any graph by leveraging this type of tradeoffs

# LOW STRETCH SPANNING TREES

| |
|---|
| [Alon-Karp-Peleg-West '91]: |
| [Elkin-Emek-Spielman-Teng '05]:<br>A low stretch spanning tree with |
| [Abraham-Bartal-Neiman '08,<br>Koutis-M-Peng `11,<br>Abraham-Neiman `12]:<br>A spanning tree with<br>total stretch O(m log n) in<br>O(m log n) time. |

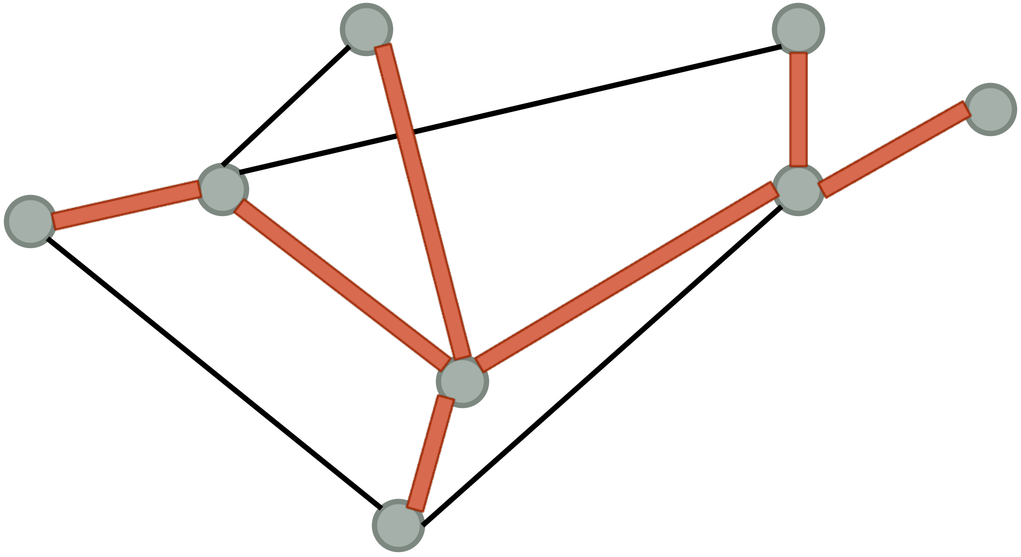
# SOLVER IN ACTION

Sample off tree edges where
P(e) = stretch of e.



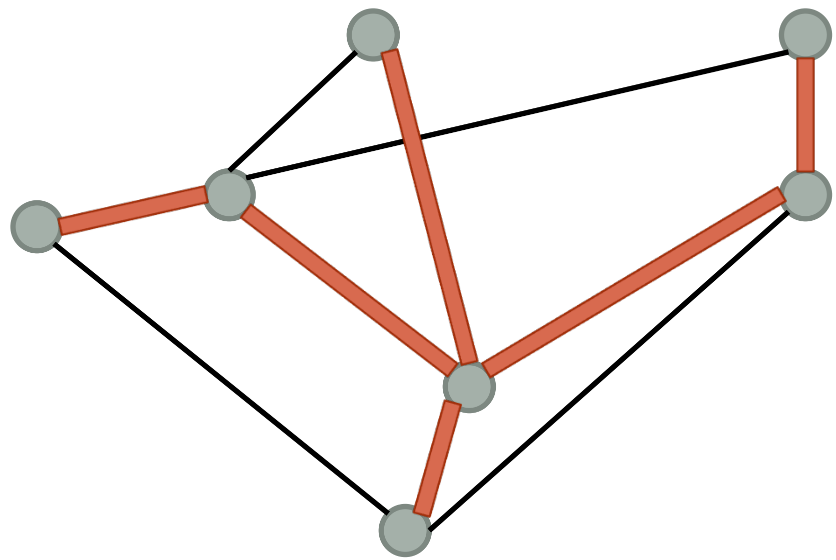
# SOLVER IN ACTION

Eliminate degree 1 or 2 nodes



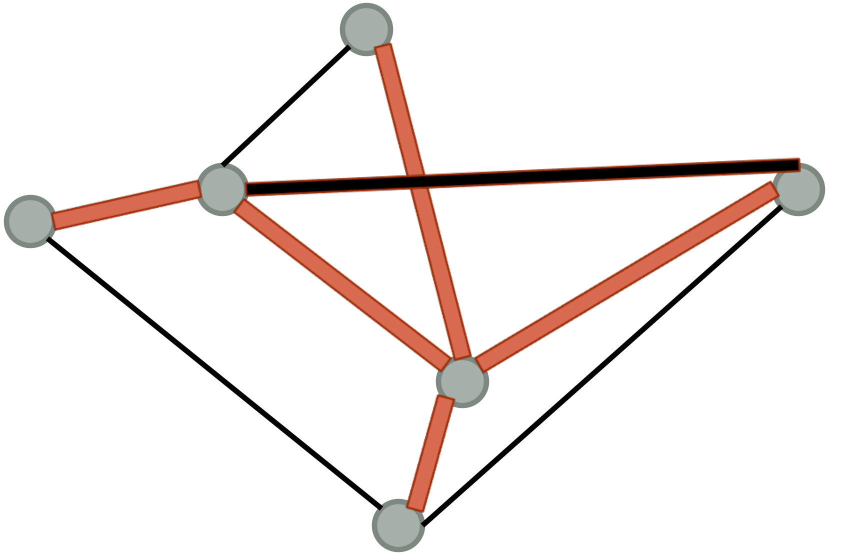
# SOLVER IN ACTION

Eliminate degree 1 or 2 nodes



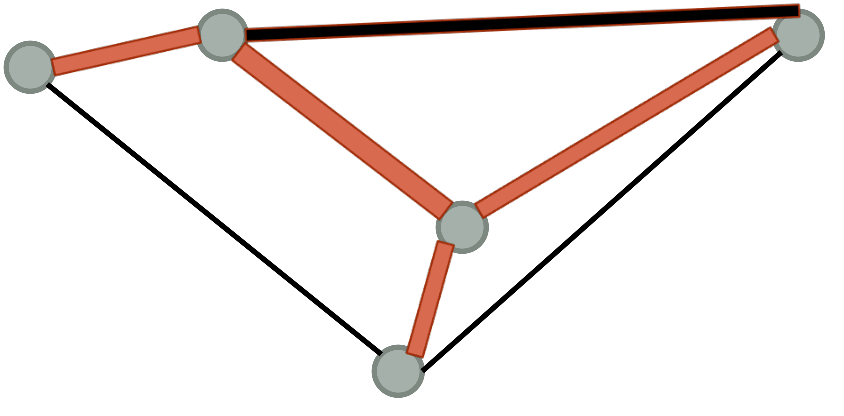
# SOLVER IN ACTION

Eliminate degree 1 or 2 nodes



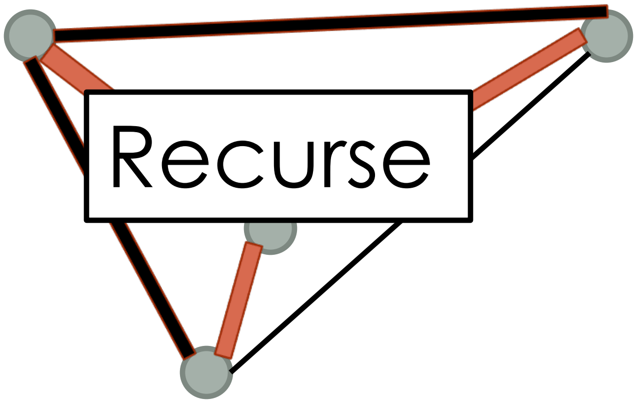
# SOLVER IN ACTION

Eliminate degree 1 or 2 nodes

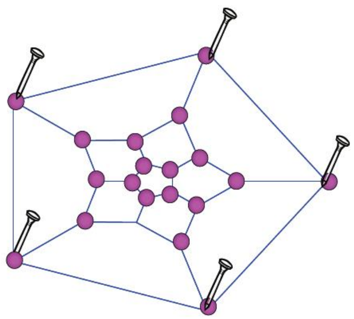


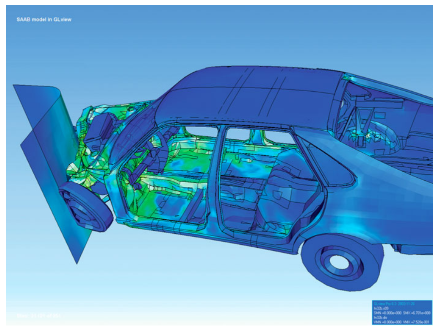
# SOLVER IN ACTION

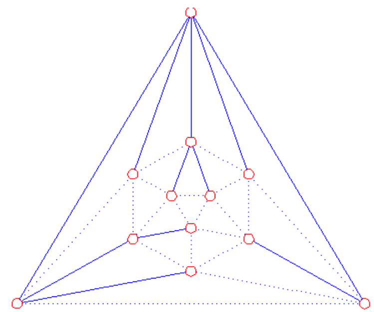
Eliminate degree 1 or 2 nodes

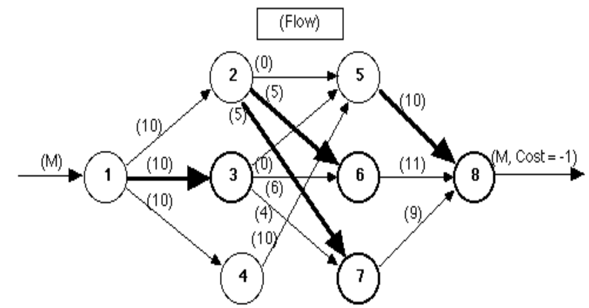


# THEORETICAL APPLICATIONS OF SDD SOLVERS: MULTIPLE ITERATIONS



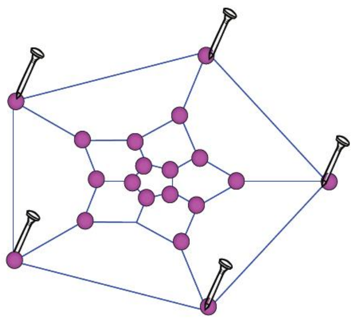


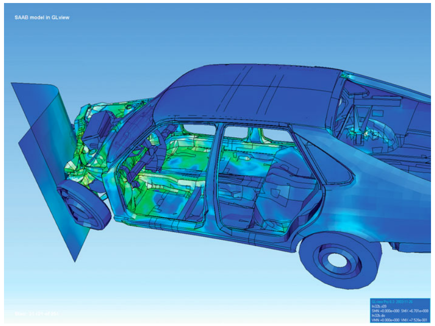


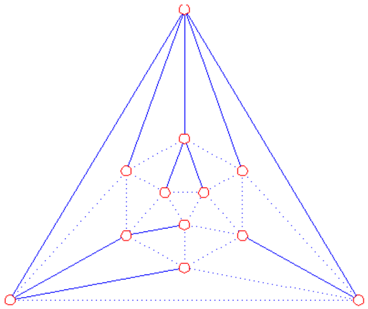


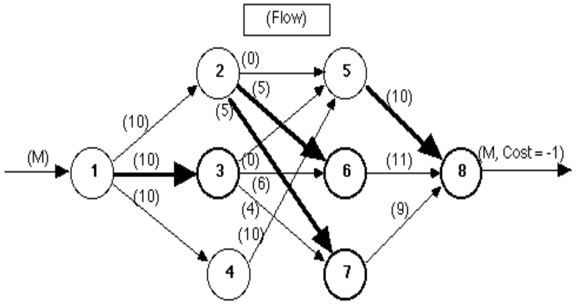

[Tutte `62] Planar graph embeddings.
[Boman-Hendrickson-Vavasis `04] Finite Element PDEs
[Zhu-Ghahramani-Lafferty, Zhou-Huang-Scholkopf `03,05] learning on graphical models.
[Kelner-Mądry `09] Generating random spanning trees in $O(mn^{1/2})$ time by speeding up random walks.

# THEORETICAL APPLICATIONS OF SDD SOLVERS: MULTIPLE ITERATIONS










[Daitsch-Spielman `08] Directed maximum flow, Min-cost-max-flow, lossy flow all can be solved via LP interior point where pivots are SDD systems in $O(m^{3/2})$ time.

# BACK TO IMAGE DENOISING

# PROBLEM WITH QUADRATIC OBJECTIVE

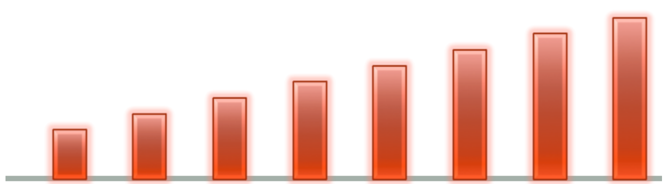
- Result too ‘smooth’, objects become blurred
- Quadratic functions favor the removal of boundaries

# FUNCTION ACCENTUATING BOUNDARIES

$L_1$ smoothness term

If $a<b<c$, $|a-b|+|b-c|$ doesn't depend on $b$

s: sharp boundary



Same smoothness term

Better fidelity

# TOTAL VARIATION OBJECTIVE

- [Rudin-Osher-Fatemi, 92] Total Variation objective: $L_2^2$ fidelity term, $L_1$ smoothness.

Fidelity(x, s) + Smoothness(x)

$(x_i - s_i)^2$ summed over all pixels i.

$|x_i - x_j|$ summed over all neighbor pixels i, j.

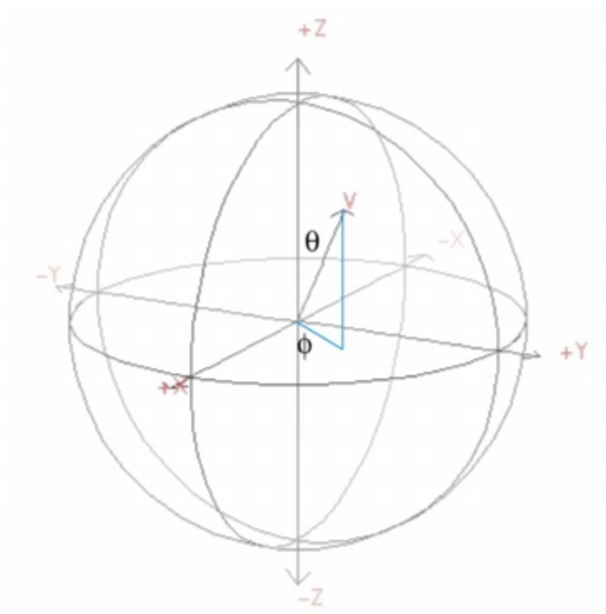
# TOTAL VARIATION MINIMIZATION

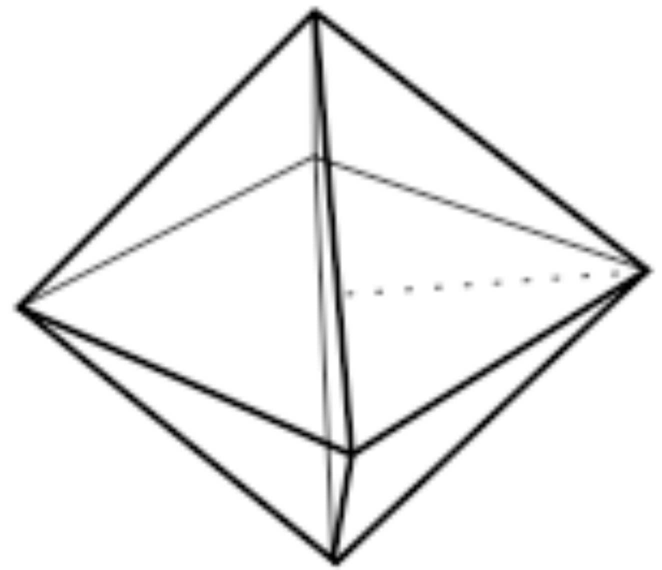
Higher weight on smoothness term

Effect: sharpen boundaries

Overdoing makes image cartoon like

# WHAT'S HARD ABOUT $L_1$?



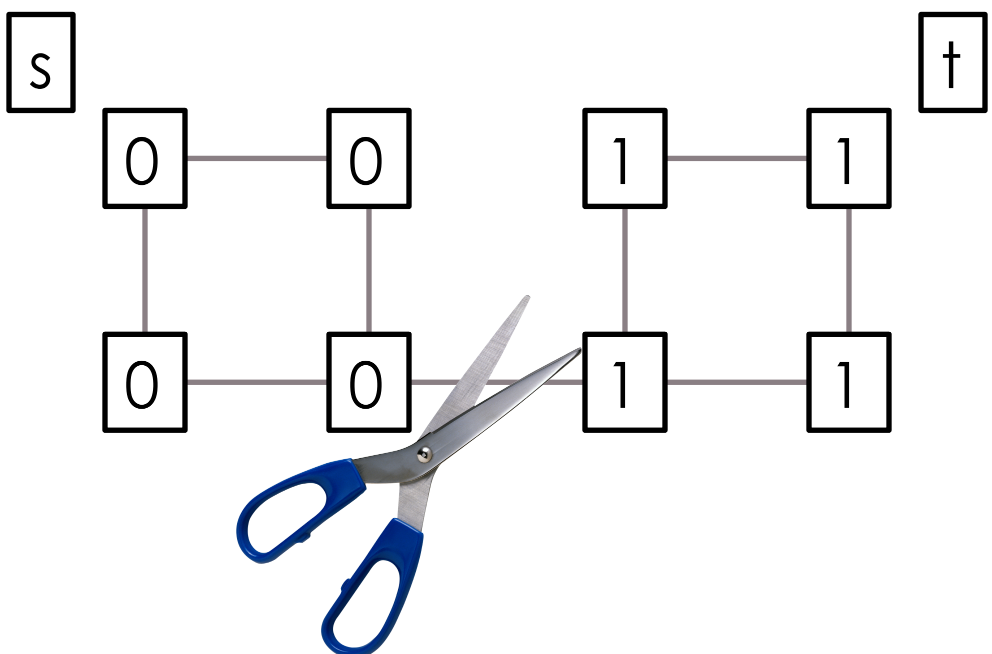
Absolute value function on n variables has $2^n$ points of discontinuity, $L_2^2$ has none.

# MIN CUT PROBLEM AS $L_1$ MINIMIZATION

Minimum s-t cut:

minimize $\Sigma \, |x_i - x_j|$

subject $x_s=0$, $x_t=1$



# MINCUT VIA. $L_2$ MINIMIZATION

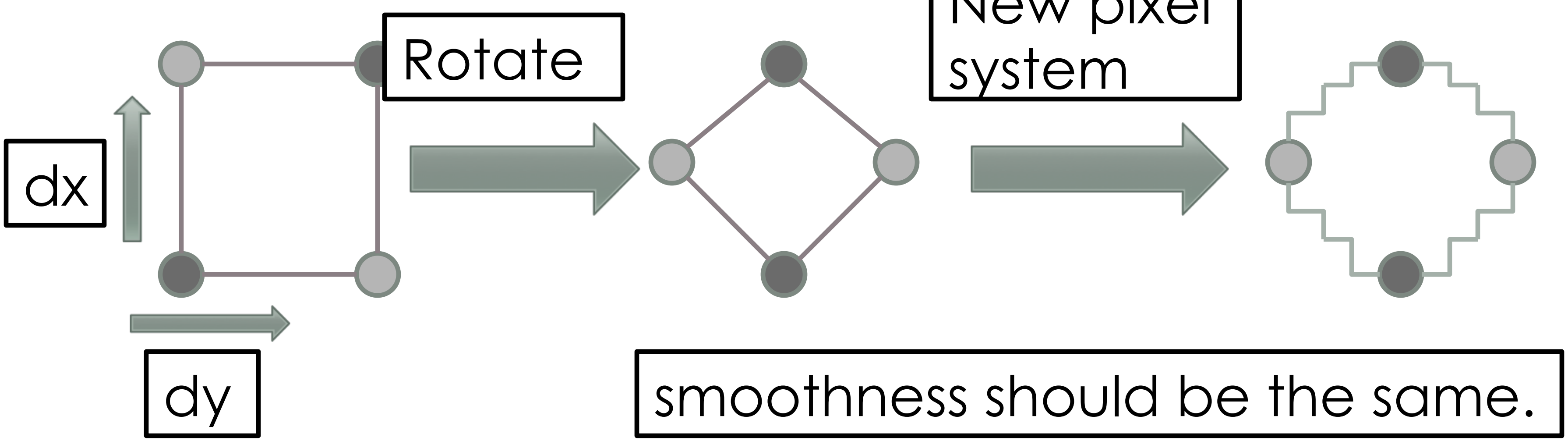
[Christiano-Kelner-Mądry-Spielman-Teng `11]: undirected max flow and mincut can be approximated using $\tilde{O}(m^{1/3})$ SDD solves.

- Multiplicative weights update method
- Gradually update the linear systems being solved

Total: $\tilde{O}(m^{4/3})$

# ISOTROPIC VERSION

[Osher]: 2D images can be rotated, instead of |dx| + |dy|, smoothness term should be $\sqrt{(dx^2 + dy^2)}$



# ALTERNATE VIEW

Alternate interpretation of absolute value: $|t| = \sqrt{t^2}$

$|dx| = \sqrt{(dx^2)}$

Each group can be viewed as an 'edge' in minimum cut problem

This took one year

# TV USING $L_2$ MINIMIZATION

- [Chin-Madry-M-Peng `12]: Can use methods based on electrical flows to obtain $(1+\varepsilon)$ approximation in $O(mk^{1/3}\varepsilon^{-8/3})$ time.
- Interpolates between various versions involving $L_1$ and $L_2$ objectives.

# WHAT IS NEW FOR 2013 AND 2014!

- Faster approximate Flow algorithms!
- Faster solvers!
- Faster exact flow algorithms!
- Faster LSST algorithms
- Parallel LSSTs and solvers

# GENERALIZED GRADIENT DECENT

- [Lee-Rao-Srivastava`13]: gradient descent view of maxflow using electrical flows
- Generalized  Decent for Classes of Lipschitz convex functions.
  - Nesterov optimization
  - Soft max of $L_\infty$
  - Smoothed version of $L_1$

# FASTER APPROXIMATE FLOW ALGORITHMS!

Sherman 13: Approximate maxflow and Mini-cut in $\tilde{O}(m^{1+\delta} poly(k, \varepsilon^{-1}))$ time

Uses:

Madry 10: Fast Approximate Mini-Cut Algorithm

# FASTER APPROXIMATE FLOW ALGORITHMS!

Kelner-Lee-Orecchi-Sidford’ 13:
Approximate maxflow and multi-commodity flow in $\tilde{O}(m^{1+\delta}\mathrm{poly}(k, \varepsilon^{-1}))$ time

Uses:

Racke’ 08: Approximate Oblivious routing ideas.

# EVEN FASTER SOLVERS

- Cohen-Kyng-Pachocki-Peng-Rao `13
  SDD linear systems Faster solver in
  - $O(m\log^{1/2} n)$ time given a LSST.

- The log appears in two places in KMP:

1. Matrix Chernoff Bounds
2. LSST tree construction

# LOW DIAMETER DECOMPOSITION

Awerbuch 85: O(log n) diameter clusters with o(m) inter-cluster edges.

M,Peng,Xu 13: O( c log n) diameter clusters with O(m/c) expected inter-cluster edges:
O(m) work and O(c $\log^2$ n) depth

Algorithm: Do BFS from each noted using an exponential delay start time.

# FASTER TREE GENERATION

- Koutis-M-Peng `11, Abraham-Neiman `12]:

LSST with stretch O(m log n) in O(m log n) time.

We do not know how to beat these bounds!

We find a tree that is good enough!

# LSST RELAXATION

Allow Steiner nodes.
But still embeddable!

Relax the definition of stretch.

Recall $STR_T(e=(a,b)) = dist_T(a,b)$
New Def: $STR_T^P(e=(a,b)) = |dist_T(a,b)|^P$
for $P < 1$

# FASTER TREE ALGORITHM FOR $L^P$- STRETCH

| | Runtime | Stretch |
|---|---|---|
| **AKPW** | $O(m \log \log n)$ | $O(\log^{O(1)} n)$ |
| **Bartal/AN** | $O(m \log n)$ | $O(\log n)$ |

Can we get the best of both worlds?

# NEARLY LINEAR TIME, POLYLOG DEPTH SOLVERS

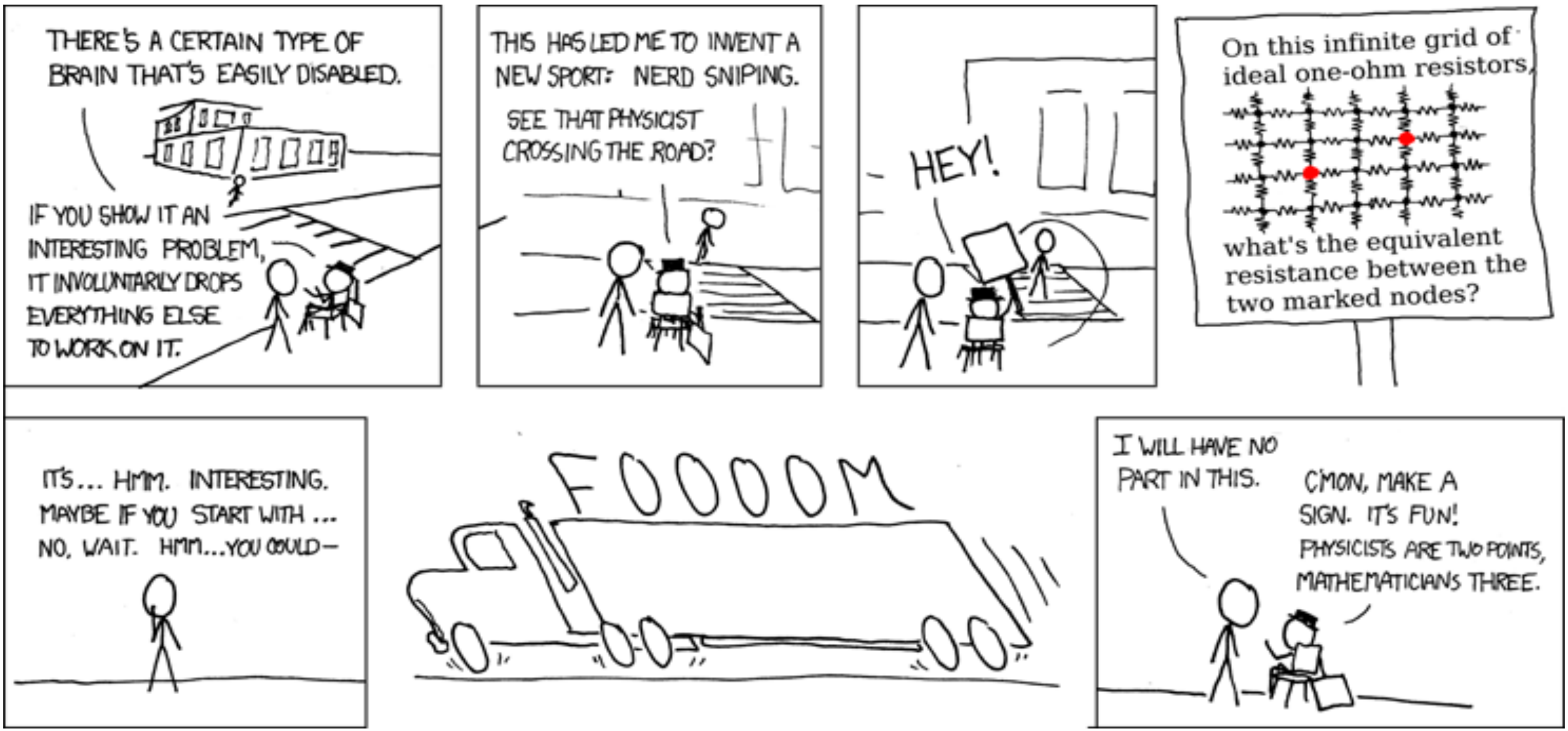
[Peng-Spielman `13]

**Input**: SDD matrix $\mathbf{M}$ with m non-zeros, condition number $\kappa$
**Output**: Sparse product $\mathbf{Z_1 \ldots Z_k} \approx_{\varepsilon} \mathbf{M}^{-1}$
**Cost**: $O(\log^c m \log^c \kappa \log(1/\varepsilon))$ time
$O(m \log^c m \log^c \kappa \log(1/\varepsilon))$ work

Approximation $\approx_{\varepsilon}$ in matrix sense

# FUTURE WORK

- Practical/parallel implementations?
  - The win over sequential is parallel!
- Near linear time exact max flow?
  - $\log(1/\varepsilon)$ dependency in runtime?
- Sub-quadratic SPD solver?



THERE'S A CERTAIN TYPE OF BRAIN THAT'S EASILY DISABLED.
IF YOU SHOW IT AN INTERESTING PROBLEM, IT INVOLUNTARILY DROPS EVERYTHING ELSE TO WORK ON IT.
THIS HAS LED ME TO INVENT A NEW SPORT: NERD SNIPING.
SEE THAT PHYSICIST CROSSING THE ROAD?
HEY!
On this infinite grid of ideal one-ohm resistors, what's the equivalent resistance between the two marked nodes?
IT'S... HMM. INTERESTING. MAYBE IF YOU START WITH ... NO, WAIT. HMM...YOU COULD—
FOOOOM
I WILL HAVE NO PART IN THIS.
C'MON, MAKE A SIGN. IT'S FUN! PHYSICISTS ARE TWO POINTS, MATHEMATICIANS THREE.