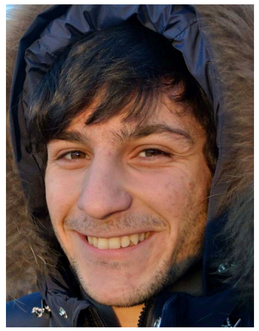
Scalable kernel methods

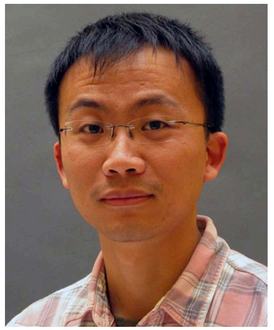
(aka N-body problems, radial basis functions)

with

Dhairya Malhotra, Chenhan Yu, James Levitt, Severin, Reiz Bill March, and Bo Xiao





GEORGE BIROS

padas.ices.utexas.edu

INSTITUTE FOR COMPUTATIONAL ENGINEERING & SCIENCES

THE UNIVERSITY OF TEXAS AT AUSTIN

Outline

- Overview of kernel methods
- High-dimensions
- Extensions to positive definite matrices
- HPC

N-body problem

Input

N points in $\mathbb{R}^d$: $x_1, \ldots, x_N$

N densities in $\mathbb{R}$: $w_1, \ldots, w_N$

Output

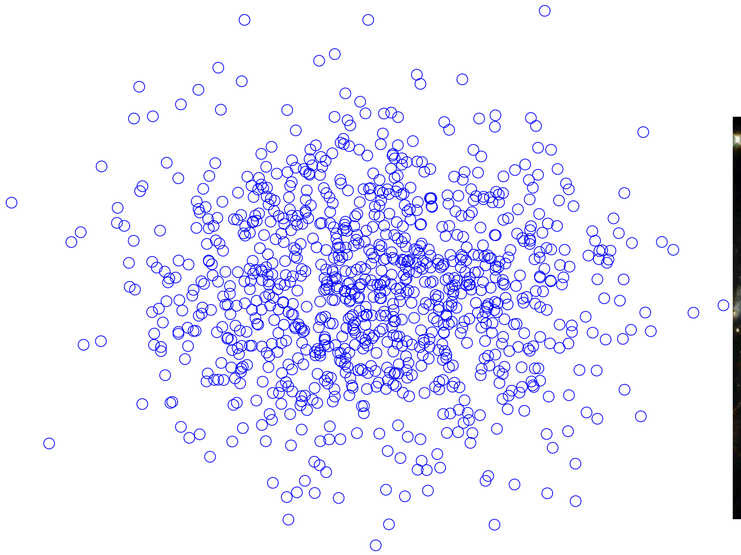
N potentials in $\mathbb{R}$: $u_1, \ldots, u_N$

$$u_i = \sum_{\substack{j=1 \\ j \neq i}}^{N} G(x_i, x_j) w_j$$

$$G(x_i, x_j) = \frac{1}{\|x_i - x_j\|_2}$$

kernel/Gram matrix

kernel function



Gaussian	$\exp(-\|x - x_j\|^2/(2h^2))$
Laplace	$\|x - x_j\|^{2-d},\ d > 2$
Matern	$\left(\sqrt{2\nu}\|x - x_j\|\right)^{\nu} K_{\nu}\left(\sqrt{2\nu}\|x - x_j\|\right)$
Polynomial	$\left(x^T x_j/h + c\right)^p$
Ornstein-Uhlenbeck	$\exp(-c\|x - x_j\|)$
Multiquadratic	$\sqrt{c^2 + \|x - x_j\|_2^2}$
Inverse multiquadratic	$1/\sqrt{c^2 + \|x - x_j\|_2^2}$

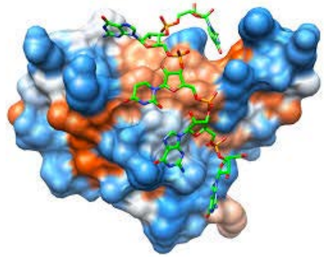
Applications

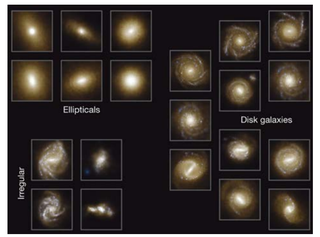
Simulation

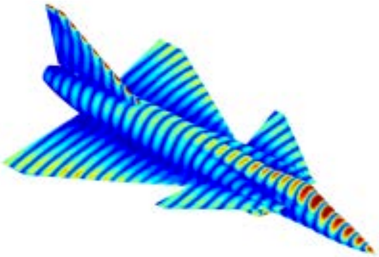
- Gravity & Coulomb
- Waves & scattering
- Fluids & transport

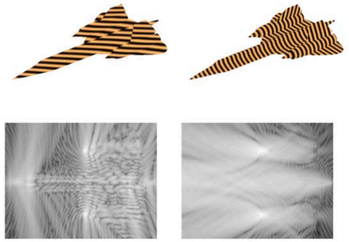
Data analysis

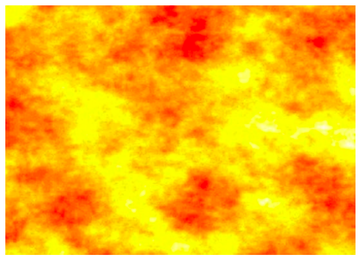
- Kernel methods in machine learning
- Approximation/Geostatistics
- Non-parametric statistics

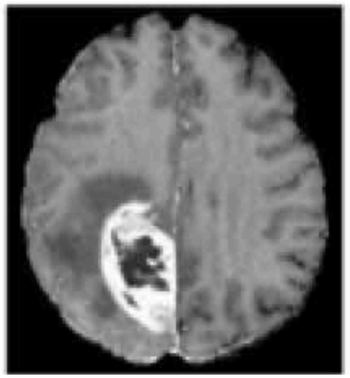


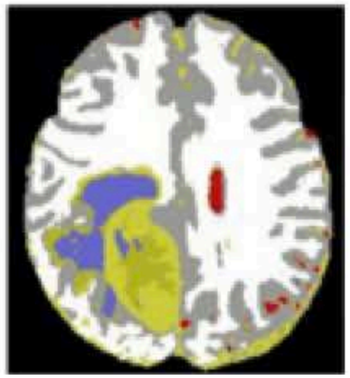












Relation to partial differential equations

classical N-body

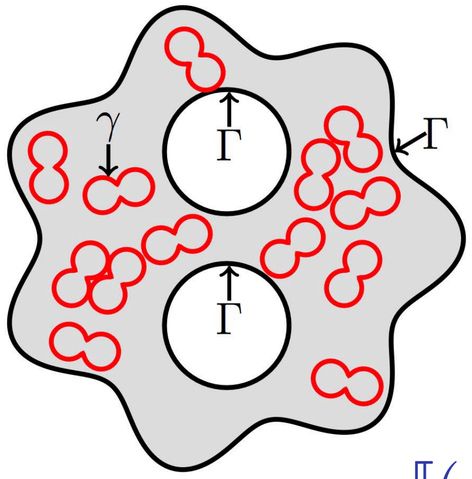
$$-\Delta u(x) = \sum_{i=1}^{N} \delta(x - x_i) f_i$$

$$u(x) = \sum_{i=1}^{N} \frac{1}{\|x - x_i\|} f_i$$

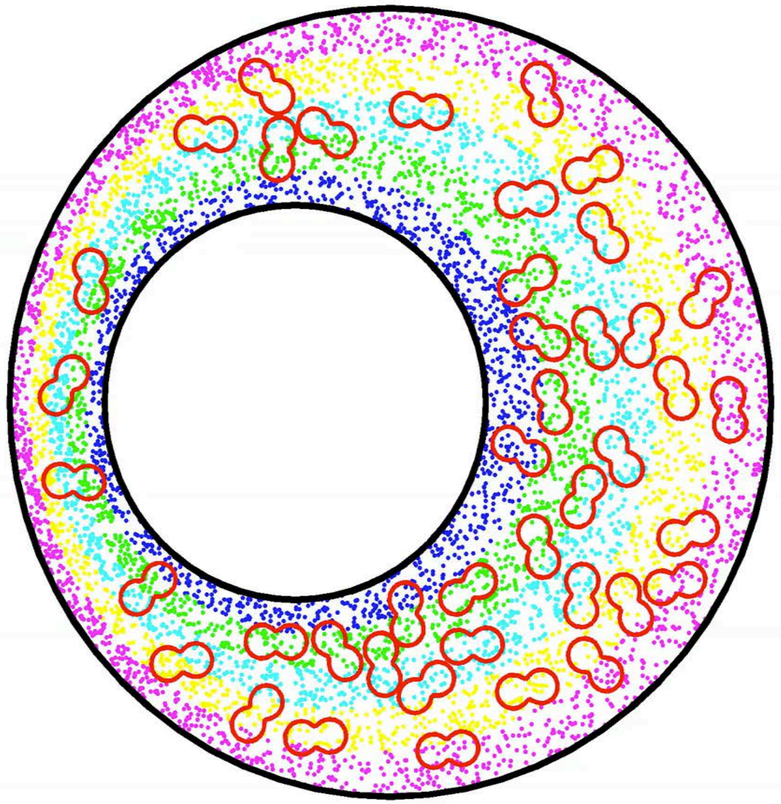
potential theory

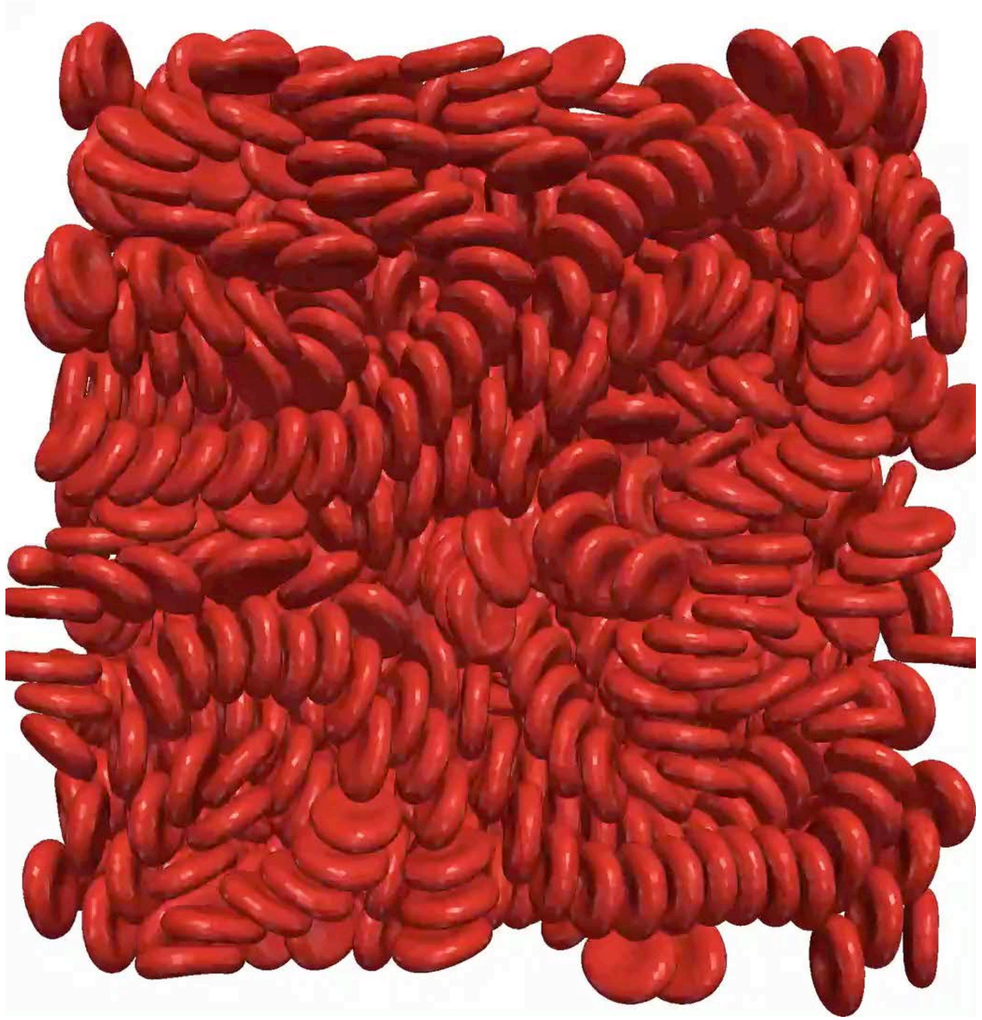
$$-\Delta u(x) = f(x)$$

$$u(x) = \int_{x'} \frac{1}{\|x - x'\|} f(x')$$



$$
\begin{aligned}
-\Delta \mathbf{v} + \nabla p &= 0, \quad && \mathbf{x} \in \Omega \\
\mathrm{div}\mathbf{v} &= 0, && \mathbf{x} \in \Omega \\
\mathbf{v} &= \mathbf{g}, && \mathbf{x} \in \Gamma \\
[\![\mathbf{v}]\!] &= 0, && \mathbf{x} \in \gamma \\
[\![(-pI + \nabla \mathbf{v} + \nabla \mathbf{v}^T)\mathbf{n}]\!] &= \mathbf{f}, && \mathbf{x} \in \gamma \\
\frac{d\mathbf{x}}{dt} &= \mathbf{v}, && \in \gamma
\end{aligned}
$$





Kernel density estimation

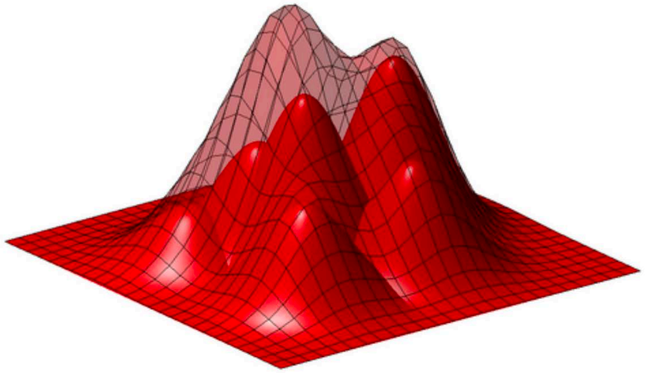
Given $\{x_j\}_{j=1}^N$

$$p_N(x) = \frac{1}{N} \sum_{j=1}^{N} G(x, x_j)$$

Silverman'99, *"Density estimation"*

Radial basis approximation

$$f(x) = \sum_{j=1}^{N} G(x, x_j) w_j$$



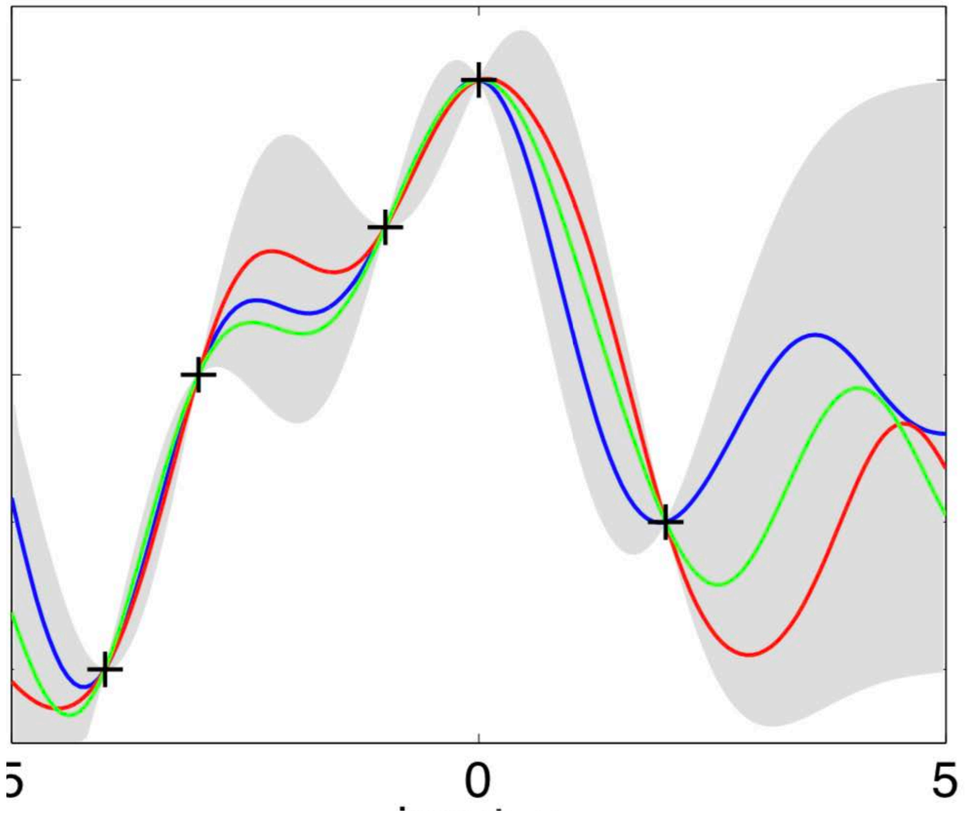
Buhmann'04, Wedland'04, Fornberg & Flyer'12
Koumoutsakos & Cottet'01

Gaussian processes

$$f(x) \sim \mathcal{N}(\mu, C)$$

$$\mu = G(:, x)^T w$$

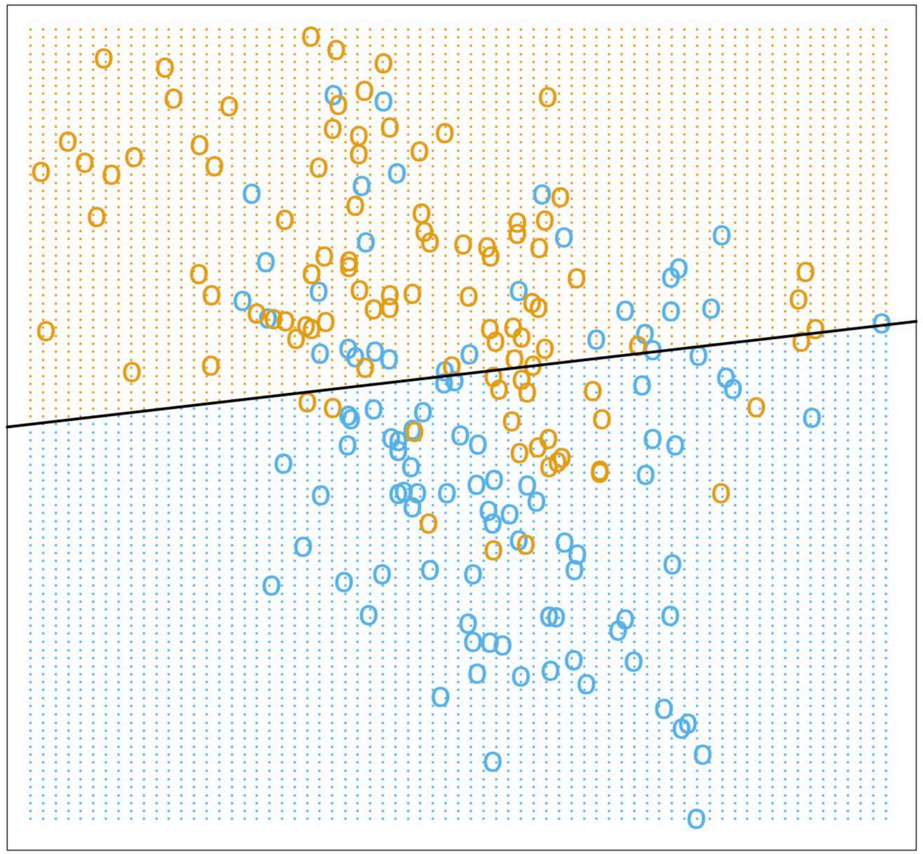
$$C = G(x, x) - G(:, x)^T G(:, :)^{-1} G(:, x)$$

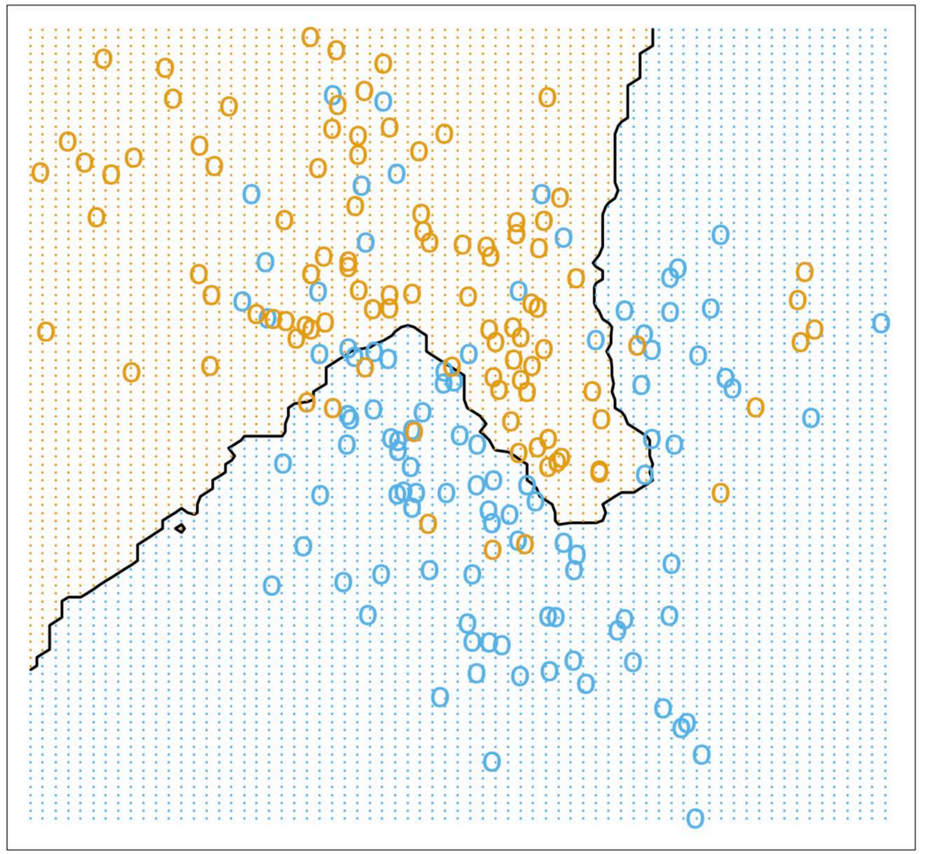


Rasmussen & Williams'06, *"Gaussian processes for machine learning"*

Siggraph 01, Carr, Beatson et al

Logistic regression / SVM





Hastie & Tibshirani & Friedman, "*The elements of Statistical Learning*"

Kernel binary classification

Inference: $c(x) = \text{sign}(x^T w + b)$

$c(x) = \text{sign} \sum_{j=1}^{N} G(x, x_j) w_j$

Training:

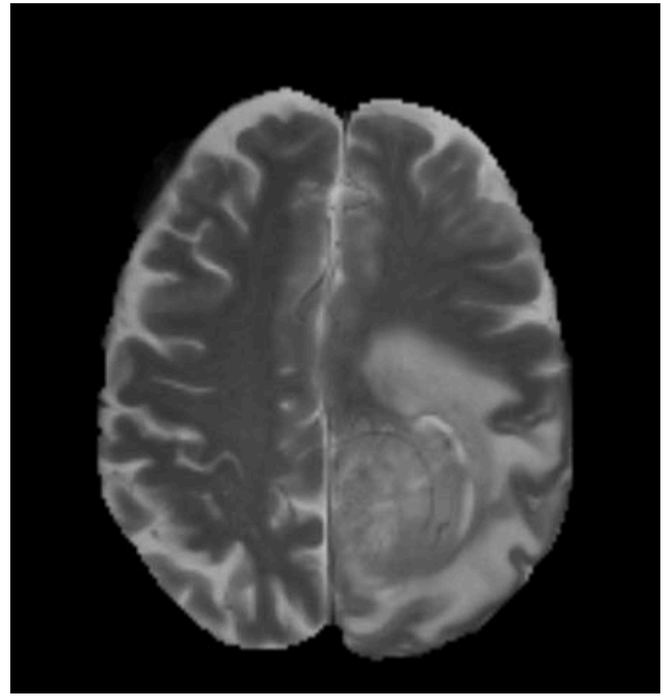
Given $\{x_i \in \mathbb{R}^d, c_i \in \{-1, 1\}\}_{i=1}^N$

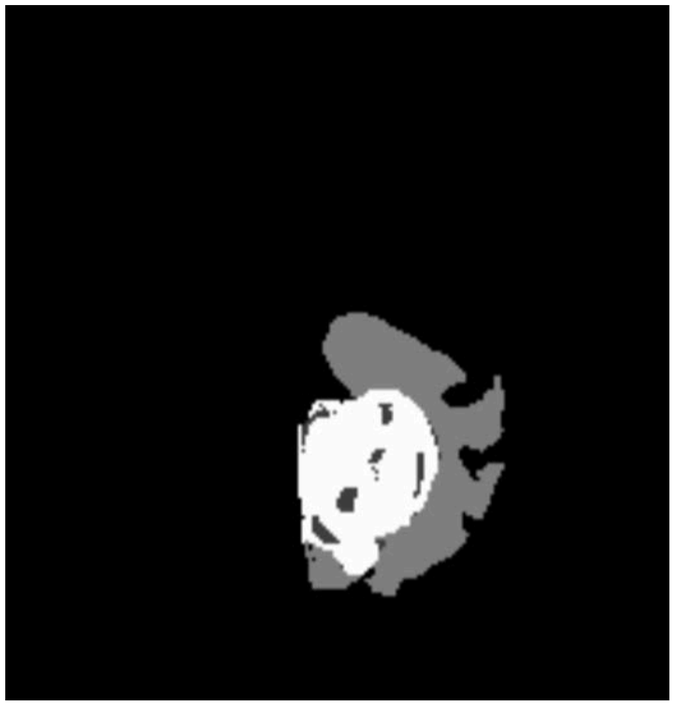
Find $\{w_j\}_{j=1}^N$ such that $\sum_{j=1}^{N} G(x_i, x_j) w_j = c_i, \quad \forall i.$

Logistic regression

$$\log \frac{p(x)}{1-p(x)} = w^T x$$

$$\log \frac{p(x)}{1-p(x)} = \sum_{j=1}^{N} G(x, x_j) w$$



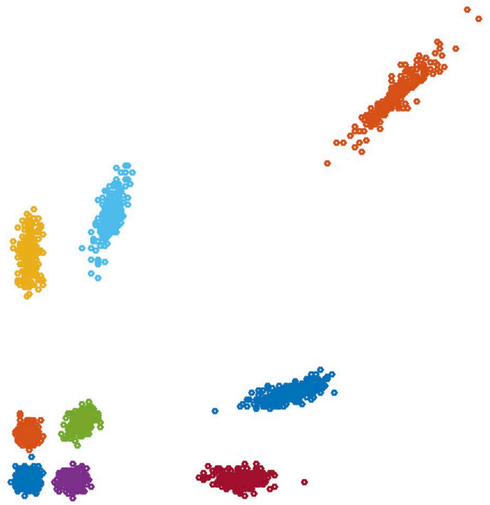


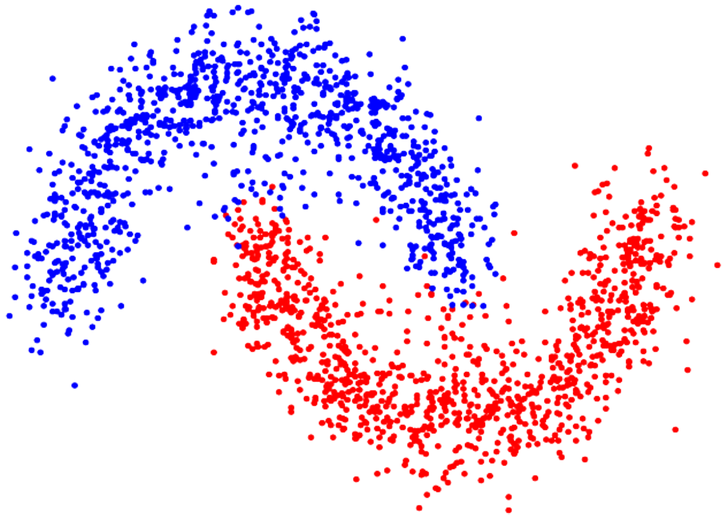
Spectral clustering

V: κ largest eigenvectors of G

Normalize $V(:, 1:\kappa)$

K-means cluster $V(:, 1:\kappa)$





Examples in CSE

- Potential and approximation theory (fractional PDEs)
- Uncertainty quantification with Gaussian processes
- Kernel PCA for model reduction
- Diffusion maps / manifold learning
- Compression of covariance matrices
- Construction of priors in inverse problems
- Identifying patterns in spatial fields (structural defects in materials)
- Chemistry (atomization energies, accelerate MD)

Summary

- Several methods for approximation, supervised and unsupervised learning
- Rich theory related to reproducing kernels, learning theory, approximation theory, convergence with respect the sample size, stability with respect perturbations
- Wasserman’06, Rasmussen’06, Taylor & Cristianini’05 Scholkopf & Smola’02

Major shortcomings

- Choice of the kernel function
- Choice of the distance metric
- Computational complexity

$$G(x_i, x_j) = \exp\left(-\frac{1}{\sigma^2}\|x_i - x_j\|_2\right)$$

Computational challenges

- N points
  - N^2 work for matvec
  - N^3 work for factorization

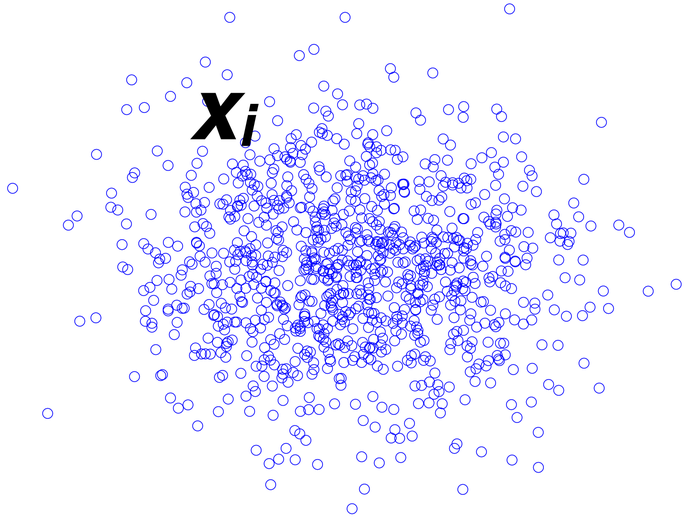
Related work — low dimensions

- Barnes & Hut’86 — treecodes
- Greengard & Rokhlin’87 — FMM
- Rokhlin’90 — high-frequency FMM
- Hackbush & Novak’89 — panel clustering
- Benderdorf’08 & Hackbush’99,’15 — H-matrices
- Greengard & Gropp’91 — parallel shared memory
- Warren & Salmon’93 — parallel distributed memory

Software libraries

- FMM3D - Greengard (NYU)
- ExaFMM - Yokota
- BBFMM - Darve
- FastCap2 - White
- ScalFMM - INRIA
- KIFMM, PVFMM - Ying, B., Malhotra
- ChaNGa, GADGET, LAMMPS and other application-specific packages (Many others based on Ewald sums)

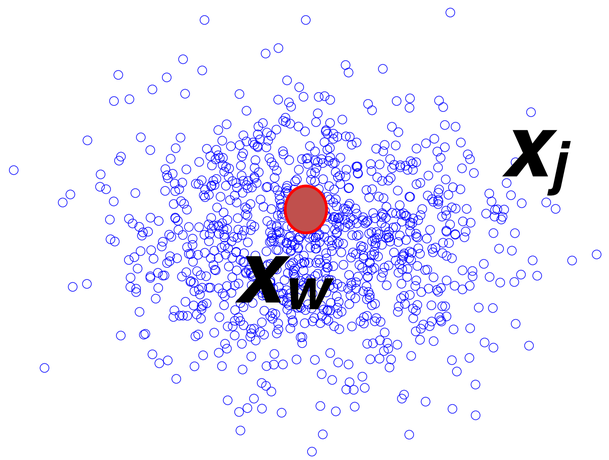
Idea I: far-field —> low rank

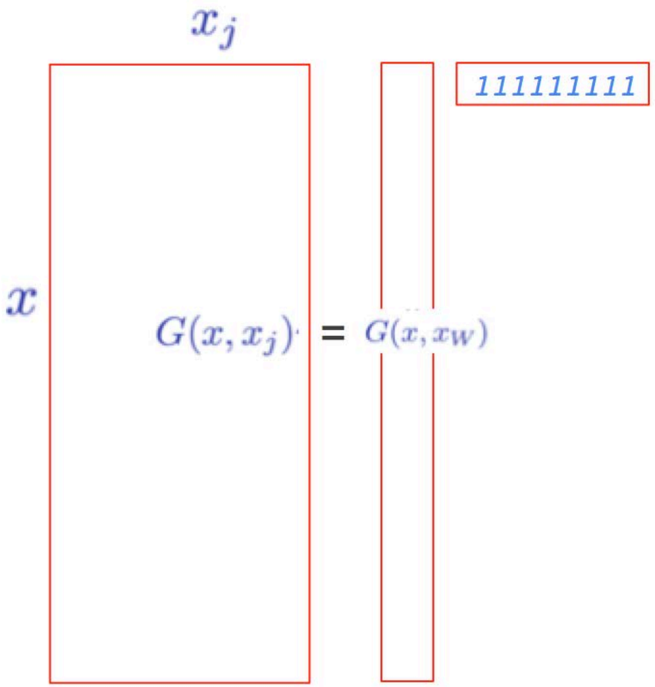


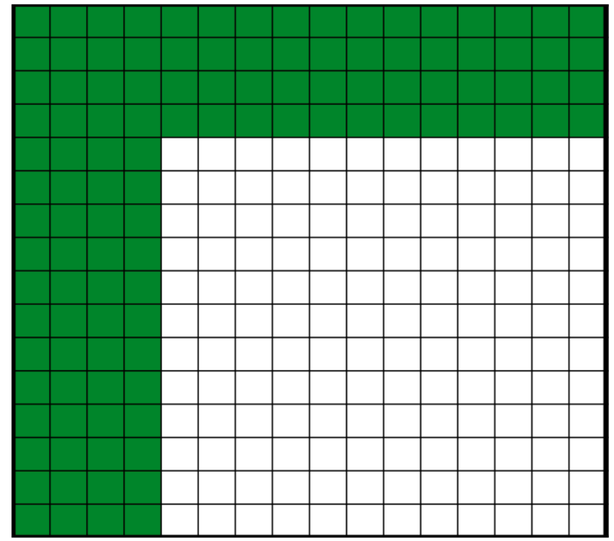
x: Target, x_j: sources
w_j: weights

$$u(x) = \sum_j G(x, x_j) w_j$$

1. compute $W = \sum_i w_j$
2. choose x_W
3. $u(x) \approx G(x, x_W) W$

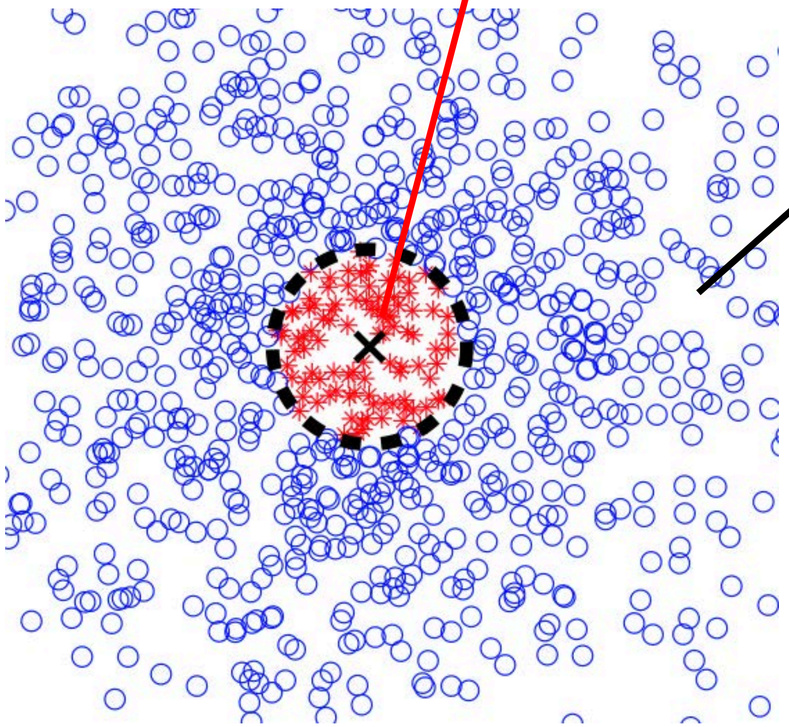




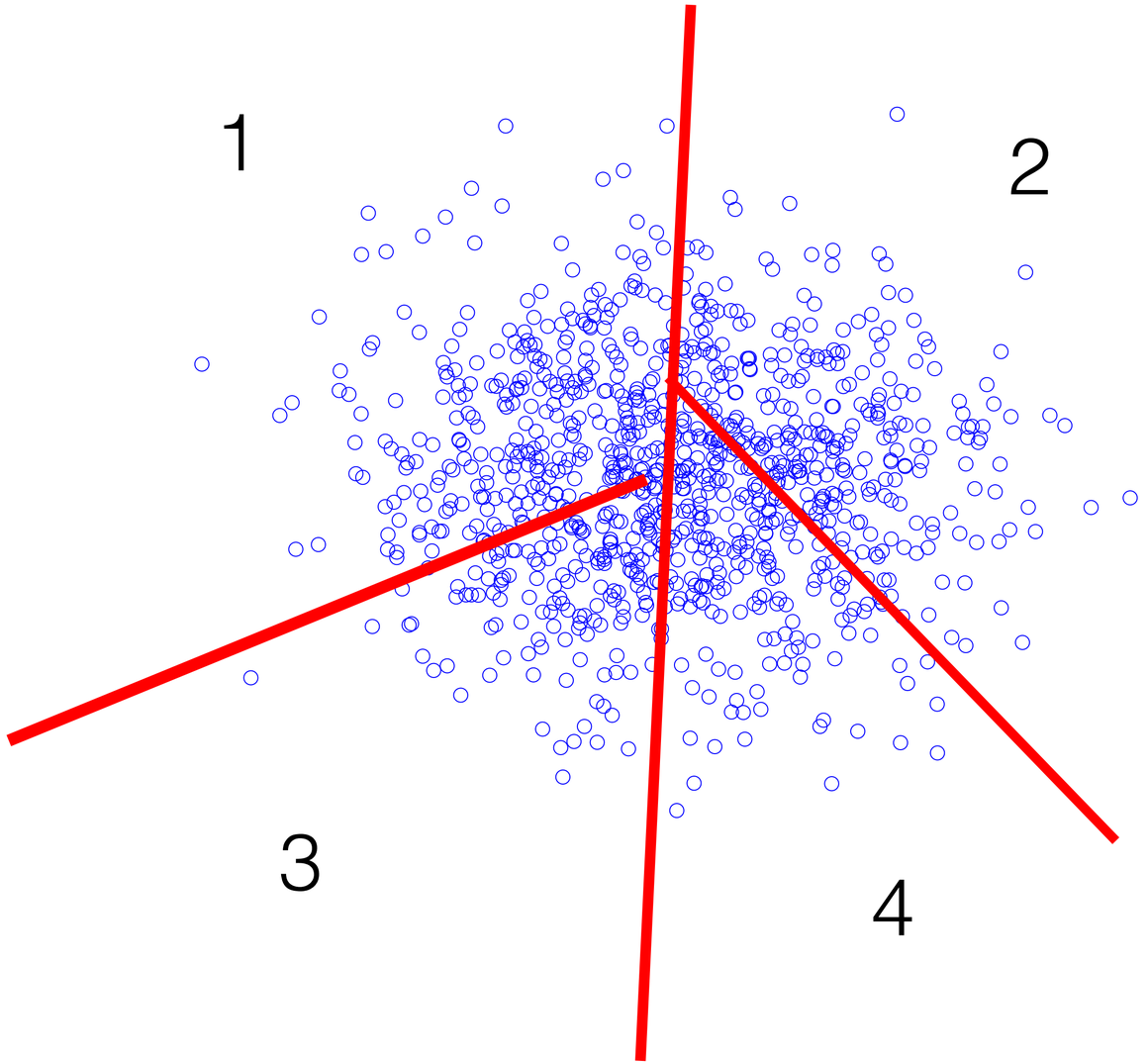


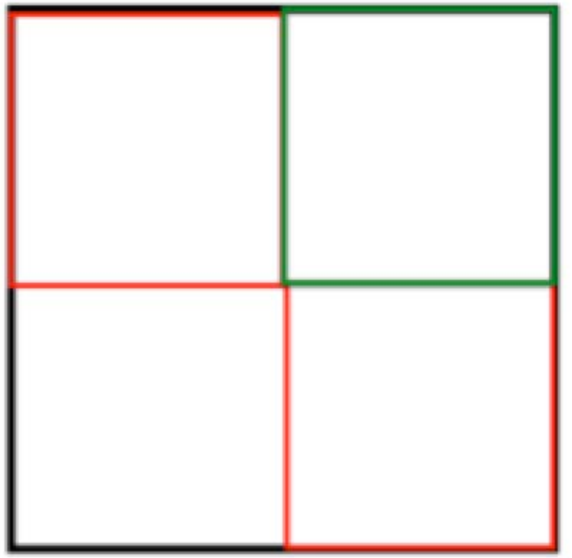
Idea II: Near/Far field split

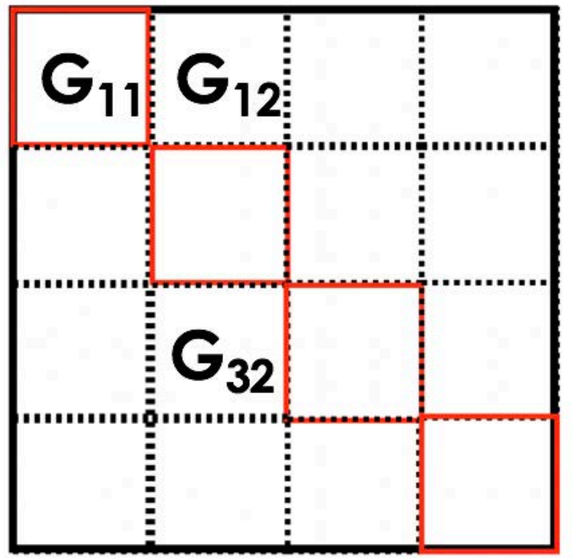
$$u_i = \sum_{\substack{j=1 \\ j \neq i}}^{N} G(x_i, x_j) w_j = \sum_{j \in \text{near}(i)} G_{ij} w_j + \sum_{j \in \text{far}(i)} G_{ij} w_j$$



Idea III: recursion







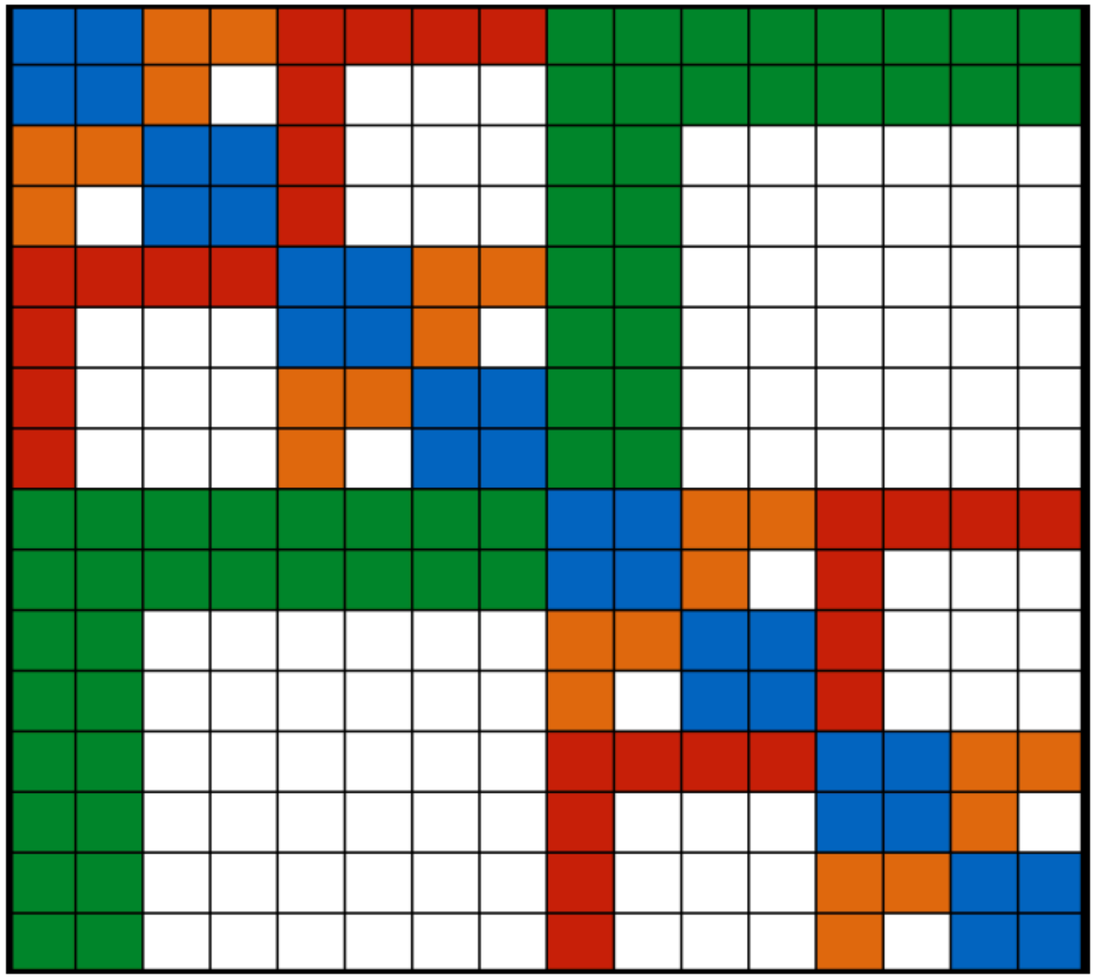
Hierarchical matrices, basic idea

$$\begin{bmatrix} G_{11} & G_{12} \\ G_{21} & G_{22} \end{bmatrix}$$

$$= \begin{bmatrix} G_{11} & 0 \\ 0 & G_{22} \end{bmatrix} + \begin{bmatrix} 0 & G_{12} \\ G_{21} & 0 \end{bmatrix}$$

$$D + UV$$

$$D + UV$$

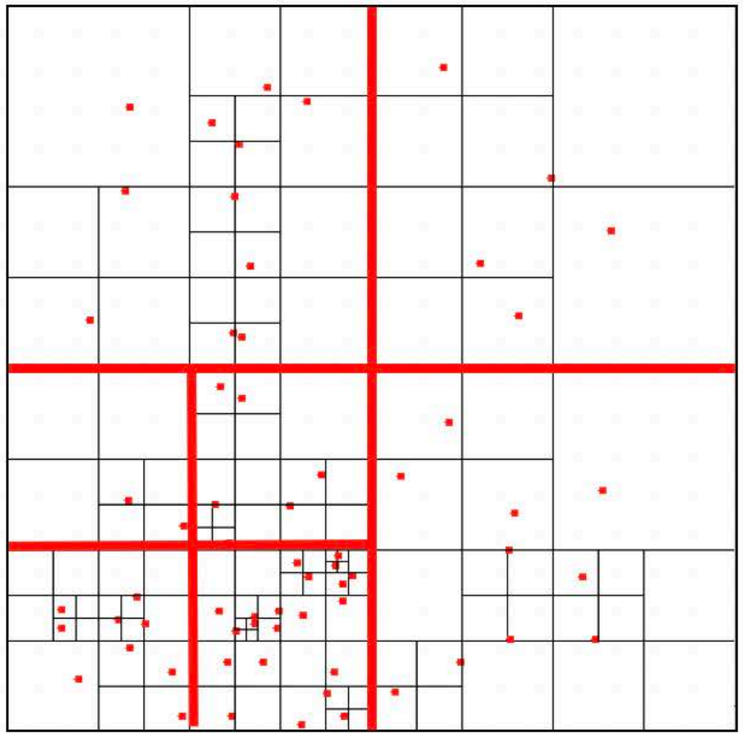


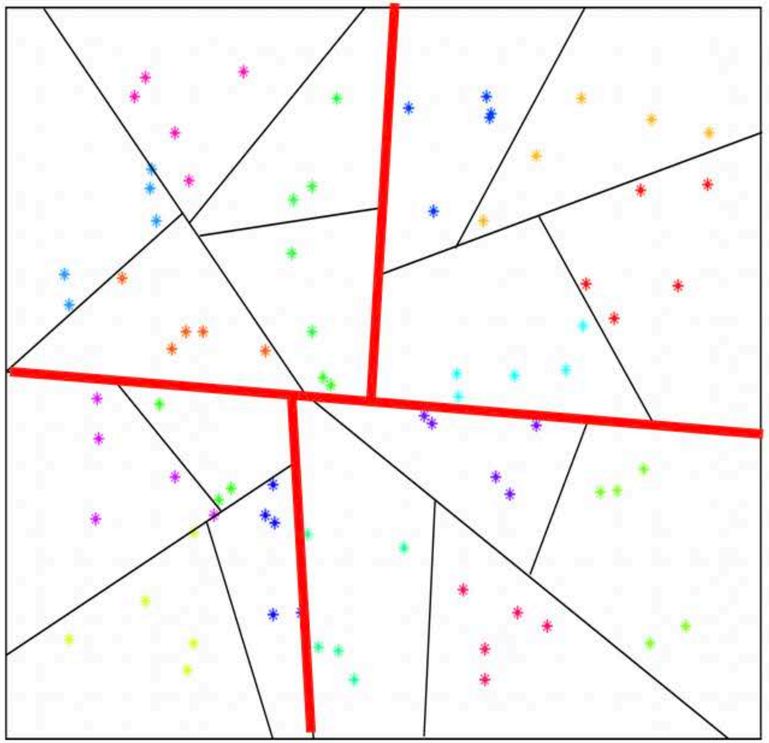
$$\mathcal{O}(N^2) \rightarrow \mathcal{O}(N \log N)$$

Questions

- Accurate far-field approximation
- Optimal complexity
- Error bounds
- HPC
- **For d=4 these have been answered**

High dimensions





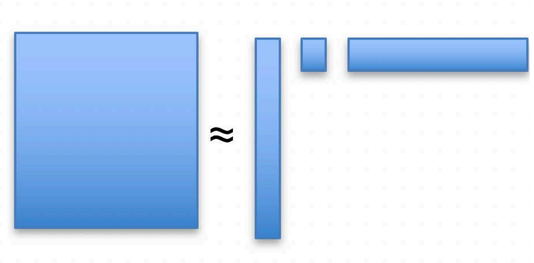
Related work — high dimensions

- Griebel et al’12 — Fast Gauss transform
- Duraiswami’06 — Improved Fast Gauss transform
- Lee, Vuduc & Gray’12 — Treecode (parallel)
- Kondor et al’16 — Wavelets in high dimensions
- Mahoney and Darve’15 — HSS matrices

Challenges in high-dimensions

- Constructing the far-field approximations
  polynomial in ambient-D
- Near-far field decomposition
  polynomial in ambient-D
- No scalable algorithms (other than Nystrom)
- Nystrom method assumes low rank
  provably not the case with increasing *N*

Randomized linear algebra — Nystrom method



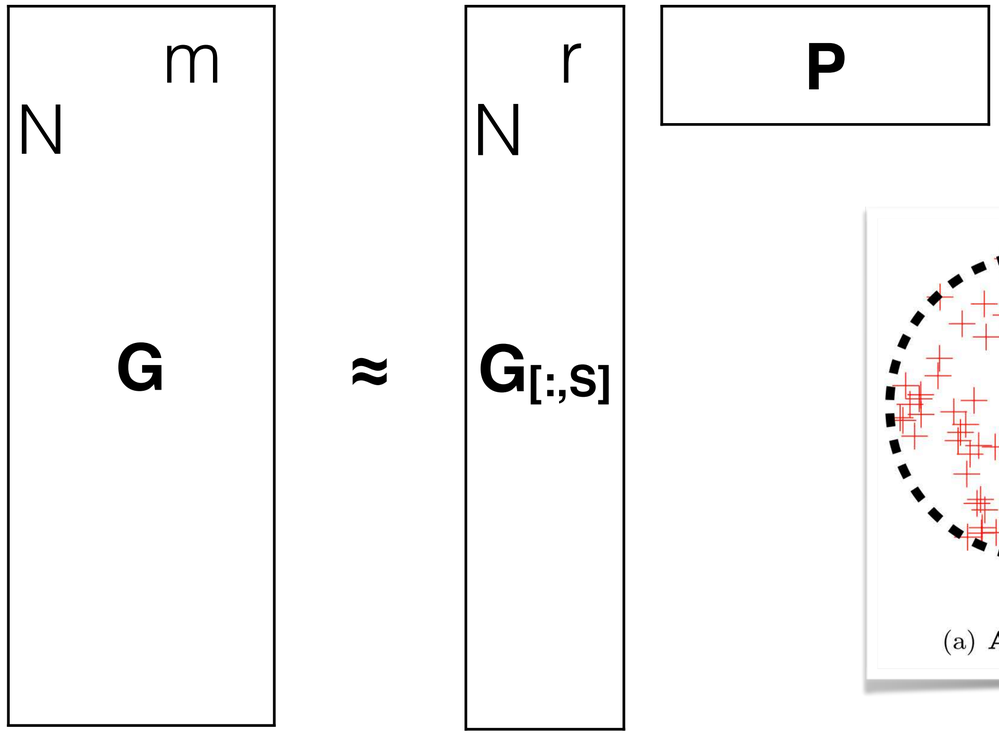
- **Low-rank** decomposition of *G*
- Random sampling of ***O(s)*** points, **s:** target **rank**

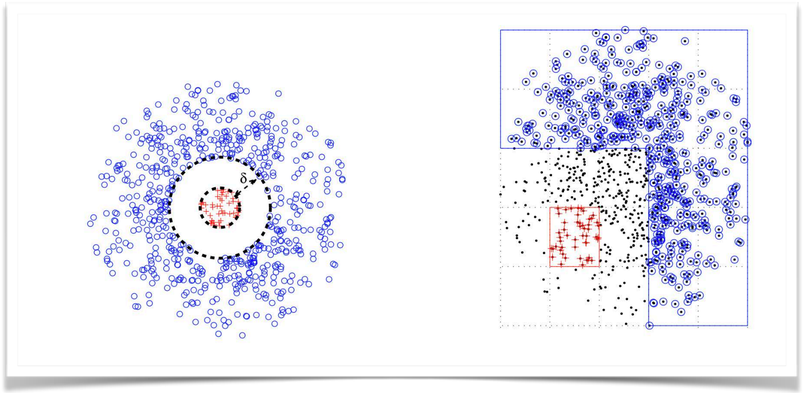
$$G \approx \tilde{G} = G_{Ns} G_{ss}^{-1} G_{sN}$$

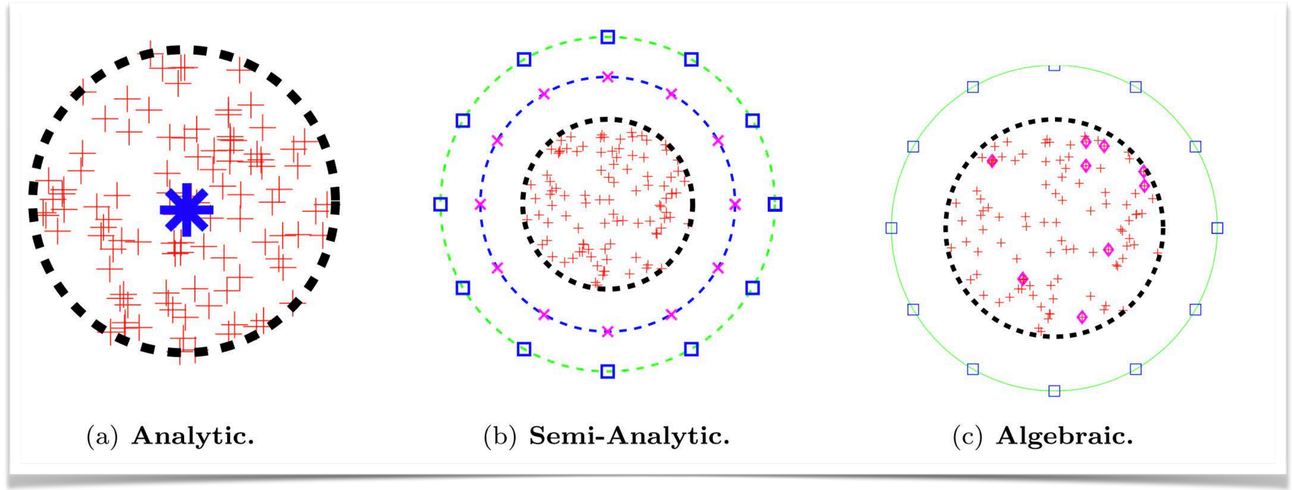
- Work $Ns + s^3$
- Error $\|G - \tilde{G}\| \leq \sqrt{1 + 6N/s}\, \sigma_{s+1}(G)$

Approximating off-diagonal blocks

Compute a rank **r** approximation of offdiagonal block **G**.
$\mathbf{G}_{[:,S]}$ is an **N**-by-**r** column submatrix of **G** (**skeleton columns**).
P is an **r**-by-**m** matrix of interpolation coefficients.







RRQR and TRSM → **O(N*m*r + m*r²)** complexity
Cheng, Gimbutas, Martinsson, Rokhlin SISC'05

Alternative algorithms for ID

Randomized methods

- Random matrix **Ω**
- Compute ID on product **ΩG**
- Use the same skeletons and **P** for an ID of **G**
- **O(N*m*r + m*r²)** or **O(N*m*log(r) + m*r²)**

Wolfe et al. ACHA'08
Martinsson et al. ACHA'11
Liberty et al. PNAS'07
Halko, Martinsson, Tropp SIREV'11

Sampling

- Sample ℓ rows of **G**
- Compute ID on $\mathbf{G}_{\mathbf{samples}}$
- Use the same skeletons and **P** for an ID of **G**
- **O(ℓ*m*r + m*r²)**

⇒ Better than random if ℓ << N

References:

Achlioptas & McSherry JACM'07
Drineas, Kannan, Mahoney SISC'06
Frieze, Kannan, Vempala JACM'04
Deshpande & Vempala '06
Deshpande et al. SIMAX'08
and others

Related Work in Sampling

- Uniform sampling - Gittens ’11, B. & March’17
- Leverage score sampling - Drineas, Mahoney, Muthukrishnan ’08
- Row ℓ^2 norm sampling - Drineas, Kannan, Mahoney ’06
- Adaptive ℓ^2 norm sampling - Deshpande & Vempala ’06
- Elementwise sampling - Achlioptas & McSherry’07, Keshavan etal’10
- Equivalent densities - Ying, Biros, Zorin ’04
- Matrix completion also related - Candes, Recht’09
- Deterministic vs. probabilistic

How many samples?

- Uniform sampling to construct ***r-rank*** approximation to ***G***
  G is *N*-by-*m* matrix

$$\|G - \tilde{G}\| \leq \sigma_{r+1} \sqrt{1 + 6N/\ell}$$

$$\ell > \text{coherence}(G) m \log\left(\frac{r}{\delta}\right)$$

δ: probability of failure

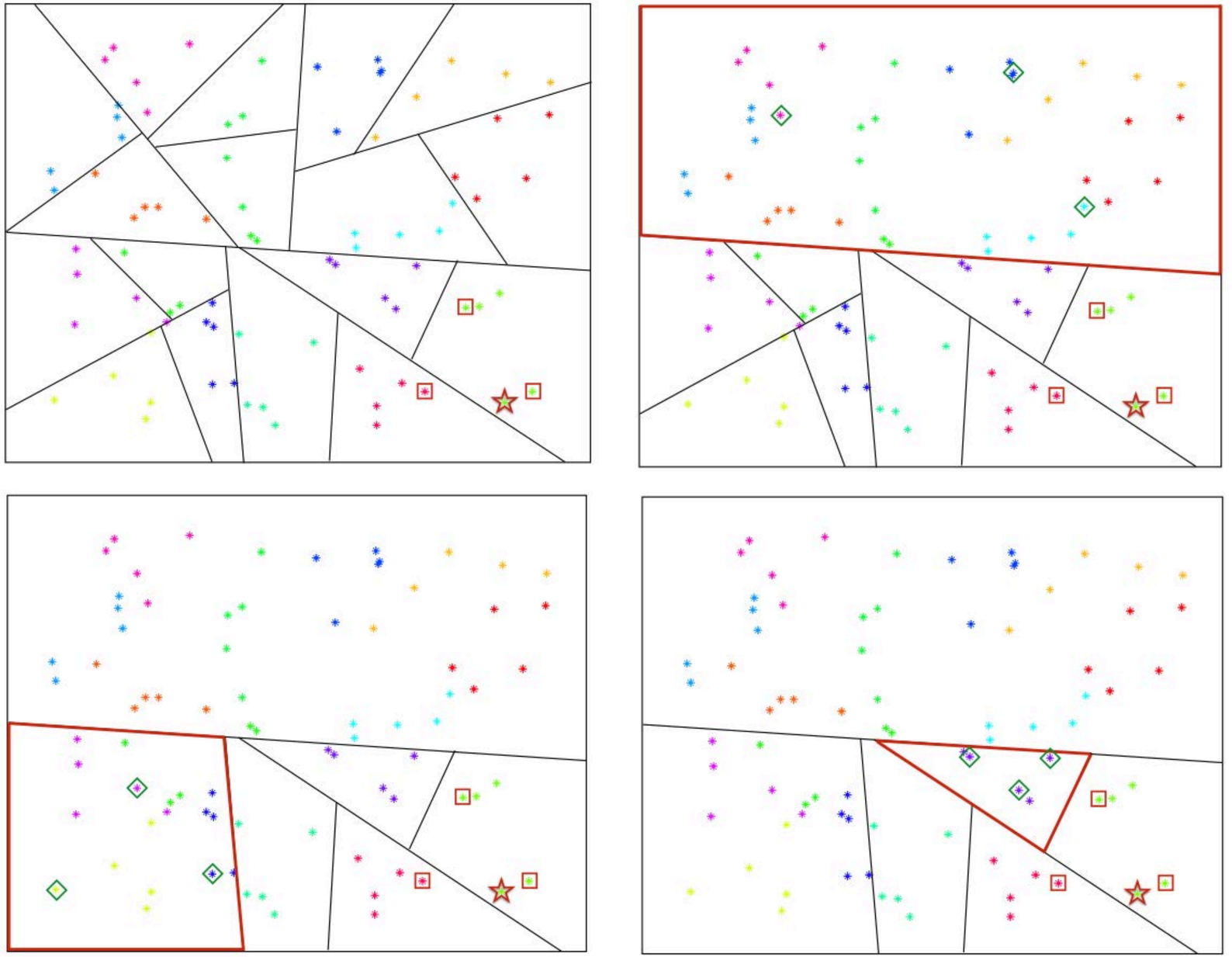
B. & March ACHA'17

ASKIT: N-body code, high-D

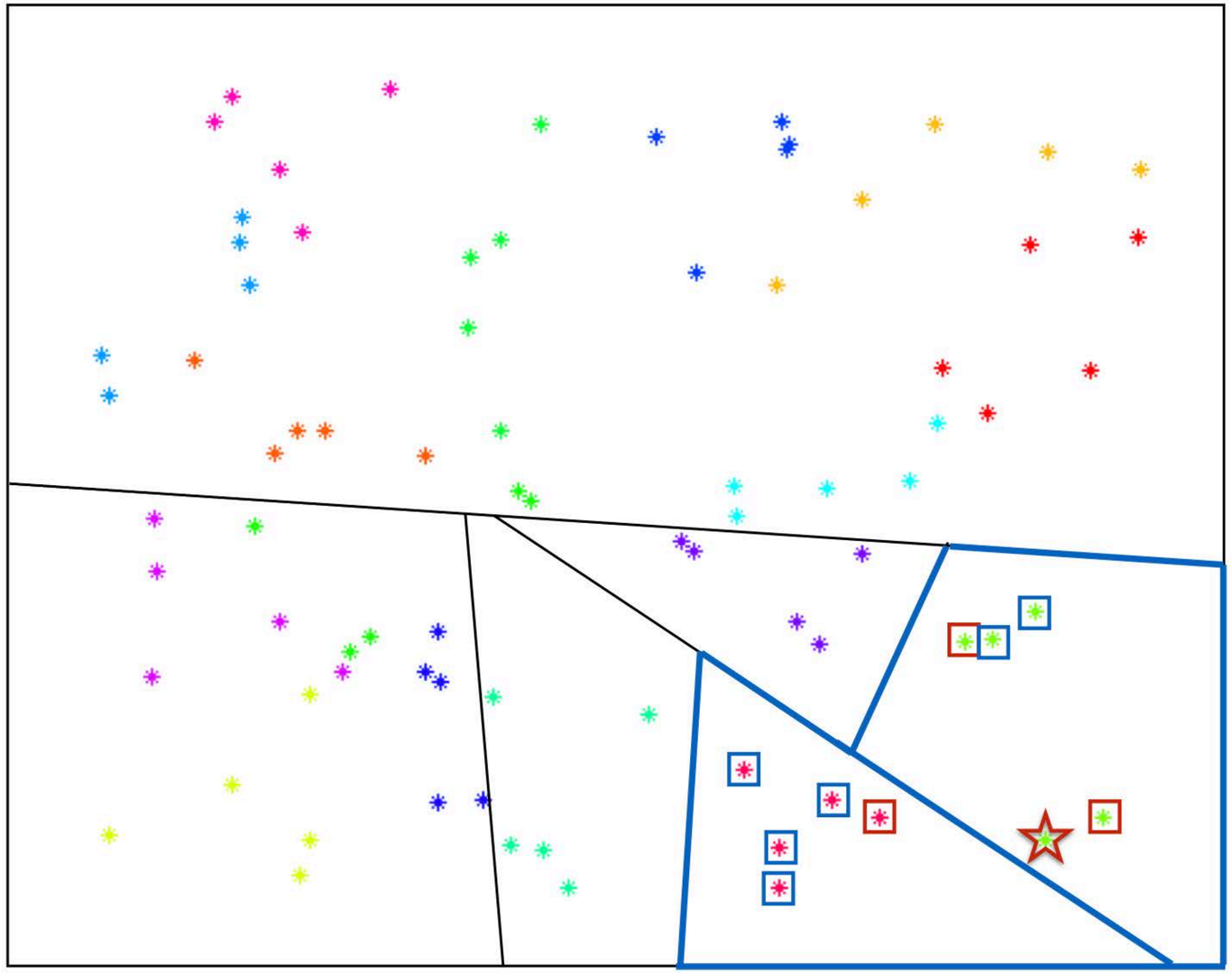
- Randomized Linear Algebra — far field approximation
- Parallel binary trees — permutation, partitioning
- Nearest neighbors — pruning and sampling
- Treecode / FMM
- MPI / OpenMP / SIMD / GPU acceleration

SISC'15,16
ACHA'15
KDD'15
SC'15
IPDPS'15,16,17

Evaluation



Evaluation



Complexity and error

- Work

RAM	skeletonize	evaluate
$(d+\kappa)N$	Ns^2	$dNs\kappa\log(\frac{N}{s})$

- Error

$$\|G-\tilde{G}\| \leq \sqrt{1+6N/s}\ \log(N/s)\ \gamma_{s+1}\ \sigma_{s+1}$$

off-diagonal

- Nystrom

$$\|G-\tilde{G}\| \leq \sqrt{1+6N/s}\ \sigma_{s+1}$$

$Ns+s^3$	diagonal

Parallel complexity

Points per MPI task $n = \frac{N}{p}$ Tree depth $D = \log \frac{N}{s}$

$$\text{Tree construction} \leq (t_s + t_w) \log^2 p \log N + (t_w \log p)(d + k) n$$

$$\text{Skeletonization} \leq t_f \left(\frac{n}{s} + \log p \right) s^3$$

$$\text{Evaluation} \leq t_s p + (t_w + t_f) d\, k\, s\, D\, n$$

Gaussian

3D, 1M points

ϵ_2	T_S	T_{LET}	T_L	T_E	$\%K$
5E-10	439	53	7	4	2.1%
5E-05	73	16	1	1	0.6%
2E-04	29	15	1	1	0.4%
1E-03	14	15	1	1	0.3%
6E-03	10	15	1	1	0.2%

64D/20D intr, 1M points

ϵ_2	T_S	T_{LET}	T_L	T_E	$\%K$
9E-06	1068	395	149	260	56%
4E-04	486	67	11	29	6.2%
5E-03	57	30	1	9	1.6%

Kernel regression

Train:

$\{x_i \in \mathbb{R}^d, c_i \in \{-1, 1\}_{i=1}^N\}$

$\{w_j\}_{j=1}^N : \sum_{j=1}^N G(x_i, x_j) w_j = c_i, \quad \forall i.$

Classify: $c(x) = \text{sign} \sum_{j=1}^N G(x, x_j) w_j$

low rank ↑

full rank ↓

COVTYPE		SUSY		MNIST2M	
h	ϵ_c	h	ϵ_c	h	ϵ_c
0.35	71.6	0.50	65.7	4	95.0
0.22	74.0	0.15	72.1	2	97.4
0.14	79.8	0.09	75.0	1	100
0.02	95.4	0.05	76.7	0.1	99.5
0.001	6.4	0.01	64.3	0.05	13.6

Kernel acceleration

Data	N	d	ϵ_2	$\%K$
Uniform	1M	64	5E-3	1.6%
Covtype	500K	54	8E-2	2.7%
SUSY	4.5M	18	5E-3	0.4 %
HIGGS	10.5M	28	1E-1	11%
BRAIN	10.5M	246	5E-3	0.9%

Kernel regression scaling

MNIST dataset for OCR
strong scaling, 8M points d=784

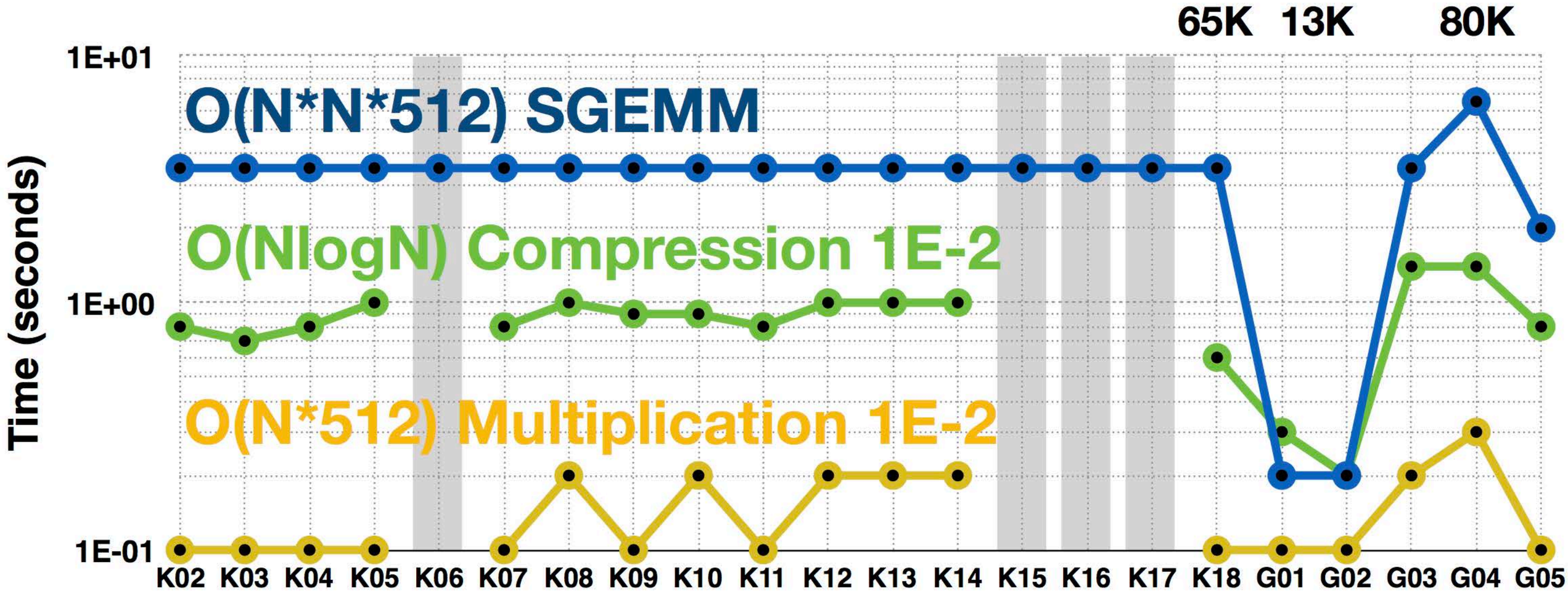
#cores	512	2,048	4,096	8,192	16,384
Skel. (Alg. 2)	1,295	465	370	305	269
Lists (Alg. 3)	729	177	87	46	23
LET (Eq. 7)	273	136	107	87	71
Eval. (Eq. 14)	157	67	42	28	23
Total	2,471	862	621	483	394
Efficiency	1.00	0.72	0.50	0.32	0.20

Extensions to arbitrary SPD matrices

- Construct HSS approximation for a generic dense SPD matrix
  FMM / N-body acceleration but no points are given

- Four components
  - Permute order to expose low-rank structure — O(N)
  - Compress blocks — O(N logN)
  - Fast Matvec — O(N)
  - HPC implementation (task par/ async + ARM, x86/KNL, GPUs)

Representative results



Geometry oblivious

- Permute matrix to expose low-rank structure
- Geometry-based algorithms: need distance between indices i,j
- Gram vectors
  distance(i,j) = function(G_{ii}, G_{ij}, G_{jj})

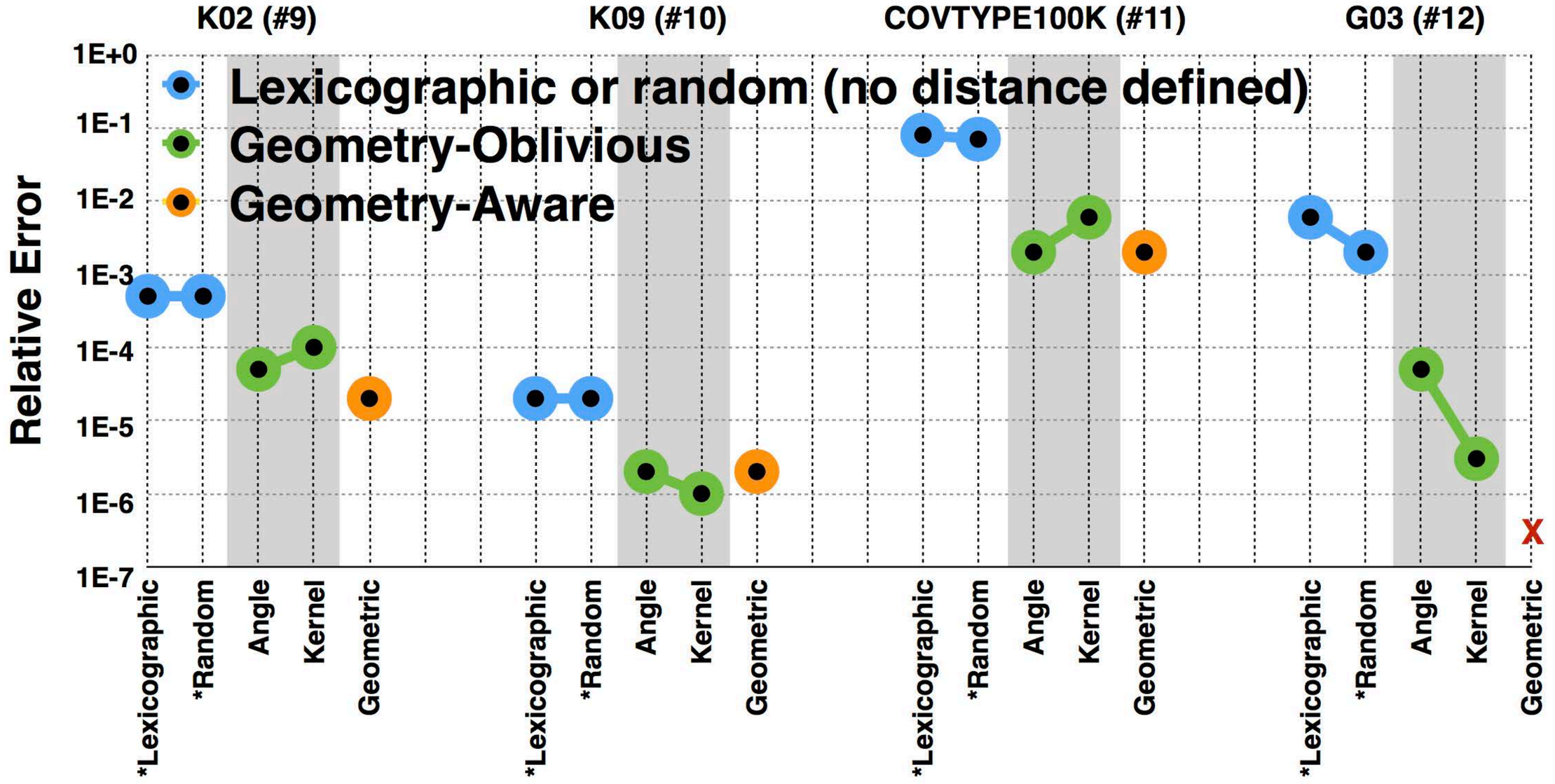
Geometry oblivious

- Gram vectors (G is SPD): $G_{ij} = \phi_i \cdot \phi_j$
- Distances

Euclidean	$\|\phi_i - \phi_j\|_2^2$	$G_{ii} + G_{jj} - 2G_{ij}$
Angle	$\sin\left(\frac{\phi_i \cdot \phi_j}{\|\phi_i\|_2 \|\phi_j\|_2}\right)$	$1 - \frac{Gij^2}{G_{ii}G_{jj}}$

GOFMM compression algo outline

- Distance(i,j) defined using matrix entries
- Symmetrically permute matrix using binary tree construction
- Use randomized nearest neighbors to find Neighbors(i)
  - Direct interactions (strong admissibility)
  - Sampling to construct far-field approximation



***Lexicographic and *Random does not use any distance (no sparse correction)**

K02 is a 2D regularized inverse Laplacian squared, resembling the Hessian operator of a PDE-constrained optimization problem.

**The Laplacian is discretized using a 5-stencil finite-difference scheme with Dirichlet boundary conditions on a regular grid.*

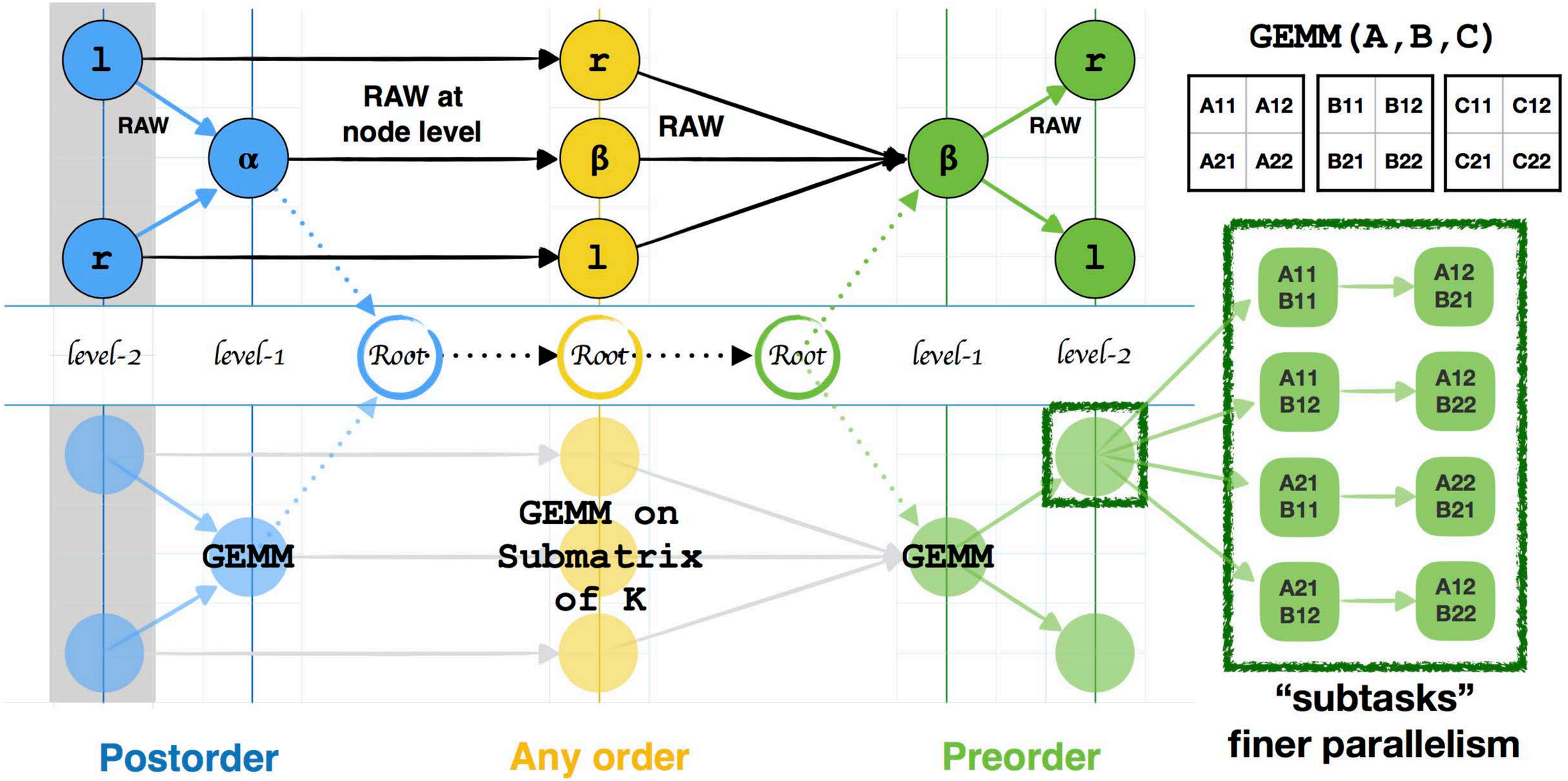
K04–K10 are kernel matrices in six dimensions (Gaussians, Laplacian, Green's function, polynomial and cosine-similarity).

G01–G05 are the inverse graph Laplacian of five graphs from UFL.

GOFMM vs Other implementations

	HODLR			STRUMPACK			GOFMM		
case	ϵ_2	Comp	Eval	ϵ_2	Comp	Eval	ϵ_2	Comp	Eval
K02	6E−5	0.6	2.7	1E−4	9.2	0.6	2E−5	1.0	0.3
K04	6E−5	0.7	2.7	1E−4	507.8	7.8	2E−5	1.0	0.5
K07	7E−5	0.9	3.1	2E−4	528.4	8.2	4E−5	0.6	0.2
K12	6E−5	0.7	2.7	2E−4	18.8	0.8	1E−4	0.6	0.2
K17	1E−1	862.2	37.6	2E−1	663.4	8.2	9E−2	48.8	3.1
G03	3E−4	**12.9**	9.7	3E−2	**29.8**	1.3	8E−5	**0.5**	0.8

MATVEC Dependencies



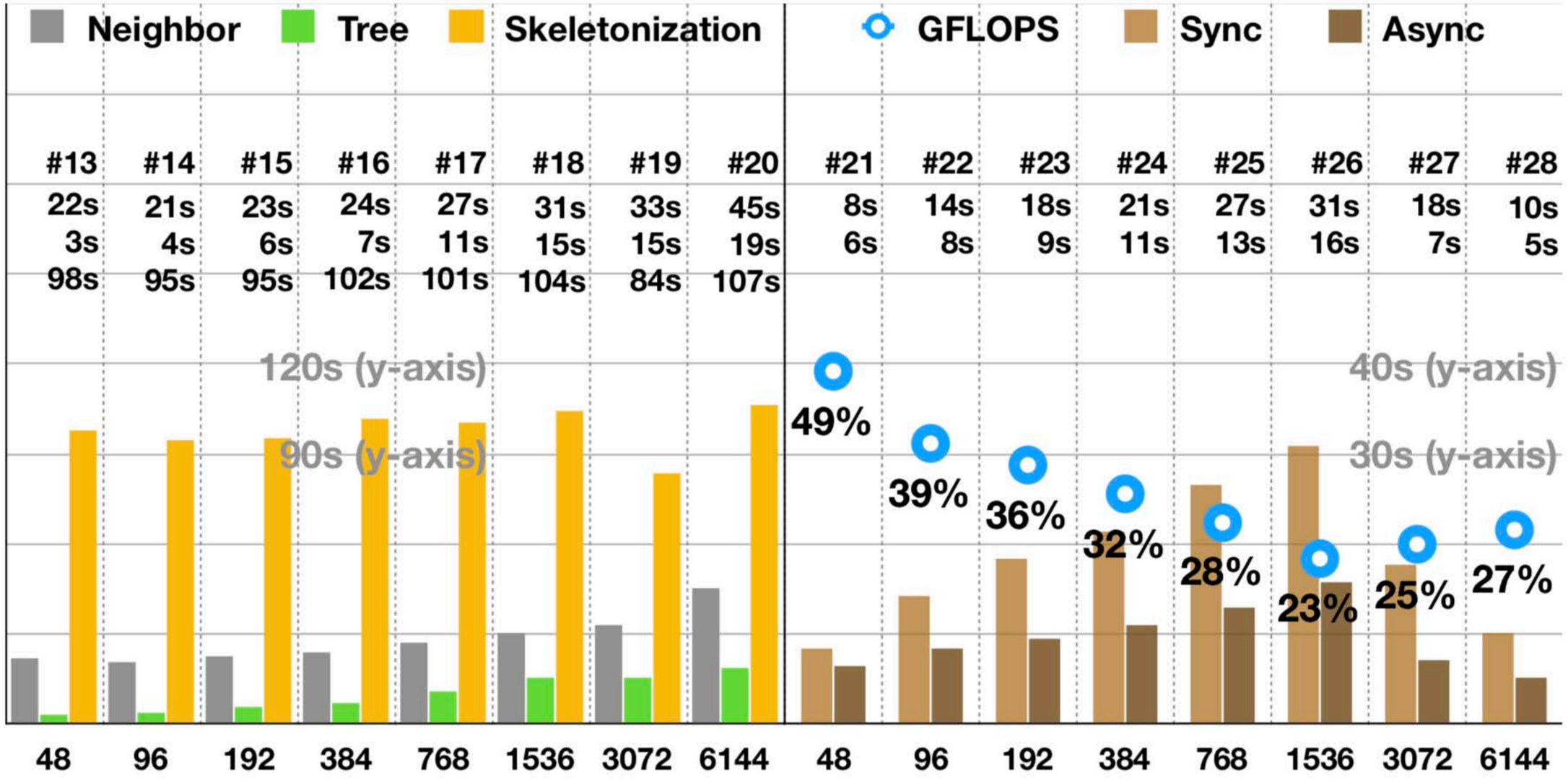
Computational primitives

- Geometric — distance calculation / projections
- Analytic — special functions / fast transforms
- Algebraic — BLAS / QR / SVD / Cholesky
- Combinatorial — hash / sort / merge / select / search
- Memory — permute / pack / gather / scatter

Multiple architectures

Arch	Budget	ϵ_2	Comp	GFs	Eval	GFs
MNIST60K, $h1$, $\kappa32$, $m512$, $s128$, $r256$						
ARM	5%	5E-3	285	3	520	12
COVTYPE100K, $h1$, $\kappa32$, $m512$, $s128$, $r256$						
ARM	5%	8E-4	71	2	61	10
COVTYPE100K, $h0.1$, $\kappa32$, $m800$, $s512$, $r512$						
CPU	12%	2E-3	30	30	4.1	679
CPU+GPU	12%	3E-3	33	29	1.7	1952
KNL	12%	2E-3	48	25	3.2	1125

GOFMM on Stampede 2



Summary

- Kernel methods in CSE
- Generalization to SPD matrices and high dimensions
- References: IPDPS'17, SC'17, SISC'16, ACHA'17, SC'18