Euclidean Vs. Non-Euclidean

- A *Euclidean space* has some number of real-valued dimensions and "dense" points.
  - There is a notion of "average" of two points.
  - A *Euclidean distance* is based on the locations of points in such a space.
- A *Non-Euclidean distance* is based on properties of points, but not their "location" in a space.

Axioms of a Distance Measure

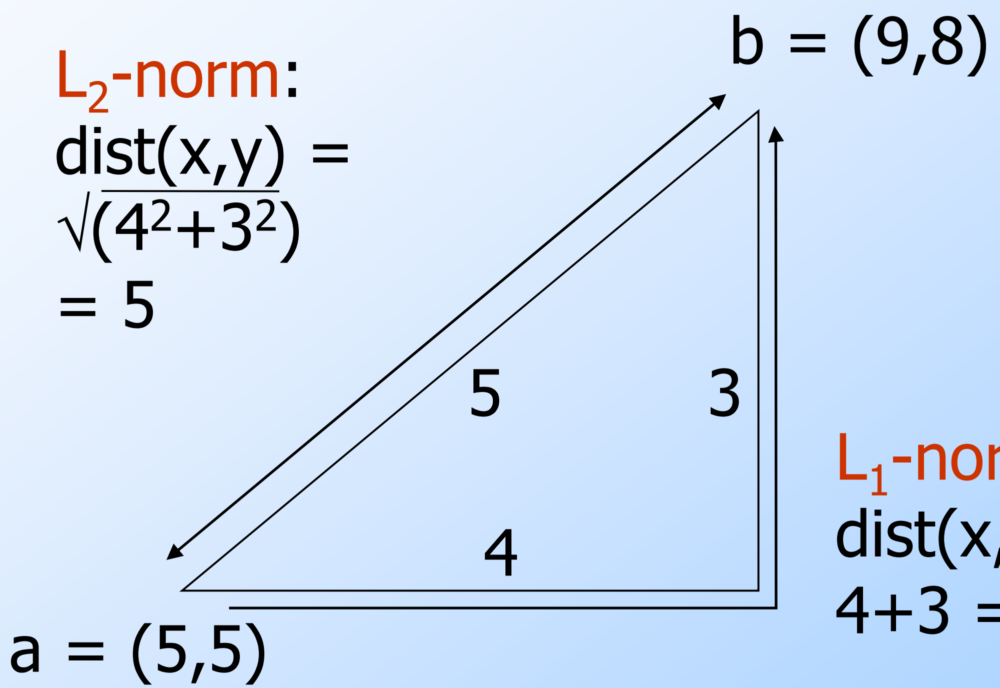
- *d* is a *distance measure* if it is a function from pairs of points to real numbers such that:
  1. $d(x,y) \geq 0$.
  2. $d(x,y) = 0$ iff $x = y$.
  3. $d(x,y) = d(y,x)$.
  4. $d(x,y) \leq d(x,z) + d(z,y)$ (*triangle inequality*).

Some Euclidean Distances

- *L_2 norm* : d(x,y) = square root of the sum of the squares of the differences between *x* and *y* in each dimension.
  - The most common notion of "distance."
- *L_1 norm* : sum of the differences in each dimension.
  - *Manhattan distance* = distance if you had to travel along coordinates only.

Examples of Euclidean Distances

L_2-norm:
$\text{dist}(x,y) = \sqrt{(4^2+3^2)} = 5$



L_1-norm:
$\text{dist}(x,y) = 4+3 = 7$

Another Euclidean Distance

- L_∞ *norm* : d(x,y) = the maximum of the differences between x and y in any dimension.
- Note: the maximum is the limit as n goes to ∞ of the L_n norm: what you get by taking the n^{th} power of the differences, summing and taking the n^{th} root.

Non-Euclidean Distances

- *Jaccard distance* for sets = 1 minus Jaccard similarity.
- *Cosine distance* = angle between vectors from the origin to the points in question.
- *Edit distance* = number of inserts and deletes to change one string into another.
- *Hamming Distance* = number of positions in which bit vectors differ.

Jaccard Distance for Sets (Bit-Vectors)

- Example: p_1 = 10111; p_2 = 10011.
- Size of intersection = 3; size of union = 4, Jaccard similarity (not distance) = 3/4.
- d(x,y) = 1 – (Jaccard similarity) = 1/4.

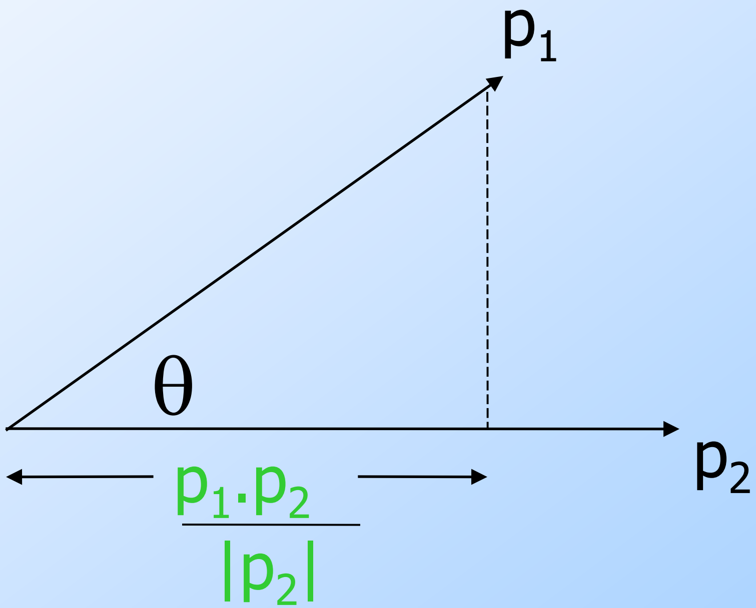
Why J.D. Is a Distance Measure

- $d(x,x) = 0$ because $x \cap x = x \cup x$.
- $d(x,y) = d(y,x)$ because union and intersection are symmetric.
- $d(x,y) \geq 0$ because $|x \cap y| \leq |x \cup y|$.
- $d(x,y) \leq d(x,z) + d(z,y)$ trickier – requires lsh to be covered next.

Cosine Distance

- Think of a point as a vector from the origin (0,0,…,0) to its location.
- Two points' vectors make an angle, whose cosine is the normalized dot-product of the vectors: $p_1.p_2/|p_2||p_1|$.
  - Example: $p_1 = 00111$; $p_2 = 10011$.
  - $p_1.p_2 = 2$; $|p_1| = |p_2| = \sqrt{3}$.
  - $\cos(\theta) = 2/3$; θ is about 48 degrees.

Cosine-Measure Diagram



$d\ (p_1, p_2) = \theta = \text{arccos}(p_1.p_2/|p_2||p_1|)$

Why C.D. Is a Distance Measure

- d(x,x) = 0 because arccos(1) = 0.
- d(x,y) = d(y,x) by symmetry.
- d(x,y) $\geq$ 0 because angles are chosen to be in the range 0 to 180 degrees.
- Triangle inequality: physical reasoning. If I rotate an angle from *x* to *z* and then from *z* to *y*, I can't rotate less than from *x* to *y*.

Edit Distance

- The *edit distance* of two strings is the number of inserts and deletes of characters needed to turn one into the other. Equivalently:
- $d(x,y) = |x| + |y| - 2|LCS(x,y)|$.
  - LCS = *longest common subsequence* = any longest string obtained both by deleting from *x* and deleting from *y*.

Example: LCS

- $x = abcde$; $y = bcduve$.
- Turn x into y by deleting *a*, then inserting *u* and *v* after *d*.
  - Edit distance = 3.
- Or, LCS(x,y) = *bcde*.
- Note: $|x| + |y| - 2|LCS(x,y)| = 5 + 6 - 2*4 = 3$ = edit distance.

Why Edit Distance Is a Distance Measure

- d(x,x) = 0 because 0 edits suffice.
- d(x,y) = d(y,x) because insert/delete are inverses of each other.
- d(x,y) $\geq$ 0: no notion of negative edits.
- Triangle inequality: changing *x* to *z* and then to *y* is one way to change *x* to *y*.

Variant Edit Distances

- Allow insert, delete, and *mutate*.
  - Change one character into another.
- Minimum number of inserts, deletes, and mutates also forms a distance measure.
- Ditto for any set of operations on strings.
  - Example: substring reversal OK for DNA sequences

Hamming Distance

- *Hamming distance* is the number of positions in which bit-vectors differ.
- Example: $p_1 = 10101$; $p_2 = 10011$.
- $d(p_1, p_2) = 2$ because the bit-vectors differ in the 3rd and 4th positions.

Why Hamming Distance Is a Distance Measure

- d(x,x) = 0 since no positions differ.
- d(x,y) = d(y,x) by symmetry of "different from."
- d(x,y) $\geq$ 0 since strings cannot differ in a negative number of positions.
- Triangle inequality: changing *x* to *z* and then to *y* is one way to change *x* to *y*.

- Hamming distance

- ■ Number of positions in which two strings (of equal length) differ
- □ Minimum number of substitutions required to change one
- string into the other
- □ Minimum number of errors that could have transformed one
- string into the other.
- ■ Used mostly for binary numbers and to measure communication

Edit distances

■ Compare two strings based on individual characters
■ Minimal number of edits required to transform one string into the other.
□ Edits: Insert, Delete, Replace (and Match)
□ Alternative: Smallest edit cost
□ Give different cost to different types of edits
□ Give different cost to different letters
■ Naive approach: editdistance(Jones,Johnson)
□ DDDDDIIIIIII = 12
□ But: Not minimal!
■ Levenshtein distance: Basic form
□ Each edit has cost 1