Dictionaries and Distances

Grace Ngai

Much of this course and lots of the lecture notes were inspired by or derived from Brown University's CS931. Our thanks go to Prof Shriram Krishnamurthi and Hammurabi Mendes for their kind permission in allowing us to use their materials.

The Big Picture

Define the problem

Find the Data

Build a concordance of a text
- *Locations* of words
- *Frequencies* of words

Write the Instructions

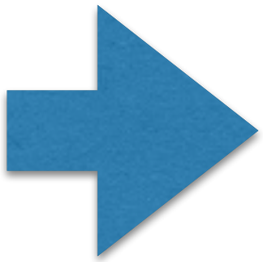
Python

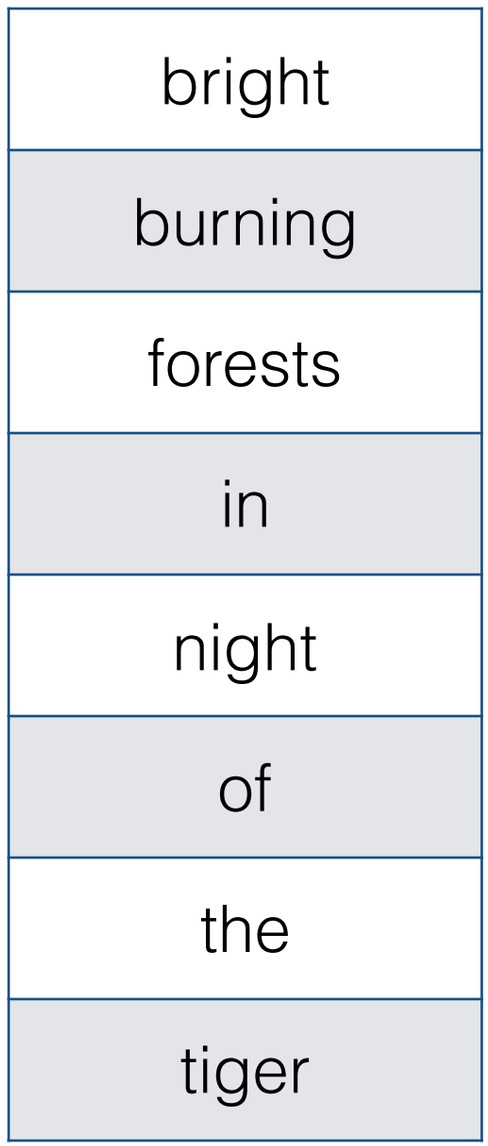
Word frequencies over time
Author of texts
Bias of authors (e.g. liberal media bias)
Worldwide trends (e.g. oil vs. gold)
…

Solution

After HW2-4

tiger tiger burning bright
in the forests of the night





Build a Concordance

- All the words in a text
- Locations of the words
- Frequencies of the words
- So far: Build a *vocabulary* of all the words in a text

Frequencies of Words

tiger tiger burning bright
in the forests of the night

bright	1
burning	1
forests	1
in	1
night	1
of	1
the	2
tiger	2

WordFrequency Function

Input

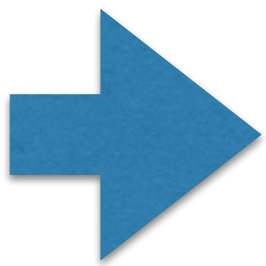
tiger tiger burning bright
in the forests of the night

bright	1
burning	1
forests	1
in	1
night	1
of	1
the	2
tiger	2

WordFrequency Function

Input

tiger tiger burning bright
in the forests of the night



Output

bright	1
burning	1
forests	1
in	1
night	1
of	1
the	2
tiger	2

WordFrequency Function

Input

tiger tiger burning bright
in the forests of the night

Output

bright	1
burning	1
forests	1
in	1
night	1
of	1
the	2
tiger	2

We could do this with lists… but it would be difficult

Python Data Structures

- A *data structure* is simply a way to store information.

- The simplest kinds of data structure are variables.
    - One name, one value

- Lists are another kind of data structure
    - *Linear* organization (one name, lots of values, indexed by a range of integers.)

Getting Word Frequencies

tiger tiger burning bright in the forests of the night

word	frequency
bright	1
burning	1
forests	1
in	1
night	1
of	1
the	2
tiger	2

Getting Word Frequencies

tiger tiger burning bright in the forests of the night

word	frequency
bright	1
burning	1
forests	1
in	1
night	1
of	1
the	2
tiger	2

Values

Keys

We want to remember each **word** that appeared in the text, and also its **frequency**

Getting Word Frequencies

tiger tiger burning bright in the forests of the night

word	frequency
bright	1
burning	1
forests	1
in	1
night	1
of	1
the	2
tiger	2

Values

Keys

To use this data, we want a function, **`wordFreq`**, that returns the frequency of a word.
e.g. **`wordFreq("bright")`** will return 1

We *could* do this with lists. But it would be cumbersome.

A New Data Structure: Python Dictionaries

- Dictionaries link *keys* with *values*
  - e.g. The word is the key, and the definition is the value.
- In Python,
  - Dictionary keys can be numbers, strings, etc
  - Values can be *any* kind of data

e•merge (ĭ-mûrj′) *v.* **e•merged, e•merg•ing.** **1.** To rise up or come forth into view; appear. **2.** To come into existence. **3.** To become known or evident. [Lat. *emergere.*] **—e•mer′gence** *n.* **—e•mer′gent** *adj.*

e•mer•gen•cy (ĭ-mûr′jən-sē) *n., pl.* **-ies.** An unexpected situation or occurrence that demands immediate attention.

e•mer•i•tus (ĭ-mĕr′ĭ-təs) *adj.* Retired but retaining an honorary title: *a professor emeritus.* [Lat., p.p. of *emereri,* to earn by service.]

em•er•y (ĕm′ə-rē, ĕm′rē) *n.* A fine-grained impure corundum used for grinding and polishing. [< Gk *smuris.*]

e•met•ic (ĭ-mĕt′ĭk) *adj.* Causing vomiting. [< Gk. *emein,* to vomit.] **—e•met′ic,** *n.*

–emia *suff.* Blood: *leukemia.* [< Gk. *haima,* blood.]

em•i•grate (ĕm′ĭ-grāt′) *v.* **-grat•ed, -grat•ing.** To leave one country or region to settle in another. [Lat. *emigrare.*] **—em′i•grant** *n.* **—em′i•gra′tion** *n.*

é•mi•gré (ĕm′ĭ-grā′) *n.* An emigrant, esp. a refugee from a revolution. [Fr.]

em•i•nence (ĕm′ə-nəns) *n.* **1.** a position of great distinction or superiority. **2.** A rise or elevation of ground; hill.

em•i•nent (ĕm′ə-nənt) *adj.* **1.** Outstanding, as in reputation; distinguished. **2.** Towering above others; projecting. [< Lat. *eminēre,* to stand out.] **—em′i•nent•ly** *adv.*

em•phat•ic (ĕm-făt′ĭk) *adj.* Expressed or performed with emphasis. [< Gk. *emphatikos.*]**—em•phat′i•cal•ly** *adv.*

em•phy•se•ma (ĕm′fi-sē′mə) *n.* A disease in which the air sacs of the lungs lose their elasticity, resulting in an often severe loss of breathing ability. [< Gk. *emphusēma.*]

em•pire (ĕm′pīr′) *n.* **1.** A political unit, usu. larger than a kingdom and often comprising a number of territories or nations, ruled by a single central authority. **2.** Imperial dominion, power, or authority. [<Lat. *imperium.*]

em•pir•i•cal (ĕm-pîr′i-kəl) *adj.* Also **em•pir•ic** (-pir′ik). **1.** Based on observation or experiment. **2.** Relying on practical experience rather than theory. [<Gk. *empeirikos,* experienced.] **—em•pir′i•cal•ly** *adv.*

em•pir•i•cism (ĕm-pîr′ĭ-sĭz′əm) *n.* **1.** The view that experience, esp. of the senses, is the only source of knowledge. **2.** The employment of empirical methods, as in science.—**em•pir′i•cist** *n.*

em•place•ment (ĕm-plās′mənt) *n.* **1.** A prepared position for guns within a fortification. **2.** Placement. [Fr.]

em•ploy (ĕm-ploi′) *v.* **1.** To engage or use the services of. **2.** To put to service; use. **3.** To devote or apply (one's time or energies) to an activity. —*n.* Employment. [< Lat. *implicare,* to involve.] **—em•ploy′a•ble** *adj.*

em•ploy•ee (ĕm-ploi′ē, ĕm′ploi-ē′) *n.* Also **em•ploy•e.** One who works for another.

ă pat ā pay â care ä father ĕ pet ē be ĭ pit ī tie î pier ŏ pot ō toe ô paw, for oi noise o͝o took o͞o boot ou out th thin *th* this ŭ cut û urge yoo abuse zh vision ə about, item, edible, gallop, circus

Python Dictionaries

Key Type	Value Type	Example Key	Example Value

Python Dictionaries

Key Type	Value Type	Example Key	Example Value
String	Integer	“bright”	1
String	Integer	“tiger"	2

Python Dictionaries

Key Type	Value Type	Example Key	Example Value
String	Integer	“bright”	1
String	Integer	“tiger"	2
String	String	“Grace"	“2766-7279”
String	String	“Chan Tai Man”	“15000001d"

Python Dictionaries

Key Type	Value Type	Example Key	Example Value
String	Integer	“bright”	1
String	Integer	“tiger"	2
String	String	“Grace"	“2766-7279”
String	String	“Chan Tai Man”	“15000001d"
Float	String	1.0	“one point oh”
Float	String	3.1415	“pi”

Python Dictionaries

Key Type	Value Type	Example Key	Example Value
String	Integer	“bright”	1
String	Integer	“tiger"	2
String	String	“Grace"	“2766-7279”
String	String	“Chan Tai Man”	“15000001d"
Float	String	1.0	“one point oh”
Float	String	3.1415	“pi”
String	List	“tiger"	[0, 1]

Using Python Dictionaries

tiger tiger burning bright in the forests of the night

word	frequency
bright	1
burning	1
forests	1
in	1
night	1
of	1
the	2
tiger	2

```
>>> freq = {}
>>> freq
{}
>>>
```

Initializes a dictionary

Using Python Dictionaries

tiger tiger burning bright in the forests of the night

word	frequency
bright	1
burning	1
forests	1
in	1
night	1
of	1
the	2
tiger	2

```
>>> freq = {}
>>> freq
{}
>>> freq["bright"] = 1
>>> freq
{"bright": 1}
>>>
```

Insert a new key
Key = “bright”
Value = 1

Using Python Dictionaries

tiger tiger burning bright in the forests of the night

word	frequency
bright	1
burning	1
forests	1
in	1
night	1
of	1
the	2
tiger	2

```
>>> freq = {}
>>> freq
{}
>>> freq[“bright”] = 1
>>> freq
{“bright”: 1}
>>> freq[“tiger”] = 2
>>> freq
{“tiger”: 2, “bright”: 1}
```

Insert a new key
Key = “tiger”
Value = 2

Using Python Dictionaries

tiger tiger burning bright in the forests of the night

word	frequency
bright	1
burning	1
forests	1
in	1
night	1
of	1
the	2
tiger	2

```
>>> freq = {}
>>> freq
{}
>>> freq[“bright”] = 1
>>> freq
{“bright”: 1}
>>> freq[“tiger”] = 2
>>> freq
{“tiger”: 2, “bright”: 1}
>>> freq[“tiger”]
2
>>> freq[“bright”]
1
>>>
```

Retrieve (look up) a value using the key

Using Python Dictionaries

tiger tiger burning bright in the forests of the night

word	frequency
bright	1
burning	1
forests	1
in	1
night	1
of	1
the	2
tiger	2

```
>>> freq2 = {“tiger” : 2,
“burning” : 1}
>>> freq2
{‘tiger’: 2, ‘burning’: 1}
```

Initializing a new dictionary with 2 key-value pairs

Using Python Dictionaries

tiger tiger burning bright in the forests of the night

word	frequency
bright	1
burning	1
forests	1
in	1
night	1
of	1
the	2
tiger	2

```
>>> freq2 = {“tiger” : 2,
“burning” : 1}
>>> freq2
{‘tiger’: 2, ‘burning’: 1}
>>> freq2[“tiger”] =
freq2[“tiger”] + 1
>>> freq2
{‘tiger’: 3, ‘burning’: 1}
>>>
```

Update the value of a key (or assign a new value to a key)

Things we can do with dictionaries

Function	Input	Output	Example
`keys()`	None	All the keys in the dictionary in a list	`>>> freq.keys()` `['the', 'cat']`
`values()`	None	All the values in the dictionary in a list	`>>> freq.values()` `[2, 1]`
`<key> in <dict>`	key	Boolean (True if the dictionary has that key)	`>>> "cat" in freq` `True`

`pop(<key>)`	key	The value linked to the key. Deletes the key. (Will return an error if the key doesn't exist!)	`>>> freq`
			`["the" : 2; "cat" : 1]`
			`>>> freq.pop("cat")`
			`1`
			`>>> freq`
			`["the" : 2]`

Act 2-5, Tasks 1 and 2

Calculating Similarities

- Our final problem will be to calculate the similarities between legislators (again!)

- But this time we will use the words that they spoke during LegCo, rather than their votes.

- Suppose that we have all the speeches that the legislators made during LegCo hearings, one file for each legislator

Sample of Speech

- Converted to lower-case and the punctuation removed

```
president a point of order mr wong kwokhing said just now that  apart from
mr lee cheukyan there are four members   president before reading out my
urgent question i have to put up a strong protest against mr wong kwokhings
groundless impugning of members motives in raising questions and denounce mr
wong kwokhings shameful act of smearing president i will now read out my
urgent question it has been reported that on the rd and th of this month
some members of the public who participated in the occupy central movement
and gathered in mong kok as well as journalists covering the activities were
assaulted and injured posing a serious threat to their personal safety in
addition some female participants of the assembly were allegedly indecently
assaulted and sexually harassed there were media reports that police
officers at the scene had let go the assaulters furthermore a reporter of
the british broadcasting corporation said that based on the information
obtained from the police the attacks obviously involved triad members
```

- Dr Helena Wong

How do we compare?

- Compare the meaning of the speech
  - Quite hard — needs the computer to be able to understand language
- Compare the ways in which certain words (e.g. CY Leung, democracy) are used
  - Kind of like the “liberal media bias” problem
  - Doable with Python, but still not easy
- Let’s just compare the words that they use for a start!

Comparing Words

- If we look at any text (book, speech, essay, powerpoint) in English, there are some words that appear very frequently.
    - the, of, and, that, in, a, were, to, …
- If we just simply compare all words, we may end up with the "conclusion" that everybody's speeches are very alike!

Cyd Ho

the	172
of	78
to	57
and	50
in	38
is	29
that	21
for	20
not	19
police	18
by	16
this	16
we	15
a	15
tear	13
there	13
chief	12
will	12
i	11
gas	11
at	11
are	10
from	10
it	10
president	10
has	9
on	9

Helena Wong

the	143
of	55
to	44
and	39
a	31
police	30
in	30
that	26
is	22
ipcc	21
have	18
members	17
are	16
this	16
who	16
for	15
mr	15
i	15
on	14
as	13
has	13
central	12
officers	11
public	11
occupy	10
or	10
there	10

Regina Ip

the	128
of	58
and	41
to	37
in	36
that	30
is	30
i	26
a	24
are	18
have	17
not	17
has	16
for	15
movement	15
occupy	14
on	14
central	13
people	13
many	13
by	12
president	12
he	12
such	12
or	11
been	11
mr	11

Comparing Words

- There is a set of words in English (and in any other language) called “stopwords”
- These words do not carry much information.
  - Mostly articles, determiners, pronouns, prepositions, etc.
    - the, of, and, that, in, a, were, to, …
  - But they are useful for determining authorship
    - Each person has his/her distinct “signature” when writing or speaking
  - To determine point of view, it is useful to look at keywords instead of stopwords

Cyd

police	18
tear	13
chief	12
gas	11
president	10
government	9
central	7
time	7
executive	7
secretary	7
assembly	7
fired	7
persons	7
members	6
officials	6
month	6
bill	6
hong	6
administration	5
peoples	5
like	5
peaceful	5
decisionmaking	5
kong	5
deputy	4
aforesaid	4
handling	4

Helena

police	30
ipcc	21
members	17
central	12
public	11
officers	11
occupy	10
secretary	9
against	8
president	8
question	7
security	7
assembly	7
member	7
government	6
participants	6
council	6
wong	6
triad	5
order	5
people	5
ip	4
protester	4
supplementary	4
demonstrations	4
complaints	4
matter	4

Regina

movement	15
occupy	14
central	13
people	13
president	12
kong	10
hong	10
political	9
secretary	8
ordinance	6
police	6
forces	6
government	5
incident	5
council	5
legislative	5
public	5
foreign	4
parties	4
wong	4
october	4
external	4
societies	4
organizers	4
year	4
organized	4
organization	4

Our strategy

- We will adopt a simple strategy to get a taste of this problem.

- We will pick the top *n* non-stopwords.

- Then we will build a word-frequency dictionary for each legislator, based on these words.

Calculating Similarity

Helena	
word	count
central	61
occupy	19
order	28
peace	0
peaceful	13
police	127
public	112

24241 non-stopwords

Cyd	
word	count
central	56
occupy	18
order	69
peace	5
peaceful	11
police	134
public	289

40301 non-stopwords

Regina	
word	count
central	54
occupy	25
order	11
peace	2
peaceful	1
police	56
public	23

10740 non-stopwords

Clearly, using just the number of words (the count) is not the best idea!

Normalize by the total number of words each person "contributed"

Helena	
word	Frequency
central	0.002516
occupy	0.000784
order	0.001155
peace	0
peaceful	0.005363
police	0.005239
public	0.004620

24241 non-stopwords

Cyd	
word	Frequency
central	0.001389
occupy	0.000447
order	0.001712
peace	0.000124
peaceful	0.000273
police	0.003325
public	0.007171

40301 non-stopwords

Regina	
word	Frequency
central	0.005028
occupy	0.002328
order	0.0010242
peace	0.000186
peaceful	0.000093
police	0.0052142
public	0.0021415

10740 non-stopwords

Calculating Similarity

	Helena	Cyd	Regina
word	Frequency	Frequency	Frequency
central	0.002516	0.001389	0.005028
occupy	0.000784	0.000447	0.002328
order	0.001155	0.001712	0.0010242
peace	0	0.000124	0.000186
peaceful	0.005363	0.000273	0.000093
police	0.005239	0.003325	0.0052142
public	0.004620	0.007171	0.0021415

- As a first step, we can simply calculate the probability that two people will mention the same word in their speeches.
- And compare the similarity across a lot of words over a long (e.g. one year) period of time.
- Not the best way to do comparisons, but it is a start.

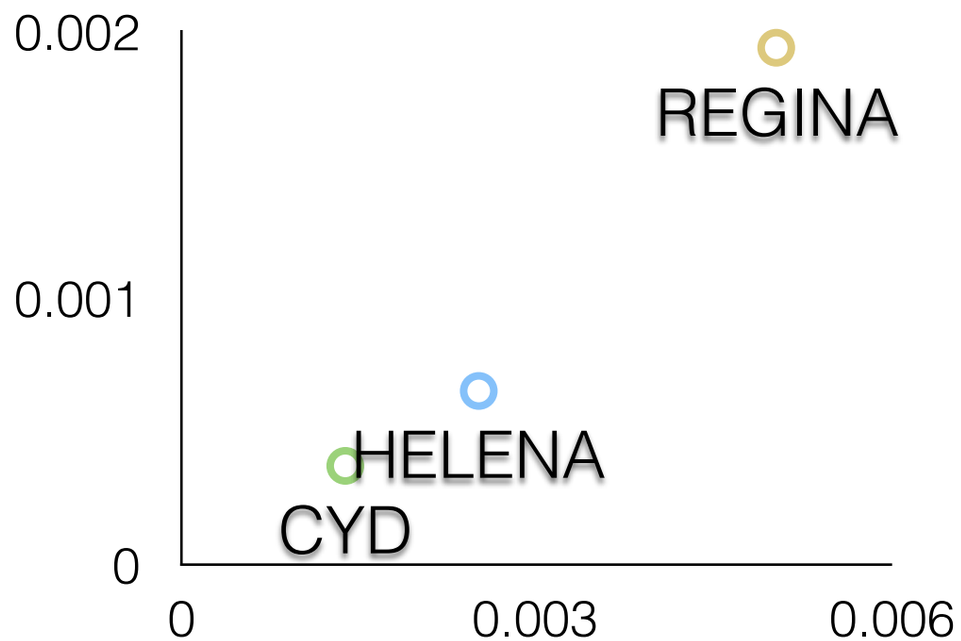
Calculating Similarity

- One very traditional way of calculating similarities between vocabularies is by calculating the *distance* between them

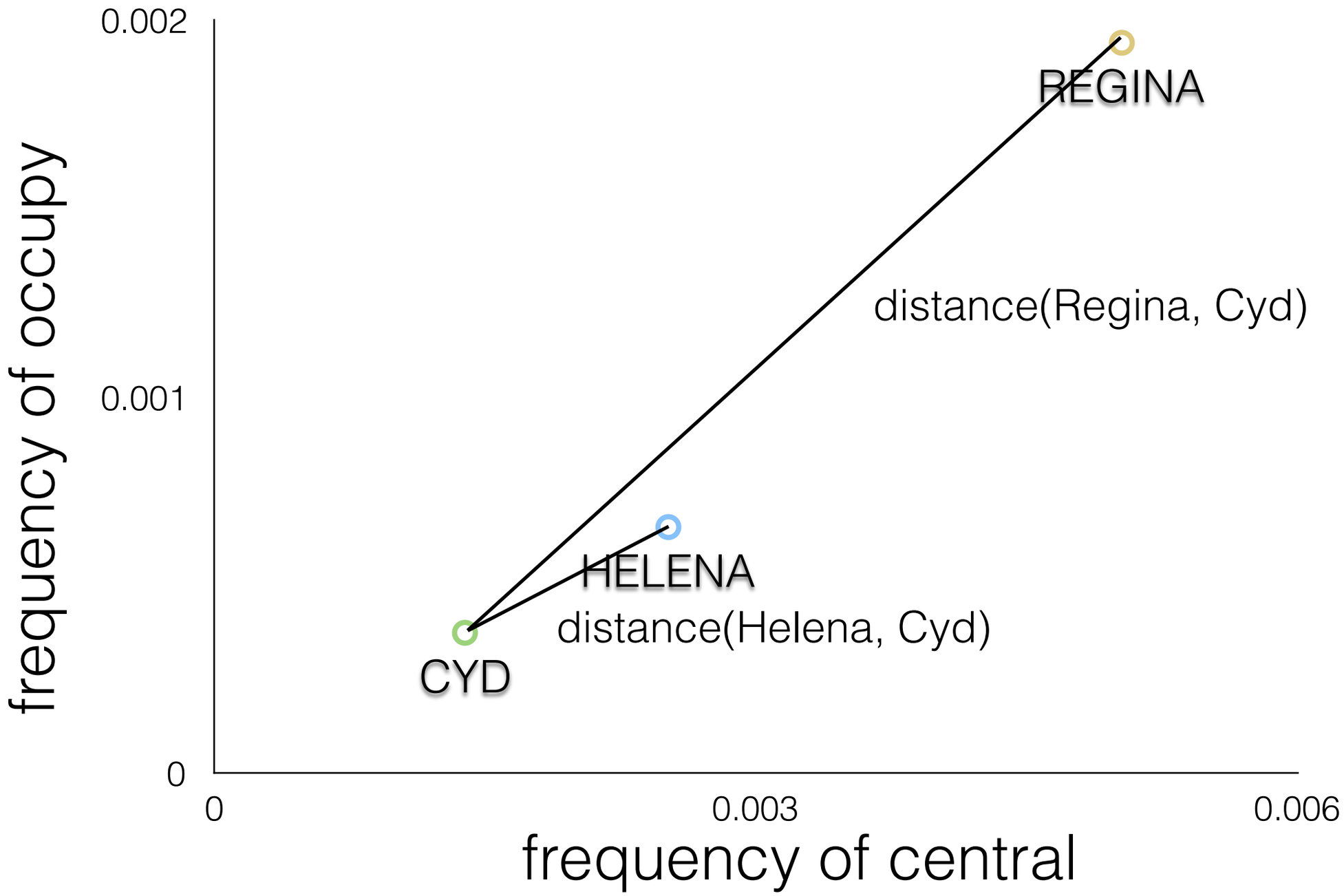
- Supposing we are interested in two words, *occupy* and *central*.

- We can visualize the vocabulary as a point in a 2-dimensional space
  - x = frequency of central
  - y = frequency of occupy

	Frequency		
word	**Helena**	**Cyd**	**Regina**
central	0.002516	0.001389	0.005028
occupy	0.000784	0.000447	0.002328

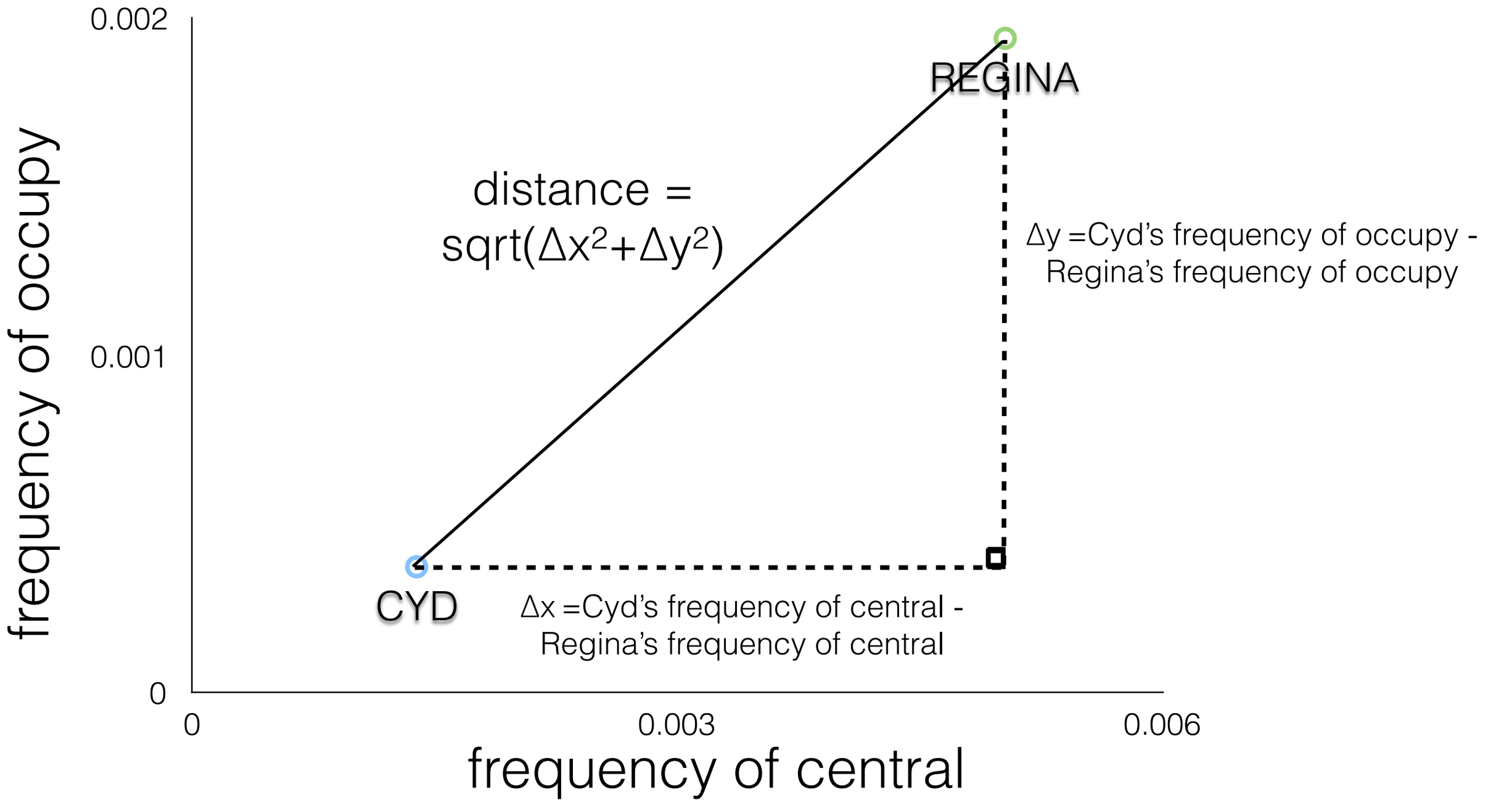


	Frequency		
word	Helena	Cyd	Regina
central	0.002516	0.001389	0.005028
occupy	0.000784	0.000447	0.002328



	Frequency	
word	Cyd	Regina
central	0.001389	0.005028
occupy	0.000447	0.002328

Given any two points, the distance between them can be calculated using the Pythagoras Theorem



Euclidean Distance

word	Helena Frequency	Cyd Frequency	Difference	Difference^2
central	0.002516	0.001389	0.001127	0.000001270129
occupy	0.000784	0.000447	0.000337	0.000000113569
order	0.001155	0.001712	-0.000557	0.000000310249
peace	0	0.000124	-0.000124	0.000000015376
peaceful	0.005363	0.000273	0.00509	0.0000259081
police	0.005239	0.003325	0.001914	0.000003663396
public	0.004620	0.007171	0.002551	0.000006507601

Sum of difference^2
0.00000777773

Euclidean Distance
0.0027888582

Euclidean Distance

word	Helena Frequency	Regina Frequency	Difference	Difference^2
central	0.002516	0.005028	−0.002512	0.000006310144
occupy	0.000784	0.002328	−0.001544	0.000002383936
order	0.001155	0.0010242	0.0001308	0.00000001710864
peace	0	0.000186	−0.000186	0.000000034596
peaceful	0.005363	0.000093	0.00527	0.0000277729
police	0.005239	0.0052142	0.000024800000000	0.000000000150400
public	0.004620	0.0021415	0.0024785	0.0000614296225

Sum of difference^2
0.00004266226193

Euclidean Distance
0.0065316354713042 6

HW2-6

- For HW2-6 (will be finalized and released next week, to be due on Dec 7), we will
    - Read in a few legislators' speeches, some keywords, and
    - Calculate the distances between the given legislators
    - Print out the result into a csv file
    - Import into Google Sheets and look at it using conditional formatting
    - Come up with some conclusions (in English)