# Integral Equations and Fast Algorithms
# Lecture 26: Fast Algorithms

CS 598AK · November 21, 2013

# Outline

Fast Algorithms

# Integral equations + computers

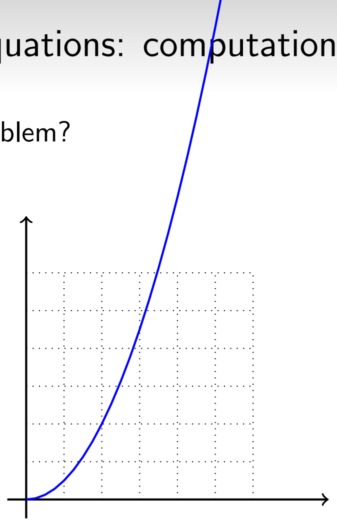
$$O(n^2)$$

# Integral equations: computational expense

Why is $O(n^2)$ a problem?

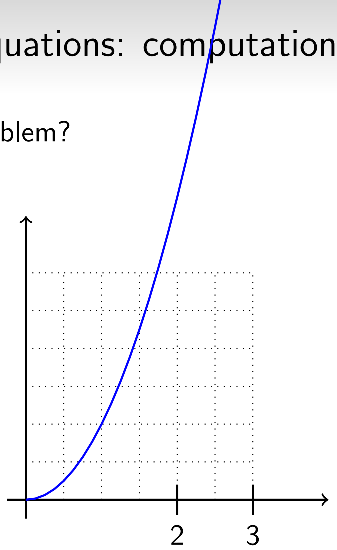
# Integral equations: computational expense

Why is $O(n^2)$ a problem?



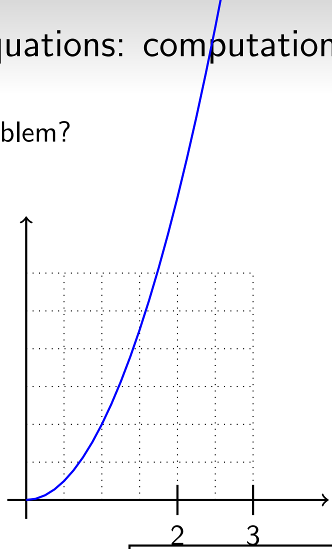
# Integral equations: computational expense

Why is $O(n^2)$ a problem?



# Integral equations: computational expense

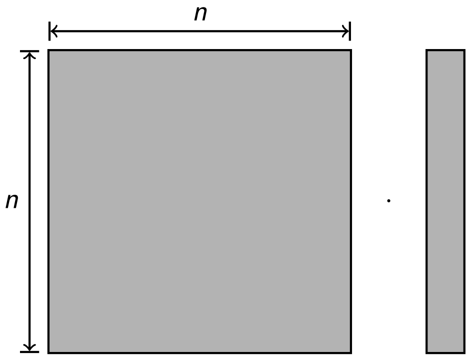
Why is $O(n^2)$ a problem?




Why is $O(n^2)$ *storage complexity* a problem specifically?

# Integral equations: computational expense

$O(n^2)$ not in principle incorrect:
Natural complexity of a dense mat-vec.

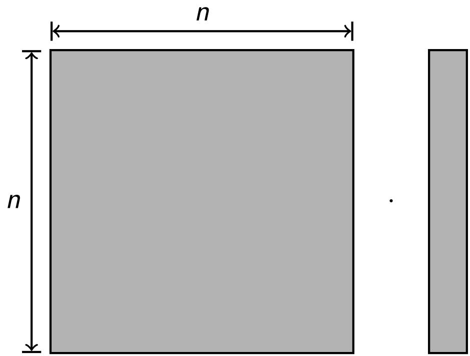



**Or**, to be more precise:

$$O(\langle \#\text{sources} \rangle \cdot \langle \#\text{targets} \rangle)$$

# Integral equations: computational expense

$O(n^2)$ not in principle incorrect:
Natural complexity of a dense mat-vec.

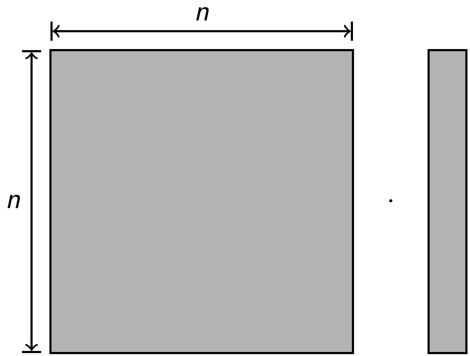



**Or**, to be more precise:

$$O(\langle \#\text{sources} \rangle \cdot \langle \#\text{targets} \rangle) = O(\langle \#\text{rows} \rangle \cdot \langle \#\text{columns} \rangle)$$

# Integral equations: computational expense

$O(n^2)$ not in principle incorrect:
Natural complexity of a dense mat-vec.




**Or**, to be more precise:

$$O(\langle \#\text{sources} \rangle \cdot \langle \#\text{targets}$$

Special dense matrices with faster mat-vecs?

# Faster dense mat-vecs

If $A = uv^T$, then

$$Ax = \underbrace{(uv^T)}_{O(n^2)} x$$

# Faster dense mat-vecs

If $A = uv^T$, then

$$Ax = \underbrace{(uv^T)}_{O(n^2)} x = \underbrace{u \underbrace{(v^T x)}_{O(n)}}_{O(n)}$$

# Faster dense mat-vecs

If $A = uv^T$, then

$$Ax = \underbrace{(uv^T)}_{O(n^2)} x = \underbrace{u \underbrace{(v^T x)}_{O(n)}}_{O(n)}$$

If $A = u_1 v_1^T + u_2 v_2^T + \cdot + u_k v_k^T$, then

$$A = ?$$

# Faster dense mat-vecs

If $A = uv^T$, then

$$Ax = \underbrace{(uv^T)}_{O(n^2)} x = \underbrace{u \underbrace{(v^T x)}_{O(n)}}_{O(n)}$$

If $A = u_1 v_1^T + u_2 v_2^T + \cdot + u_k v_k^T$, then

$$A = ?$$

Computational cost?

# Faster dense mat-vecs

If $A = uv^T$, then

$$Ax = \underbrace{(uv^T)}_{O(n^2)} x = \underbrace{u \underbrace{(v^T x)}_{O(n)}}_{O(n)}$$

If $A = u_1 v_1^T + u_2 v_2^T + \cdot + u_k v_k^T$, then

$$A =?$$

Computational cost?

Relation to 1D fast algorithm (HW 4)?

# Faster dense mat-vecs

If $A = uv^T$, then

$$Ax = \underbrace{(uv^T)}_{O(n^2)} x = \underbrace{u \underbrace{(v^T x)}_{O(n)}}_{O(n)}$$

If $A = u_1 v_1^T + u_2 v_2^T + \cdot + u_k v$

Computational cost?

Relation to 1D fast algorithm (HW 4)?

Relation to separation-of-variables?

# Faster dense mat-vecs

If $A = uv^T$, then

$$Ax = \underbrace{(uv^T)}_{O(r}x = u\underbrace{(v^T x)}$$

If $A = u_1 v_1^T + u_2 v_2^T + \cdot + u_k v$

Computational cost?

Relation to 1D fast algorithm (HW 4)?

Relation to separation-of-variables?

Can any matrix be written like this?

# Faster dense mat-vecs

If $A = uv^T$, then

$$Ax = \underbrace{(uv$$

$O($

If $A = u_1 v_1^T + u_2 v_2^T + \cdot + u_k v$

Computational cost?

Relation to 1D fast algorithm (HW 4)?

Relation to separation-of-variables?

Can any matrix be written like this?

Demo

# Faster dense mat-vecs

If $A = uv^T$, then

$$Ax = \underbrace{(uv$$

$O($

If $A = u_1 v_1^T + u_2 v_2^T + \cdot + u_k v$

Computational cost?

Relation to 1D fast algorithm (HW 4)?

Relation to separation-of-variables?

Can any matrix be written like this?

Demo

Define numerical rank

# Numerical rank

$$\text{rank}_\varepsilon(A) := \min\{k : \exists U \in \mathbb{R}^{n\times k}, V \in \mathbb{R}^{m\times k}, \|A - UV^T\| < \varepsilon\}$$

# Particle low-rank demo

# Particle low-rank demo

Drawback?

# Particle low-rank demo

Drawback?

Wouldn't it be nice if the low-rank factorization were reusable for lots of targets?

# Multipole Expansions

Multivariate Taylor expansion:

$$f(y) \approx \sum_{|\alpha| \le p} \left( \frac{\partial^{|\alpha|}}{\partial \eta^\alpha} f(\eta)|_{\eta = c} \right) \frac{(y-c)^\alpha}{\alpha!}$$

# Multipole Expansions

Multivariate Taylor expansion:

$$f(y) \approx \sum_{|\alpha| \le p} \left( \frac{\partial^{|\alpha|}}{\partial \eta^{\alpha}} f(\eta)|_{\eta=c} \right) \frac{(y-c)^{\alpha}}{\alpha!}$$

What's a multi-index $\alpha$?

# Multipole Expansions

Multivariate Taylor expansion:

$$f(y) \approx \sum_{|\alpha| \le p} \left( \frac{\partial^{|\alpha|}}{\partial \eta^\alpha} f(\eta)|_{\eta=c} \right) \frac{(y-c)^\alpha}{\alpha!}$$

What's a multi-index $\alpha$?

What are these?
$\partial^{|\alpha|}/\partial x^\alpha$, $x^\alpha$, $\alpha!$, $|\alpha|$

# Multipole Expansions

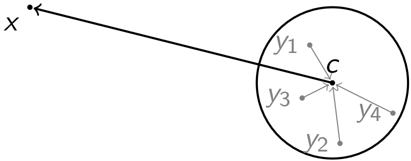
Multivariate Taylor expansion:

$$f(y) \approx \sum_{|\alpha| \leq p} \left( \frac{\partial^{|\alpha|}}{\partial \eta^{\alpha}} f(\eta)|_{\eta=c} \right) \frac{(y-c)^{\alpha}}{\alpha!}$$

# Multipole Expansions

Multivariate Taylor expansion:

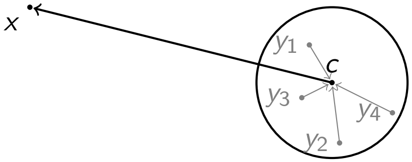
$$f(y) \approx \sum_{|\alpha| \le p} \left( \frac{\partial^{|\alpha|}}{\partial \eta^\alpha} f(\eta)|_{\eta=c} \right) \frac{(y-c)^\alpha}{\alpha!}$$



# Multipole Expansions

Multivariate Taylor expansion:

$$f(y) \approx \sum_{|\alpha| \le p} \left( \frac{\partial^{|\alpha|}}{\partial \eta^\alpha} f(\eta)|_{\eta=c} \right) \frac{(y-c)^\alpha}{\alpha!}$$




Now add target point $x$ as a parameter:

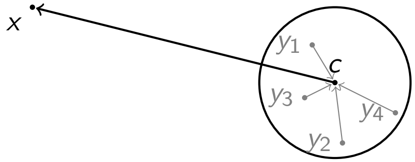
$$f(x, y) \approx \sum_{|\alpha| \le p} \left( \frac{\partial^{|\alpha|}}{\partial \eta^\alpha} f(x, \eta)|_{\eta=c} \right) \frac{(y-c)^\alpha}{\alpha!}$$

# Multipol

Potential confusion: "Basis" and "coefficient" swap roles!

Multivariate Taylor expansion:

$$f(y) \approx \sum_{|\alpha|\le p} \left( \frac{\partial}{\partial \eta^{\alpha}} f(\eta)|_{\eta=c} \right) \frac{}{\alpha!}$$




Now add target point $x$ as a parameter:

$$f(x, y) \approx \sum_{|\alpha|\le p} \left( \frac{\partial^{|\alpha|}}{\partial \eta^{\alpha}} f(x, \eta)|_{\eta=c} \right) \frac{(y-c)^{\alpha}}{\alpha!}$$

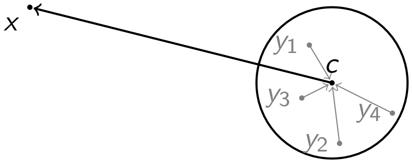
# Multipol

Multivariate Taylor expansion:

$$f(y) \approx \sum_{|\alpha| \le p} \left( \frac{\partial}{\partial} \right.$$

Potential confusion: "Basis" and "coefficient" swap roles!

Where is this accurate?

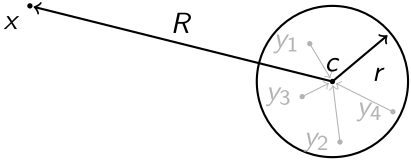



Now add target point $x$ as a parameter:

$$f(x, y) \approx \sum_{|\alpha| \le p} \left( \frac{\partial^{|\alpha|}}{\partial \eta^{\alpha}} f(x, \eta)|_{\eta = c} \right) \frac{(y-c)^{\alpha}}{\alpha!}$$

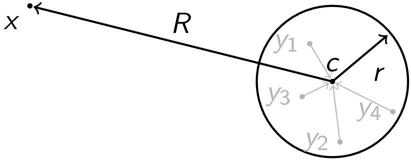
# Multipole Error Estimate

$$f(x,y) = \sum_{|\alpha|\le p} \left( \frac{\partial^{|\alpha|}}{\partial \eta^{\alpha}} f(x,\eta)|_{\eta=c} \right) \frac{(y-c)^{\alpha}}{\alpha!} + \ldots ?$$



# Multipole Error Estimate

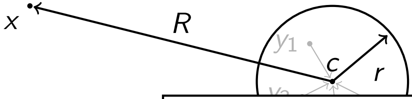
$$f(x, y) = \sum_{|\alpha| \le p} \left( \frac{\partial^{|\alpha|}}{\partial \eta^\alpha} f(x, \eta)|_{\eta = c} \right) \frac{(y - c)^\alpha}{\alpha!} + \ldots ?$$




Remainder term?
(Assume log kernel.)

# Multipole Error Estimate

$$f(x,y) = \sum_{|\alpha|\le p} \left( \frac{\partial^{|\alpha|}}{\partial \eta^{\alpha}} f(x,\eta)|_{\eta=c} \right) \frac{(y-c)^{\alpha}}{\alpha!} + \ldots ?$$




Remainder term?
(Assume log kernel.)

How does this relate to low-rank property?

# Fast multipole method

Use multipole expansions (and more) to compute

$$\varphi(x_i) = \sum_{j=1}^{n_{\text{sources}}} f(|x_i - y_j|) q_j \qquad (i = 1, \ldots, n_{\text{targets}})$$

quickly—in $O(n_{\text{sources}} + n_{\text{targets}})$ operations

# Fast multipole method

Use multipole expansions (and more) to compute

$$\varphi(x_i) = \sum_{j=1}^{n_{\text{sources}}} f(|x_i - y_j|) q_j \qquad (i = 1, \ldots, n_{\text{targets}})$$

quickly—in $O(n_{\text{sources}} + n_{\text{targets}})$ operations

$O(n) \ldots$ (some problems $O(n \log n)$)

# Fast multipole method

Use multipole expansions (and more) to compute

$$\varphi(x_i) = \sum_{j=1}^{n_{\text{sources}}} f(|x_i - y_j|)q_j \qquad (i = 1, \ldots, n_{\text{targets}})$$

quickly—in $O(n_{\text{sources}} + n_{\text{targets}})$ operations

$O(n) \ldots$ (some problems $O(n \log n)$)

Restriction on kernel?

# Fast multipole method

Use multipole expansions (and more) to compute

$$\varphi(x_i) = \sum_{j=1}^{n_{\text{sources}}} f(|x_i - y_j|) q_j \qquad (i = 1, \ldots, n_{\text{targets}})$$

quickly—in $O(n_{\text{sources}} + n_{\text{targets}})$ operations

$O(n) \ldots$ (some problems $O(n \log n)$)

Restriction on kernel?

Difficult bit: lots of far field to sum

# Fast multipole method

Use multipole expansions (and more) to compute

$$\varphi(x_i) = \sum_{j=1}^{n_{\text{sources}}} f(|x_i - y_j|)q_j \qquad (i = 1, \ldots, n_{\text{targets}})$$

quickly—in $O(n_{s}$

$O(n) \ldots$ (some problems $O(n \log n)$)

Restriction on kernel?

Difficult bit: lots of far field to sum

Near field: easy. Why?

# Fast multipole method

Use multipole expansions (and more) to compute

$$\varphi(x_i) = \sum^{n_{\text{sources}}}$$

quickly—in $O(n_{\text{s}}$

$O(n) \ldots$ (some problems $O(n \log n)$)

Restriction on kernel?

Difficult bit: lots of far field to sum

Near field: easy. Why?

Near field: replaceable. Why?

# Fast multipole method

Use multipole ex

$\varphi(x_i) =$

quickly—in $O(n_s$

$O(n) \dots$ (some problems $O(n \log n)$)

Restriction on kernel?

Difficult bit: lots of far field to sum

Near field: easy. Why?

Near field: replaceable. Why?

Only point particles.
Relationship to layer potentials?

# Fast multipole method

Use multipole ex

$\varphi(x_i) =$

quickly—in $O(n_s$

$O(n)$ ... (some problems $O(n \log n)$)

Restriction on kernel?

Difficult bit: lots of far field to sum

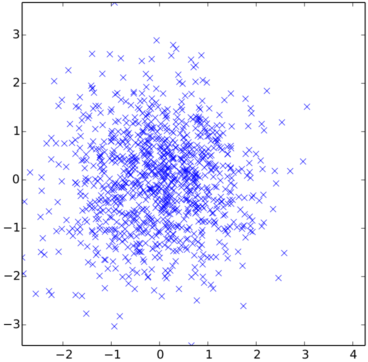
Near field: easy. Why?

Near field: replaceable. Why?

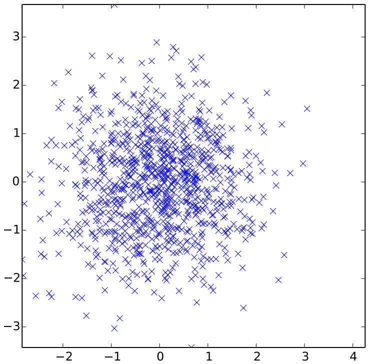
Only point particles.
Relationship to layer potentials?

Can it help if $n_{\text{sources}} = 1$?

# Structuring the Computation

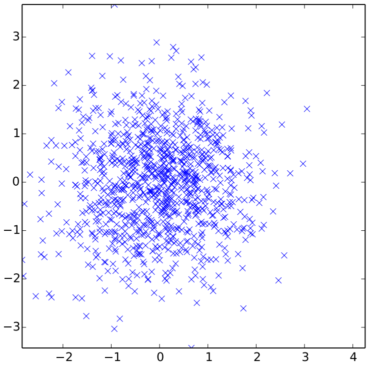


# Structuring the Computation

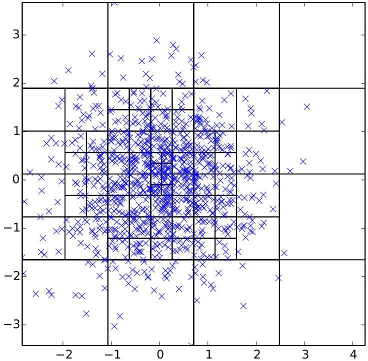



Distinction: sources/targets?

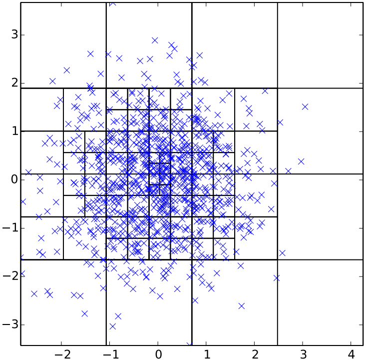
# Structuring the Computation



# Structuring the Computation

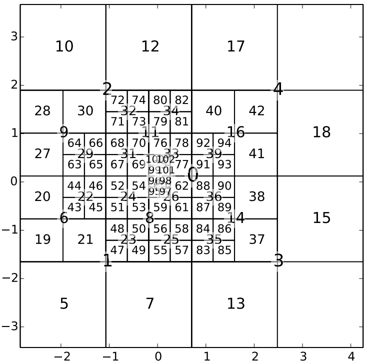


# Structuring the Computation

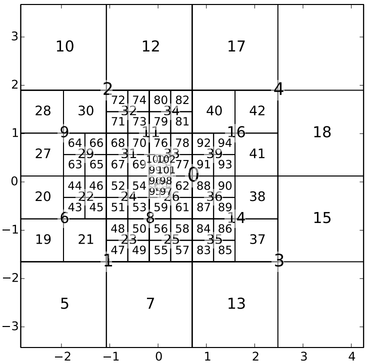



Condition for opening of new box?

# Structuring the Computation

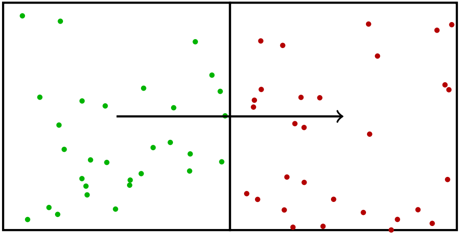


# Structuring the Computation

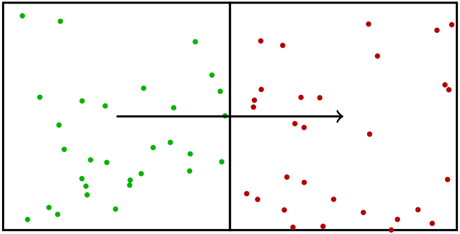



See the tree?

# Box separation

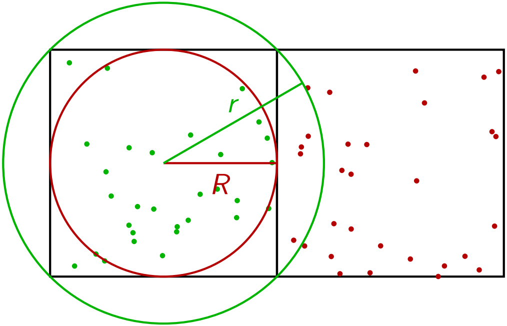


# Box separation

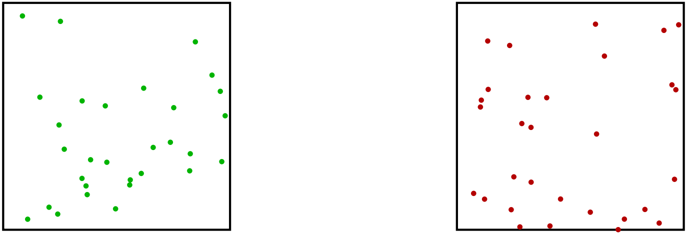


Can a multipole mediate this interaction?

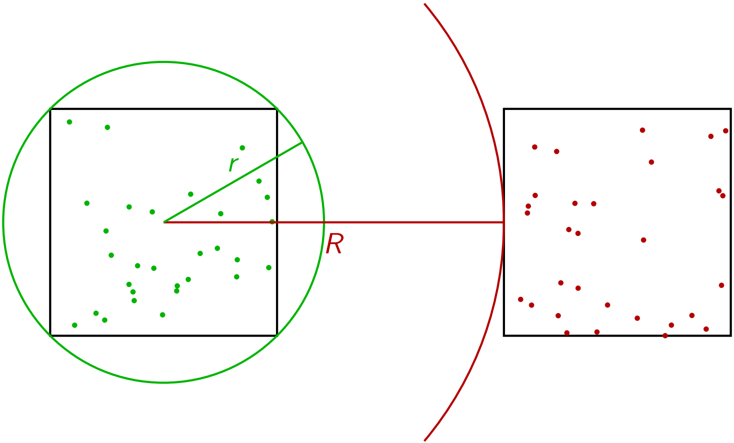
# Box separation



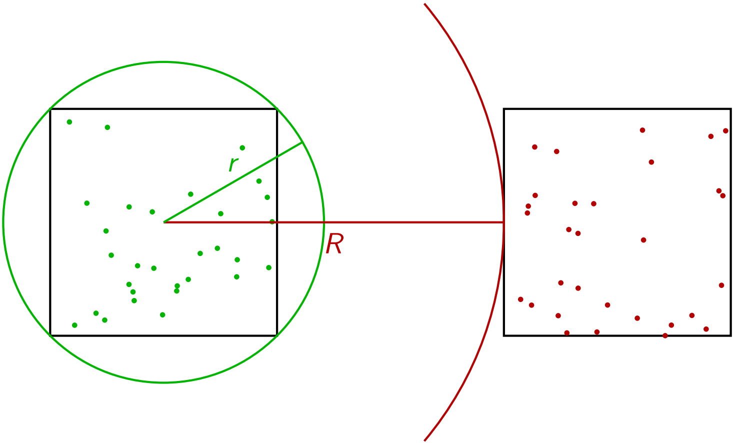
# Box separation



# Box separation

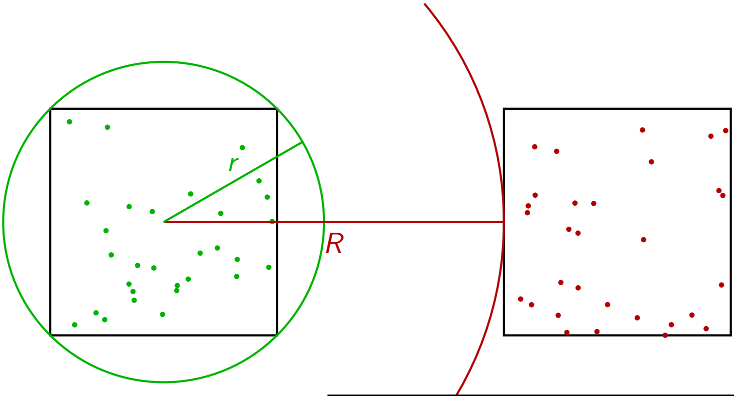


# Box separation




"Well-separated"

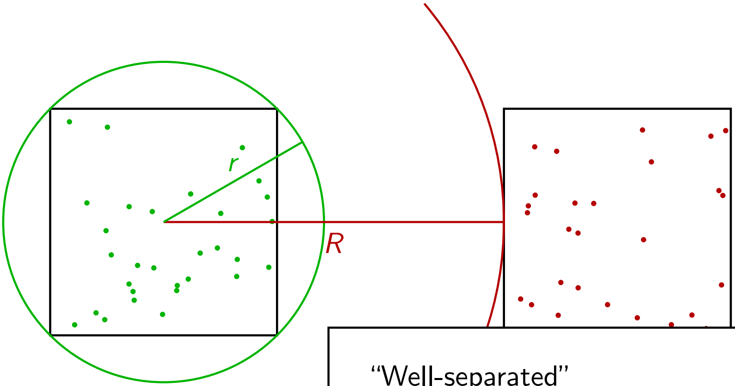
# Box separation




“Well-separated”

Error estimate?

# Box separation

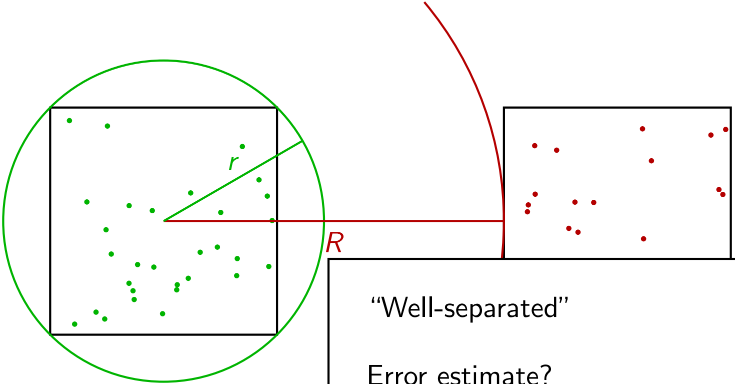



“Well-separated”

Error estimate?

In 3D?

# Box separation



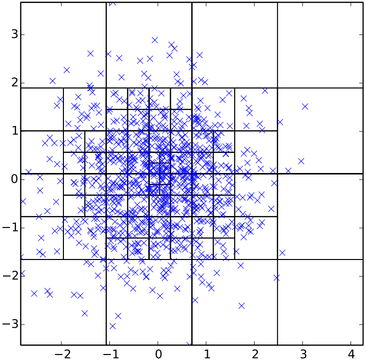

"Well-separated"

Error estimate?

In 3D?

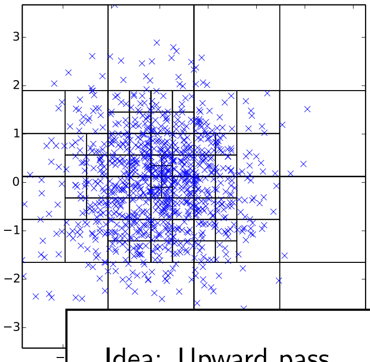
In *n*D?

# Structuring the Computation, part II




Idea: Upward pass
Algorithm?

# Structuring the Computation, part II




Idea: Upward pass
Algorithm?

Analytical basis for upward pass?

# Implementing the upward pass

$$f(x,y) \approx \sum_{|\alpha|\le p} \left( \frac{\partial^{|\alpha|}}{\partial \eta^{\alpha}} f(x,\eta)|_{\eta=c} \right) \frac{(y-c)^{\alpha}}{\alpha!}$$

# Implementing the upward pass

$$f(x,y) \approx \sum_{|\alpha| \le p} \left( \frac{\partial^{|\alpha|}}{\partial \eta^{\alpha}} f(x,\eta)|_{\eta=c} \right) \frac{(y-c)^{\alpha}}{\alpha!}$$

What are we trying to accomplish here?

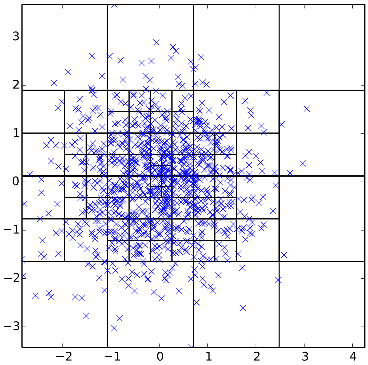
# Implementing the upward pass

$$f(x, y) \approx \sum_{|\alpha| \leq p} \left( \frac{\partial^{|\alpha|}}{\partial \eta^{\alpha}} f(x, \eta)|_{\eta = c} \right) \frac{(y - c)^{\alpha}}{\alpha!}$$

What are we trying to accomplish here?

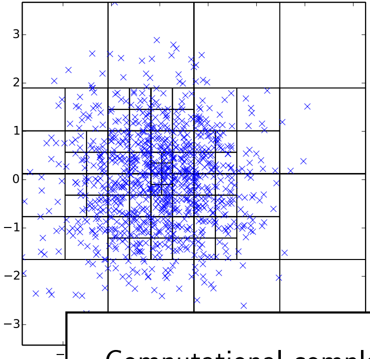
"Translation operator"

# Structuring the Computation, part III




Computational complexity with upward pass?

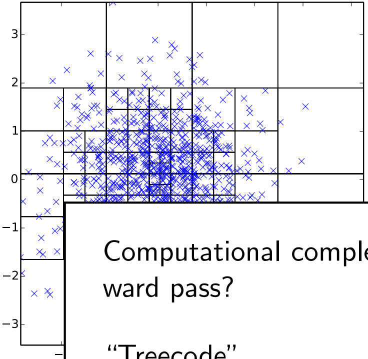
# Structuring the Computation, part III




Computational complexity with upward pass?

"Treecode"

# Structuring the Computation, part III



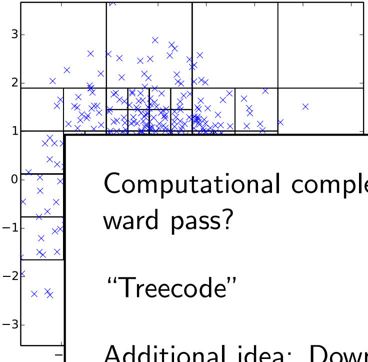

Computational complexity with upward pass?

"Treecode"

Additional idea: Downward pass

Algorithm?

# Structuring the Computation, part III



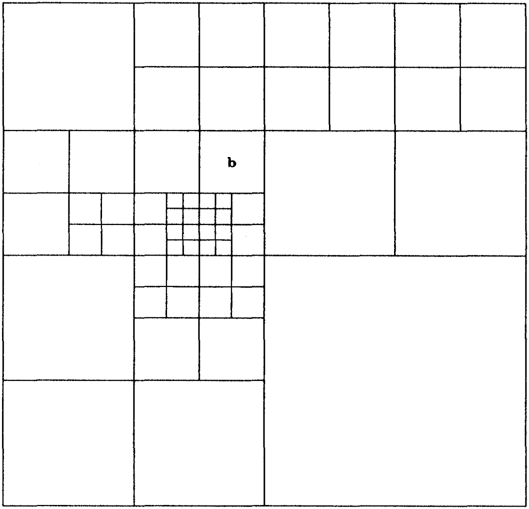

Computational complexity with upward pass?

"Treecode"

Additional idea: Downward pass
Algorithm?

Analytical idea for downward pass?

# Structuring the Computation, part IV




[Carrier et al. '88]

# Interaction Lists: Terminology

List 1 If $b$ is childless, $L_1(b)$ consists of $b$ and all childless boxes adjacent to $b$. If $b$ has children, $L_1(b)$ is empty.

List 2 $L_2(b)$ is formed by all the children of the colleagues of b's parent that are well-separated from b.

List 3 $L_3(b)$ is empty if $b$ is a parent box, and consists of all descendants of b's colleagues whose parents are adjacent to $b$, but who are not adjacent to $b$ themselves, if $b$ is a childless box.

List 4 $L_4(b) := \{\text{box } c : b \in L_3(c)\}$

List 5 $L_5(b)$ consists of all boxes that are well-separated from b's parent

[Carrier et al. '88]

# Interaction Lists: Terminology

List 1 If $b$ is childless, $L_1(b)$ consists of $b$ and all childless boxes adjacent to $b$. If $b$ has children, $L_1(b)$ is empty.

List 2 $L_2(b)$ is formed by all the children of the colleagues of b's parent that are well-separated from b.

List 3 $L_3(b)$ is empty if $b$ is a parent box, and consists of all descendants of b's colleagues whose parents are adjacent to $b$, but who are not adjacent to $b$ themselves, if $b$ is a childless box.

List 4 $L_4(b) := \{\text{box } c : b \in L_3(c)\}$

List 5 $L_5(b)$ consists of all boxes that are well-separated from b's parent

[Carrier et al. '88]

Colleagues?

# Interaction Lists: Terminology

List 1 If $b$ is childless, $L_1(b)$ consists of $b$ and all childless boxes adjacent to $b$. If $b$ has children, $L_1(b)$ is empty.

List 2 $L_2(b)$ is formed by all the children of the colleagues of b's parent that are well-separated from b.

List 3 $L_3(b)$ is empty if $b$ is a parent box, and consists of all descendants of b's colleagues whose parents are adjacent to $b$, but who are not adjacent to $b$ themselves, if $b$ is a childless box.

List 4 $L_4(b) := \{$

List 5 $L_5(b)$ cons

from b's p

[Carrier et al. '88]

Colleagues?

Observation: $c \in L_3(b)$ is separated from $b$ by a distance greater than or equal to the length of the side of $c$.

# Interaction Lists: Terminology

List 1 If $b$ is childless, $L_1(b)$ consists of $b$ and all childless boxes adjacent to $b$. If $b$ has children, $L_1(b)$ is empty.

List 2 $L_2(b)$ is formed by all the children of the colleagues of b's parent that are well-separated from b.

List 3 $L_3(b)$ is empty if $b$ is a parent box, and consists of all descendan adjacent t themselves

List 4 $L_4(b) := \{$

List 5 $L_5(b)$ cons from b's p

[Carrier et al. '88]

Colleagues?

Observation: $c \in L_3(b)$ is separated from $b$ by a distance greater than or equal to the length of the side of $c$.

Observation: All boxes in $L_4(b)$ are childless and larger than $b$.

# Interaction Lists: Terminology

List 1 If $b$ is childless, $L_1(b)$ consists of $b$ and all childless boxes adjacent to $b$. If $b$ has children, $L_1(b)$ is empty.

List 2 $L_2(b)$ is formed by all the children of the colleagues of b's parent that are well-separated from b.

List 3 $L_3(b)$ is e[...] descendan[...] adjacent t[...] themselve[...]

List 4 $L_4(b) := \{$[...]

List 5 $L_5(b)$ con[...] from b's p[...]

[Carrier et al. '88]

Colleagues?

Observation: $c \in L_3(b)$ is separated from $b$ by a distance greater than or equal to the length of the side of $c$.

Observation: All boxes in $L_4(b)$ are childless and larger than $b$.

Algorithms to build lists?

# Interaction Lists: Terminology

List 1 If $b$ is childless, $L_1(b)$ consists of $b$ and all childless boxes adjacent to $b$. If $b$ has children, $L_1(b)$ is empty.

List 2 $L_2(b)$ is fo
of b's pare

List 3 $L_3(b)$ is e
descendan
adjacent t
themselve

List 4 $L_4(b) := \{$

List 5 $L_5(b)$ cons
from b's p

[Carrier et al. '88]

Colleagues?

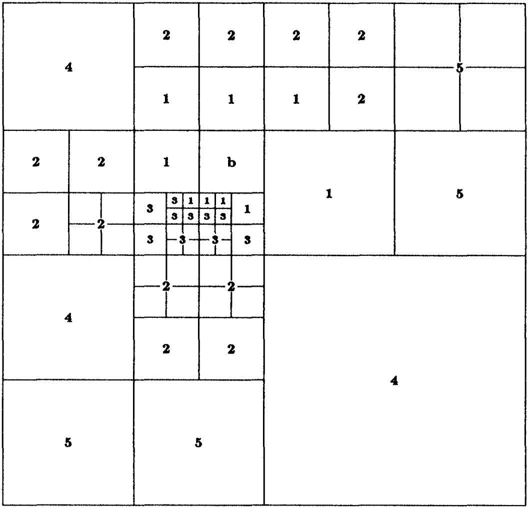
Observation: $c \in L_3(b)$ is separated from $b$ by a distance greater than or equal to the length of the side of $c$.

Observation: All boxes in $L_4(b)$ are childless and larger than $b$.

Algorithms to build lists?

Sometimes also called *UVWX* lists

# Structuring the Computation, part V




[Carrier et al. '88]

# Summing up

1. Build tree
2. Propagate multipoles upward
3. Evaluate neighbor boxes' sources directly ("list 1")
4. Translate separated siblings' ("list 2") mpoles to local
5. Evaluate sep. smaller mpoles ("list 3") at particles
6. Form locals for separated bigger mpoles ("list 4")
7. Propagate local expansions downward
8. Evaluate locals

# Summing up

1. Build tree
2. Propagate multipoles upward
3. Evaluate neighbor boxes' sources directly ("list 1")
4. Translate separated siblings' ("list 2") mpoles to local
5. Evaluate sep. smaller mpoles ("list 3") at particles
6. Form locals for separated bigger mpoles ("list 4")
7. Propagate local expansions downward
8. Evaluate locals

Most expensive step?

# What can serve as a ‘multipole’?

**Multipole:** More *concept* than formula

Lots of things are usable as “multipoles”.

- Taylor expansions
- Expansions that solve the PDE
  - Complex Taylor
  - Spherical harmonics
- ‘Skeleton’ particles
- Plane waves (directional)

# What can serve as a 'multipole'?

**Multipole:** More *concept* than formula

Lots of things are usable as "multipoles".

- Taylor expansions
- Expansions that solve the PDE
  - Complex Taylor
  - Spherical harmonics
- 'Skeleton' particles
- Plane waves (directional)

Why not use Taylor expansions?

# What can serve as a 'multipole'?

**Multipole:** More *concept* than formula

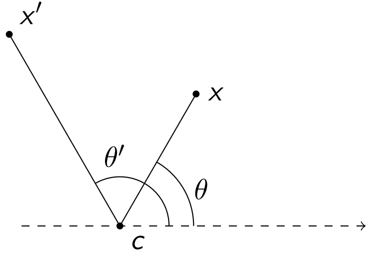
Lots of things are usable as "multipoles".

- Taylor expansions
- Expansions that solve the PDE
  - Complex Taylor
  - Spherical harmonics
- 'Skeleton' particles
- Plane waves (dire

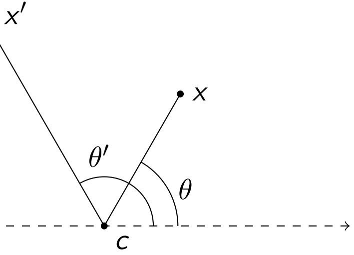
Why not use Taylor expansions?

Translation operators for each?

# Fourier-Bessel: Graf's addition theorem



# Fourier-Bessel: Graf's addition theorem

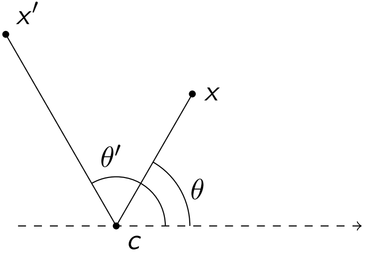



$$H_0^{(1)}(k|x-x'|) = \sum_{l=-\infty}^{\infty} H_l^{(1)}(k|x'-c|)e^{il\theta'} J_l(k|x-c|)e^{-il\theta}$$

Requires: $|x-c| < |x'-c|$

# Fourier-Bessel:

Like Taylor: Can be read two ways.




$$H_0^{(1)}(k|x-x'|) = \sum_{l=-\infty}^{\infty} H_l^{(1)}(k|x'-c|)e^{il\theta'} J_l(k|x-c|)e^{-il\theta}$$

Requires: $|x-c| < |x'-c|$

# Questions?

**?**