Lecture 6: Medians, Matrix Multiplication, Convex Hull

Business

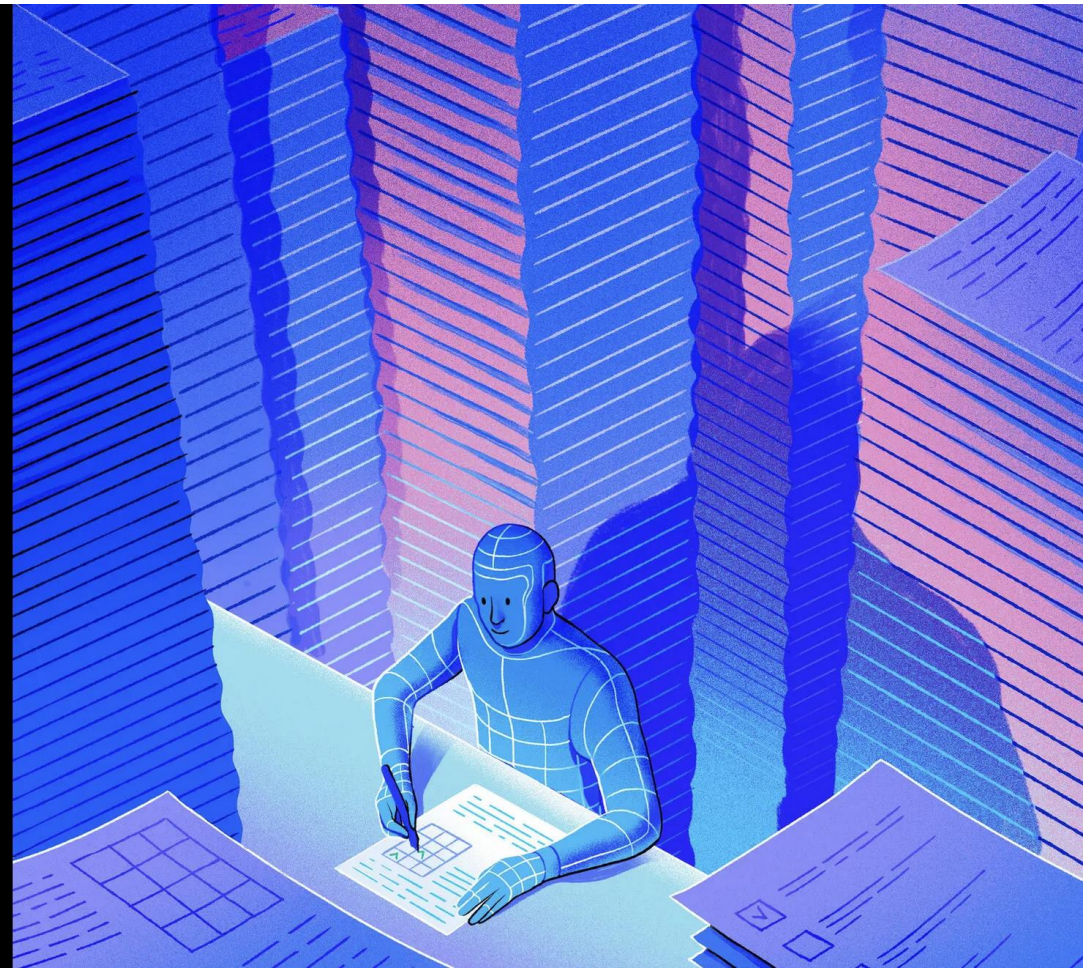
- hw 5 due wed at 11:59pm
- project 2 design experience due fri at 11:59pm
- project 2 due feb 10 at 11:59 pm
- reading for thu ch 3.1-3.2

Something Interesting

THE SHIFT

When A.I. Passes This Test, Look Out

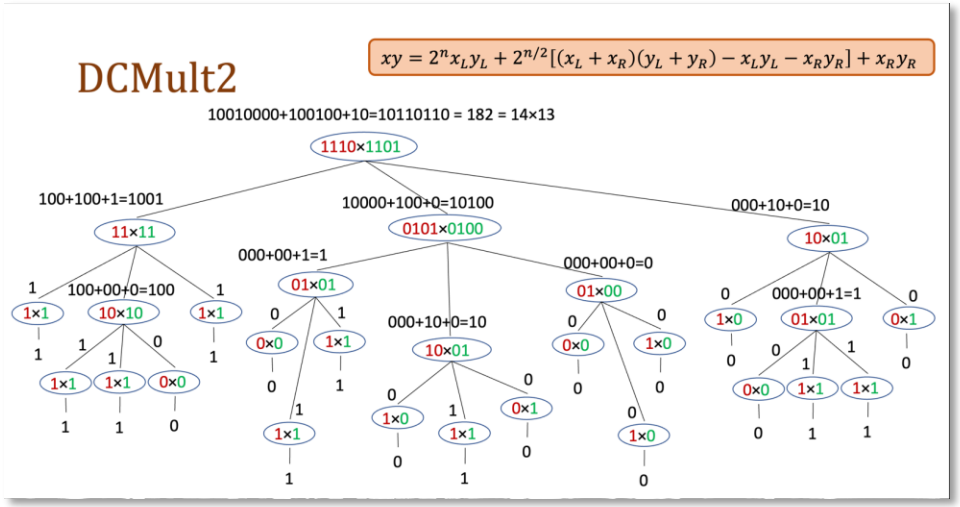
The creators of a new test called "Humanity's Last Exam" argue we may soon lose the ability to create tests hard enough for A.I. models.



https://www.nytimes.com/2025/01/23/technology/ai-test-humanitys-last-exam.html

Last Time

- divide and conquer
- multiplication
- master theorem



The Master Theorem

Theorem

if $T(n) = aT\left(\left\lceil \frac{n}{b} \right\rceil\right) + O(n^d)$ for some constants $a > 0, b > 1, d \geq 0$, then

$$T(n) = \begin{cases} O(n^d) & if\ 1 > \frac{a}{b^d} \\ O(n^d \log n) & if\ 1 = \frac{a}{b^d} \\ O(n^{\log_b a}) & if\ 1 < \frac{a}{b^d} \end{cases}$$

Today

- median
- matrix multiplication
- convex hull

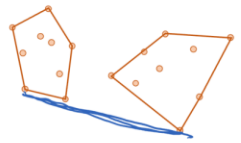
Matrix Multiplication: Divide and Conquer

$$XY = \begin{bmatrix} A & B \\ C & D \end{bmatrix} \begin{bmatrix} E & F \\ G & H \end{bmatrix} = \begin{bmatrix} P_5 + P_4 - P_2 + P_6 & P_1 + P_2 \\ P_3 + P_4 & P_1 + P_5 - P_3 - P_7 \end{bmatrix}$$

$P_1 = A(F - H)$
$P_2 = (A + B)H$
$P_3 = (C + D)E$
$P_4 = D(G - E)$
$P_5 = (A + D)(E + H)$
$P_6 = (B - D)(G + H)$
$P_7 = (A - C)(E + F)$

Convex Hull—Upper and Lower Tangents

FindUpperTangent(L, R)
  find rightmost point p in L and leftmost point q in R
  $temp = \text{line}(p, q)$
  $done = 0$
  while not done **do**
    $done = 1$
    while $temp$ is not upper tangent to L **do**
      $r \leftarrow p$'s counterclockwise neighbor
      $temp = \text{line}(r, q)$
      $p = r$
      $done = 0$
    while $temp$ is not upper tangent to R **do**
      $r \leftarrow q$'s clockwise neighbor
      $temp = \text{line}(p, r)$
      $q = r$
      $done = 0$
  return $temp$



Ungraded Quiz

- what are divide and conquer algorithms?
- what trick allows one to do multiplication in $O(n^{1.59})$ running time?
- what does the master theorem tell us?

Median: A Naïve Approach

Median(S)

sort(S) $O(n \log n)$

return $S\left[\left\lfloor \frac{\text{len}(S)}{2} \right\rfloor\right]$ $O(1)$ $O(\log n)$

time complexity: $O(n \log n)$ $O(n \log n)$

Median2

try it: Median2([8 3 4 9 5])

Median2(S)

    return Selection(S, $\lfloor \text{len}(S)/2 \rfloor$)

Selection(S, k)

    choose random pivot v

    $S_L \leftarrow S[i], \forall\, S[i] < v$

    $S_v \leftarrow S[i], \forall\, S[i] = v$

    $S_R \leftarrow S[i], \forall\, S[i] > v$

    if $k < |S_L|$ **then**

        return Selection(S_L, k)

    if $k < |S_L| + |S_v|$ **then**

        return v

    if $k \geq |S_L| + |S_v|$ **then**

        return Selection($S_R, k - |S_L| - |S_v|$)

Median2

Median2(S)

    return Selection(S, $\lfloor \text{len}(S)/2 \rfloor$)

Selection(S, k)

    choose random pivot v
    $S_L \leftarrow S[i], \forall\, S[i] < v$
    $S_v \leftarrow S[i], \forall\, S[i] = v$
    $S_R \leftarrow S[i], \forall\, S[i] > v$
    if $k < |S_L|$ **then**
        return Selection(S_L, k)
    if $k < |S_L| + |S_v|$ **then**
        return v
    if $k \geq |S_L| + |S_v|$ **then**
        return Selection($S_R, k - |S_L| - |S_v|$)

try it: Median2([8 3 4 9 5])

Selection([8 3 4 9 5], 2)
    $v = 4$
    $S_L \leftarrow [3]$
    $S_v \leftarrow [4]$
    $S_R \leftarrow [8\ 9\ 5]$
    return Selection([8 9 5], 0)

Selection([8 9 5], 0)
    $v = 9$
    $S_L \leftarrow [8\ 5]$
    $S_v \leftarrow [9]$
    $S_R \leftarrow []$
    return Selection([8 5], 0)

Selection([8 5], 0)
    $v = 5$
    $S_L \leftarrow []$
    $S_v \leftarrow [5]$
    $S_R \leftarrow [8]$
    return 5

Median2

$T(n) = 1T(\frac{n}{4/3}) + O(n)$

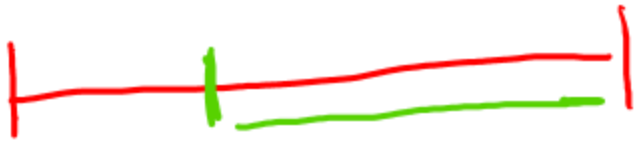
Median2(S)

    return Selection$(S, \lfloor \text{len}(S)/2 \rfloor)$

Selection(S, k)

    choose random pivot v
    $S_L \leftarrow S[i], \forall\, S[i] < v$
    $S_v \leftarrow S[i], \forall\, S[i] = v$
    $S_R \leftarrow S[i], \forall\, S[i] > v$
    if $k < |S_L|$ **then**
        return Selection(S_L, k)
    if $k < |S_L| + |S_v|$ **then**
        return v
    if $k \geq |S_L| + |S_v|$ **then**
        return Selection$(S_R, k - |S_L| - |S_v|)$

time complexity:



$\frac{a}{b^d} = \frac{1}{(4/3)^1} = \frac{3}{4} < 1$

$O(n)$

Matrix Multiplication

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \times \begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix} = \begin{bmatrix} (1 \cdot 5 + 2 \cdot 7) & (1 \cdot 6 + 2 \cdot 8) \\ (3 \cdot 5 + 4 \cdot 7) & (3 \cdot 6 + 4 \cdot 8) \end{bmatrix} = \begin{bmatrix} 19 & 22 \\ 43 & 50 \end{bmatrix}$$

Matrix Multiplication: Divide and Conquer

$$XY = \begin{bmatrix} A & B \\ C & D \end{bmatrix} \begin{bmatrix} E & F \\ G & H \end{bmatrix} = \begin{bmatrix} AE + BG & AF + BH \\ CE + DG & CF + DH \end{bmatrix}$$

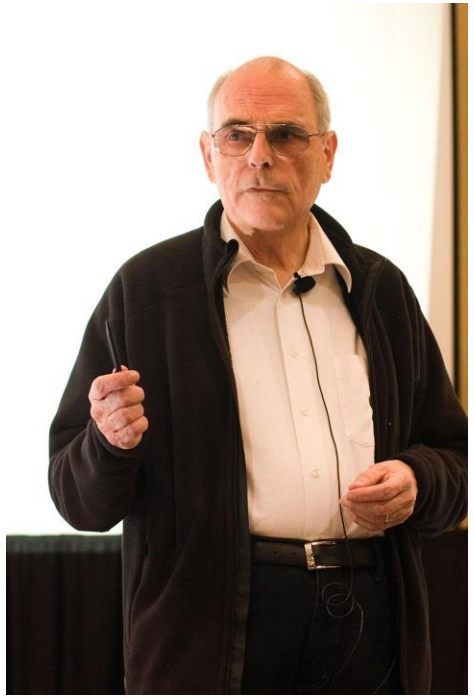
Matrix Multiplication: Divide and Conquer

$$XY = \begin{bmatrix} A & B \\ C & D \end{bmatrix} \begin{bmatrix} E & F \\ G & H \end{bmatrix} = \begin{bmatrix} P_5 + P_4 - P_2 + P_6 & P_1 + P_2 \\ P_3 + P_4 & P_1 + P_5 - P_3 - P_7 \end{bmatrix}$$

$$P_1 = A(F - H)$$
$$P_2 = (A + B)H$$
$$P_3 = (C + D)E$$
$$P_4 = D(G - E)$$
$$P_5 = (A + D)(E + H)$$
$$P_6 = (B - D)(G + H)$$
$$P_7 = (A - C)(E + F)$$

a nice video for more explanation:

https://www.youtube.com/watch?v=sZxjuT1kUd0&t=291s



Volker Strassen

Matrix Multiplication

A Refined Laser Method and Faster Matrix Multiplication

Josh Alman* Virginia Vassilevska Williams†

October 13, 2020

Abstract

The complexity of matrix multiplication is measured in terms of ω, the smallest real number such that two $n \times n$ matrices can be multiplied using $O(n^{\omega+\epsilon})$ field operations for all $\epsilon > 0$; the best bound until now is $\omega < 2.37287$ [Le Gall'14]. All bounds on ω since 1986 have been obtained using the so-called laser method, a way to lower-bound the 'value' of a tensor in designing matrix multiplication algorithms. The main result of this paper is a refinement of the laser method that improves the resulting value bound for most sufficiently large tensors. Thus, even before computing any specific values, it is clear that we achieve an improved bound on ω, and we indeed obtain the best bound on ω to date:

$$\omega < 2.37286.$$

The improvement is of the same magnitude as the improvement that [Le Gall'14] obtained over the previous bound [Vassilevska W.'12]. Our improvement to the laser method is quite general, and we believe it will have further applications in arithmetic complexity.

Matrix Multiplication

arXiv > cs > arXiv:2210.04045

Search...

Help | Advanced

Computer Science > Symbolic Computation

[Submitted on 8 Oct 2022 (v1), last revised 13 Oct 2022 (this version, v3)]

The FBHHRBNRSSSHK-Algorithm for Multiplication in $\mathbb{Z}_2^{5\times5}$ is still not the end of the story

Manuel Kauers, Jakob Moosbauer

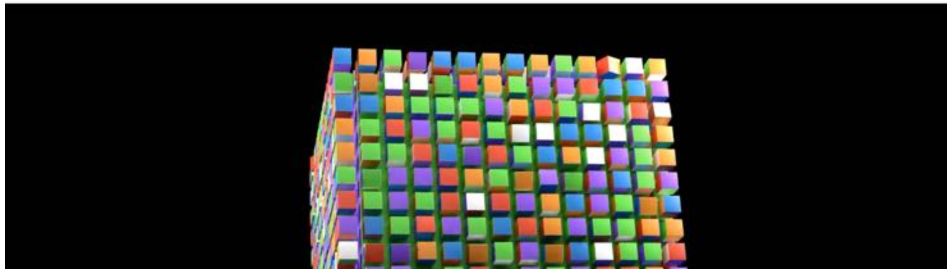
In response to a recent Nature article which announced an algorithm for multiplying 5×5-matrices over $\mathbb{Z}_2$ with only 96 multiplications, two fewer than the previous record, we present an algorithm that does the job with only 95 multiplications.

NEURAL NETWORKS

AI Reveals New Possibilities in Matrix Multiplication

8

Inspired by the results of a game-playing neural network, mathematicians have been making unexpected advances on an age-old math problem.



Divide and Conquer Speedup

- speed-up happens when we can find short-cuts during partition/merge that can be taken because of the divide and conquer
  - **sort**: fast merge with already sorted sub-lists
  - **convex hull**: fast merge of ordered sub-hulls (and dropping internal points while merging)
  - **multiply/matrix multiply**: can use the "less multiplies trick" at each level of merge
  - **quicksort**: partitioning is only $O(n)$ at each level and leads to final sorted list
  - **binary search/selection**: can discard ~half of the data at each level (i.e. just one subtask)
- master theorem tells us the complexity

Convex Combinations

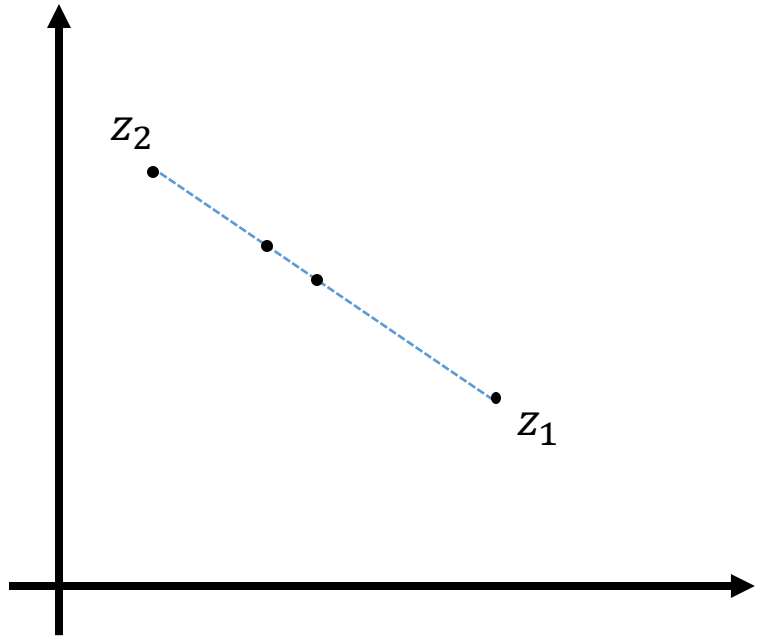
given a finite set of points $z_1, \ldots, z_n \in \mathbb{R}^m$, a point $z \in \mathbb{R}^m$ is a convex combination of these points if

$$z = \sum_j t_j z_j \,,$$

where $t_j \geq 0, \forall j$ and $\sum_j t_j = 1$

$t_1 = \frac{1}{3} \quad t_2 = \frac{2}{3}$

in $\mathbb{R}^2$:



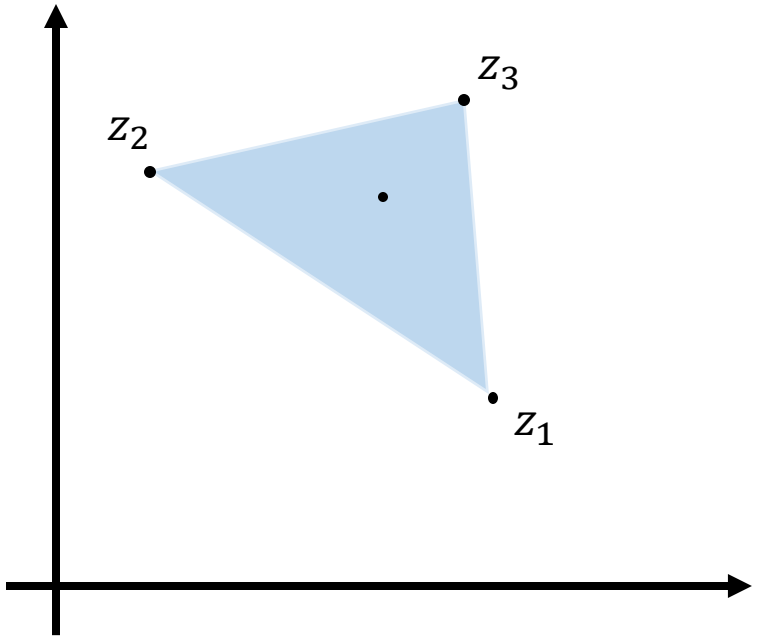
Convex Combinations

given a finite set of points $z_1, \ldots, z_n \in \mathbb{R}^m$, a point $z \in \mathbb{R}^m$ is a convex combination of these points if

$$z = \sum_j t_j z_j \,,$$

where $t_j \geq 0, \forall j$ and $\sum_j t_j = 1$

in $\mathbb{R}^2$:



Convex Sets

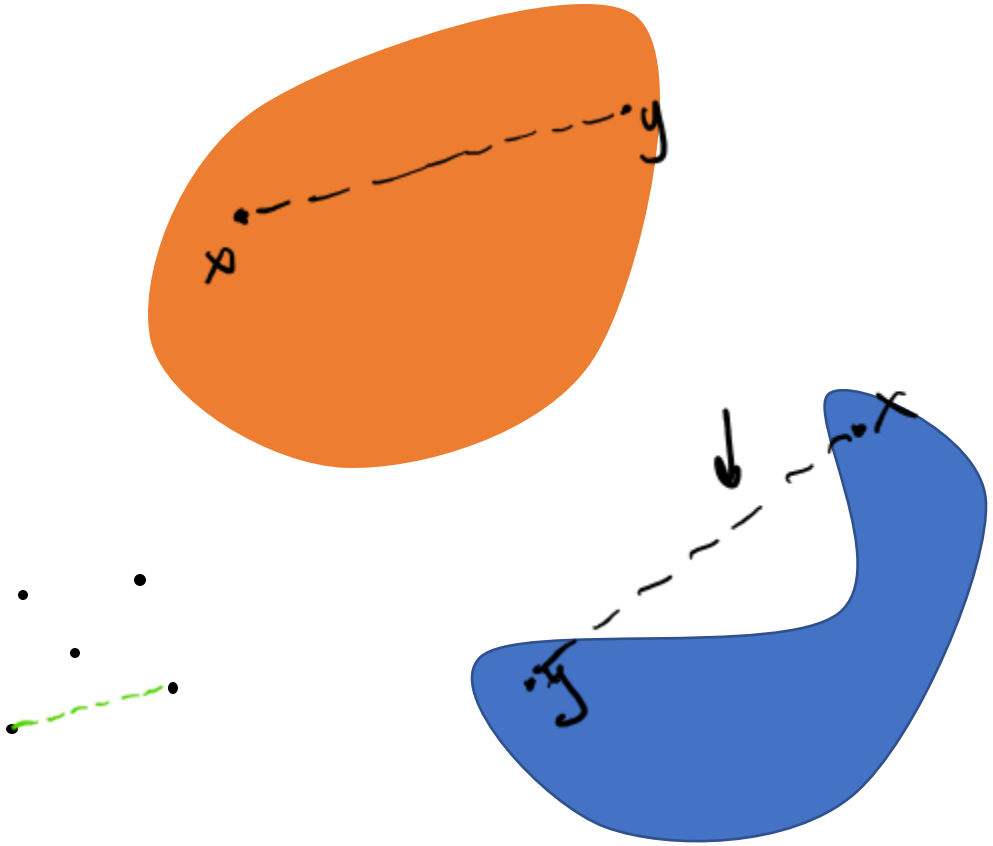
given a finite set of points $z_1, \ldots, z_n \in \mathbb{R}^m$, a point $z \in \mathbb{R}^m$ is a convex combination of these points if

$$z = \sum_j t_j z_j \,,$$

where $t_j \geq 0, \forall j$ and $\sum_j t_j = 1$

a subset S of $\mathbb{R}^m$ is convex if, for every $x, y \in S$ and t such that $0 < t < 1$, $tx + (1-t)y \in S$

consider the following sets of points in $\mathbb{R}^2$



Convex Hull

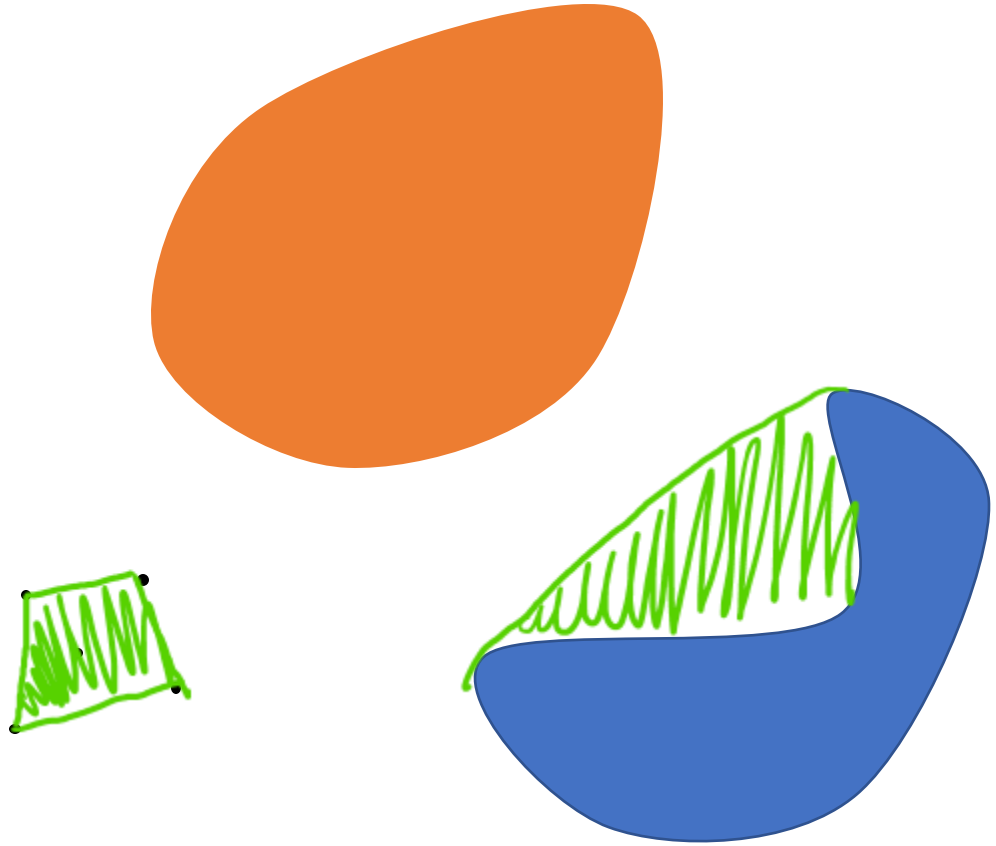
given a finite set of points $z_1, \ldots, z_n \in \mathbb{R}^m$, a point $z \in \mathbb{R}^m$ is a convex combination of these points if

$$z = \sum_j t_j z_j \,,$$

where $t_j \geq 0, \forall j$ and $\sum_j t_j = 1$

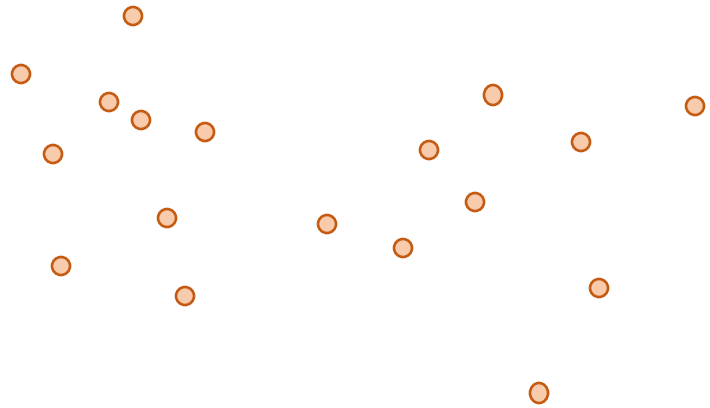
a subset S of $\mathbb{R}^m$ is convex if, for every $x, y \in S$ and t such that $0 < t < 1$, $tx + (1-t)y \in S$

the convex hull of $S \subseteq \mathbb{R}^m$, denoted $\mathrm{conv}(S)$, is the intersection of all convex sets containing S. one can think of it as the "smallest" convex set containing S.



Convex Hull: Brute Force

BruteForceCH(P)
  $hull \leftarrow \{\}$
  for $p_1, p_2 \in P$ **do**
    find line $y = mx + b$ that contains p_1 and p_2
    $count = 0$
    for $q \in P$ **do**
      if $y_q > mx_q + b$ **then**
        $count = count + 1$
    if $count = |P|$ or $count = 0$ **then**
      $hull = hull \cup \{(p_1, p_2)\}$
  return $hull$



Convex Hull: Divide and Conquer

DC(x)

$n \leftarrow \text{size}(x)$

if $n < \theta$ **then**

return adhoc(x)

decompose x into a subtasks $x_1, \ldots, x_a$ of size n/b

for $i = 1$ to a **do**

$y_i \leftarrow \text{DC}(x_i)$

$y \leftarrow \text{Combine}(y_i)$

return y

Convex Hull: Divide and Conquer

$\mathrm{DC}(x)$

$n \leftarrow \mathrm{size}(x)$

if $n < \theta$ **then**

return $\mathrm{adhoc}(x)$

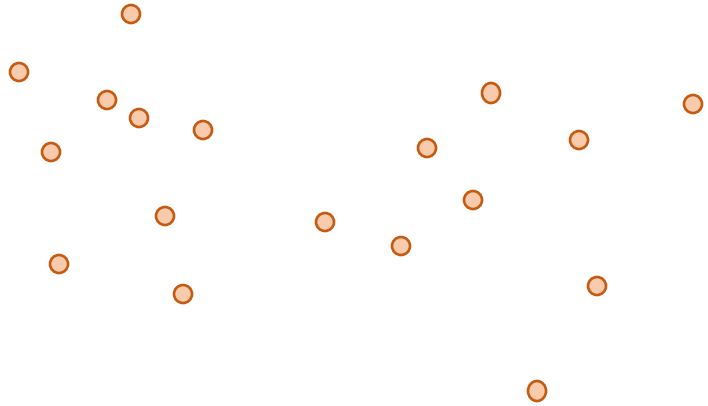
decompose x into a subtasks $x_1, \ldots, x_a$ of size n/b

for $i = 1$ to a **do**

$y_i \leftarrow \mathrm{DC}(x_i)$

$y \leftarrow \mathrm{Combine}(y_i)$

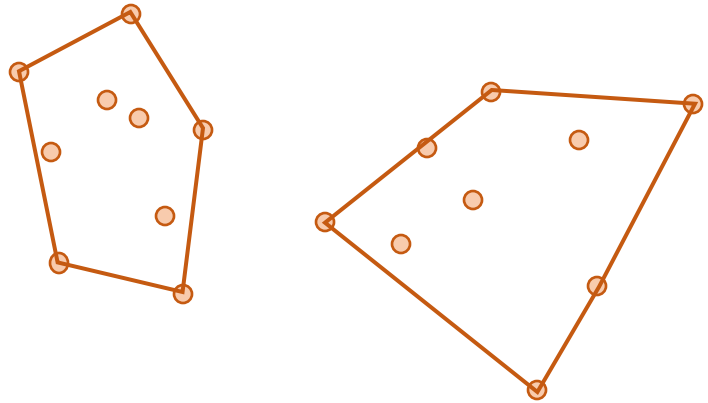
return y



Convex Hull—Merging Sub-hulls

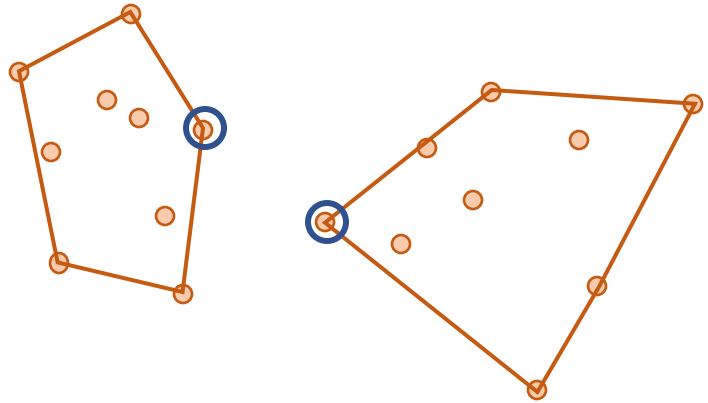
Convex Hull—Upper and Lower Tangents

FindUpperTangent(L, R)
  find rightmost point $p \in L$ and leftmost point $q \in R$
  $temp = \text{line}(p, q)$
  $done = \text{False}$
  while not $done$ **do**
    $done = \text{True}$
    while $temp$ is not upper tangent to L **do**
      $r \leftarrow p'$s counterclockwise neighbor
      $temp = \text{line}(r, q)$
      $p = r$
      $done = \text{False}$
    while $temp$ is not upper tangent to R **do**
      $r \leftarrow q'$s clockwise neighbor
      $temp = \text{line}(p, r)$
      q $= r$
      $done = \text{False}$
  return $temp$



Convex Hull—Upper and Lower Tangents

FindUpperTangent(L, R)
  find rightmost point $p \in L$ and leftmost point $q \in R$
  $temp = \text{line}(p, q)$
  $done = \text{False}$
  while not $done$ **do**
    $done = \text{True}$
    while $temp$ is not upper tangent to L **do**
      $r \leftarrow p'$s counterclockwise neighbor
      $temp = \text{line}(r, q)$
      $p = r$
      $done = \text{False}$
    while $temp$ is not upper tangent to R **do**
      $r \leftarrow q'$s clockwise neighbor
      $temp = \text{line}(p, r)$
      q $= r$
      $done = \text{False}$
  return $temp$

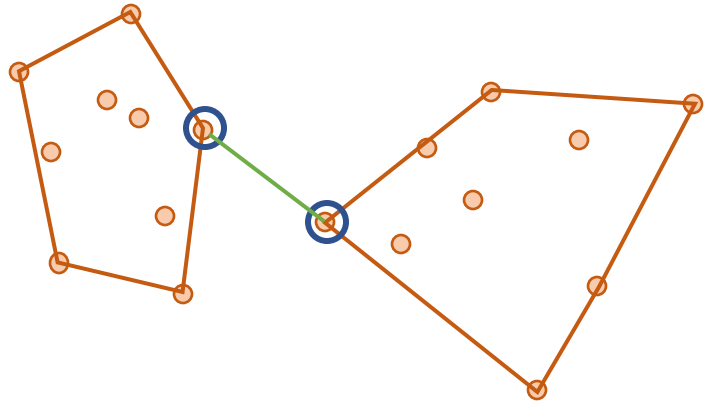


right most point in left sub-hull

left most point in left sub-hull

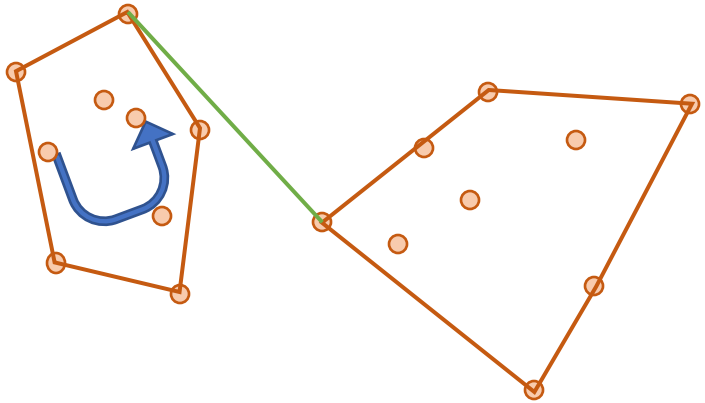
Convex Hull—Upper and Lower Tangents

FindUpperTangent(L, R)
  find rightmost point $p \in L$ and leftmost point $q \in R$
  $temp = \text{line}(p, q)$
  $done = \text{False}$
  while not $done$ **do**
    $done = \text{True}$
    while $temp$ is not upper tangent to L **do**
      $r \leftarrow p'$s counterclockwise neighbor
      $temp = \text{line}(r, q)$
      $p = r$
      $done = \text{False}$
    while $temp$ is not upper tangent to R **do**
      $r \leftarrow q'$s clockwise neighbor
      $temp = \text{line}(p, r)$
      q $= r$
      $done = \text{False}$
  return $temp$



Convex Hull—Upper and Lower Tangents

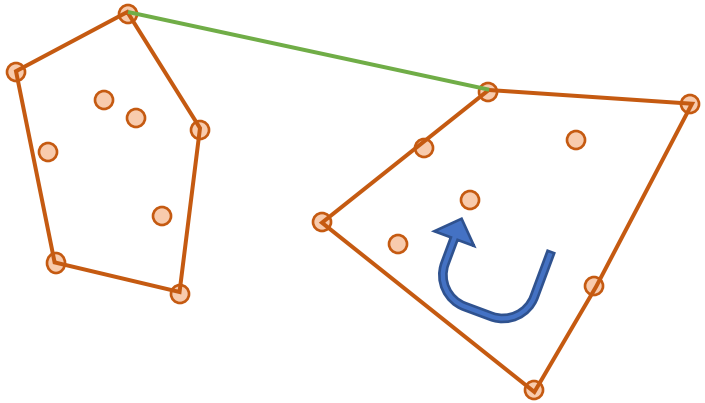
FindUpperTangent(L, R)
  find rightmost point $p \in L$ and leftmost point $q \in R$
  $temp = \text{line}(p, q)$
  $done = \text{False}$
  while not $done$ **do**
    $done = \text{True}$
    while $temp$ is not upper tangent to L **do**
      $r \leftarrow p'$s counterclockwise neighbor
      $temp = \text{line}(r, q)$
      $p = r$
      $done = \text{False}$
    while $temp$ is not upper tangent to R **do**
      $r \leftarrow q'$s clockwise neighbor
      $temp = \text{line}(p, r)$
      q $= r$
      $done = \text{False}$
  return $temp$



Counter clockwise for upper in left sub-hull

Convex Hull—Upper and Lower Tangents

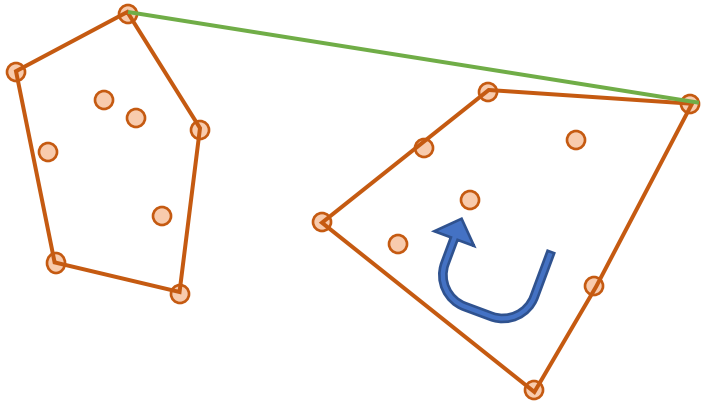
FindUpperTangent(L, R)
  find rightmost point $p \in L$ and leftmost point $q \in R$
  $temp = \text{line}(p, q)$
  $done = \text{False}$
  while not $done$ **do**
    $done = \text{True}$
    while $temp$ is not upper tangent to L **do**
      $r \leftarrow p'$s counterclockwise neighbor
      $temp = \text{line}(r, q)$
      $p = r$
      $done = \text{False}$
    while $temp$ is not upper tangent to R **do**
      $r \leftarrow q'$s clockwise neighbor
      $temp = \text{line}(p, r)$
      q $= r$
      $done = \text{False}$
  return $temp$



Clockwise for upper in right sub-hull

Convex Hull—Upper and Lower Tangents

FindUpperTangent(L, R)
  find rightmost point $p \in L$ and leftmost point $q \in R$
  $temp = \text{line}(p, q)$
  $done = \text{False}$
  while not $done$ **do**
    $done = \text{True}$
    while $temp$ is not upper tangent to L **do**
      $r \leftarrow p'$s counterclockwise neighbor
      $temp = \text{line}(r, q)$
      $p = r$
      $done = \text{False}$
    while $temp$ is not upper tangent to R **do**
      $r \leftarrow q'$s clockwise neighbor
      $temp = \text{line}(p, r)$
      q $= r$
      $done = \text{False}$
  return $temp$



Clockwise for upper in right sub-hull

Convex Hull Notes

- maintain clockwise (or counter clockwise) ordering when merging (natural if start that way).
- handle the base case properly
  - get started with appropriately ordered hulls
- need to be careful when accessing your hull data structure since it is really a circular list. if using an array then make sure indexes properly change between the 0 element and the last element when you are moving either clockwise or counterclockwise through the array.
- review project description