

# HG 2051: Language and the Computer (Computational Linguistics with Python)



Week 1: Introduction, Organization, Main Issues

# Today's lecture/session

- Introductions and preliminaries
- Administrative matters
- Course overview
  - Why computers & linguistics?
  - What this course is (and what it isn't)
- Getting Started
  - Algorithmic thinking
  - Environment Setup
  - Basics of Version Control
  - Running Python
  - Introduction to GitHub
  - Homework 1

# Instructor background

Dr. Hiram Ring

I'm a field linguist with interests in:

- language documentation/description (PhD, NTU 2015)
- linguistic typology and language contact
- historical reconstruction
- natural language processing and machine learning (since 2015)
- SEAsian languages, particularly of the Austroasiatic phylum

I also write/record/perform music ([www.hiramring.com](http://www.hiramring.com))  
and maintain a website for classification of implicit motives  
([www.implicitmotives.com](http://www.implicitmotives.com))

## Main fieldwork site

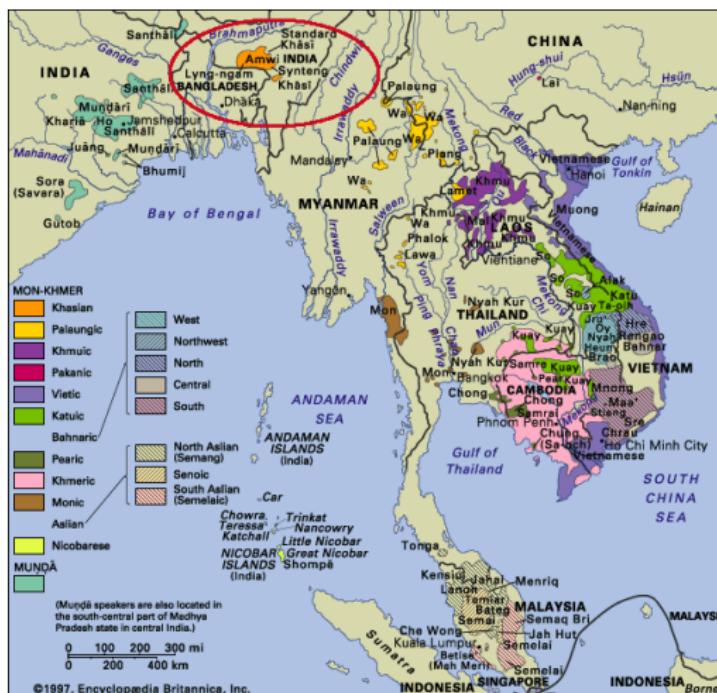
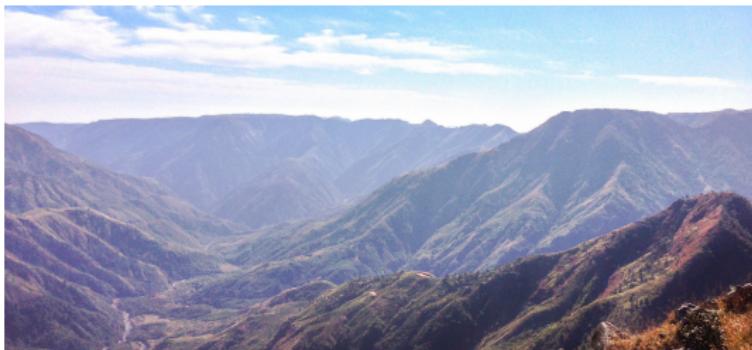


Figure 1: The Austroasiatic languages

## Main fieldwork site: Meghalaya, India



## Instructor contact

Dr. Hiram Ring

Office: SHHK-03-20

E-mail: [hiram.ring@ntu.edu.sg](mailto:hiram.ring@ntu.edu.sg)

Consultation hours: email me for an appointment – either in person, or online

## Student Introductions

- Name, year, linguistic interest (i.e. field/course of study)
- What background or knowledge do you have concerning programming?
  - Javascript, C/C++, Excel spreadsheets, R, Python etc.
  - Awareness of what programming is used for (algorithms, security, etc..) and related concerns
  - “nothing” is ok
- Why are you taking this course? (What do you hope to gain out of it?)
  - NOT “nothing”

# Administrative matters

- Schedule: <https://hg2051-ntu.github.io>
- Continuous Assessment:
  - Homework (autograded)
  - Project 1 (30%)
  - Project 2 (30%)
  - Quiz 1 (15%)
  - Quiz 2 (15%)
  - Participation (10%)
- Extra Credit: You can get 1~5% extra credit by getting a patch accepted to an open source project related to the course (e.g., NLTK). Your total grade cannot go over 100%. Contact me if you're interested.

## Continuous Assessment components

- Homework (autograded):
  - gives you practice writing actual code and submitting it
- Project 1 (CA1: 30%)
  - Write a Python program that uses text corpora to evaluate or produce a new resource, then write a brief 4-page paper describing the process and findings. [TBD]
- Project 2 (CA2: 30%)
  - Work with a group of 3-4 to develop a new resource or improve on an existing resource, writing a program to process the data. Summarize your process and findings in an 8-page cowritten paper. [TBD]

## Continuous Assessment components

- Quiz 1 (CA3a: 15%)
  - Midterm quiz (2hrs) involving a programming challenge with the ability to access online resources.
- Quiz 2 (CA3b: 15%)
  - Final quiz (2hrs) involving a programming challenge with the ability to access online resources.
- Participation (CA4: 10%)

# Why use Computers in Linguistics?

- Linguistics without computers is like taking a walk (or a long, hard hike)
  - It can be very pleasant
  - You can see lots of details
  - There is only so much ground you can cover
- Using a software tool is like catching the MRT
  - Very efficient for set routes
  - You have to adapt to it
  - Hard to customize
- Programming is like driving a car
  - It is expensive to start off (you have to learn!)
  - You are free to go where you want to

# The Goal of this Course

*To learn \*enough\* about programming to flexibly **analyze data** and then \*do something with it\**

- Coding is done in Python
- We will learn techniques and some software libraries particular to computational linguistics
- You will be able to write your own programs by the end

## HG2051 Prerequisites

- A little linguistic knowledge
  - You know what a word is
  - You know what a part of speech is
  - You know what a parse tree is
    - (If you don't know these, you will have to do a little background reading)
- A computer running Windows, Mac OS, or Linux
  - It is possible to learn using school computers, but it will be much harder
- No computational knowledge required
  - You have to be ready to learn
  - If you are a very experienced Python programmer, then you will not learn so much

## What HG2051 isn't

- We won't be learning how to build cars
  - this is the prerequisite for further NLP courses
  - ... but we won't be writing taggers and parsers (yet)
- It is not just an introduction to Python, but rather one motivated by NLP
- It is not very easy, but it is fun

## The Three Virtues of a Programmer

- **Laziness:** The quality that makes you go to great effort to reduce overall energy expenditure. It makes you write labor-saving programs that other people will find useful, and document what you wrote so you don't have to answer so many questions about it.
- **Impatience:** The anger you feel when the computer is being lazy. This makes you write programs that don't just react to your needs, but actually anticipate them. Or at least pretend to.
- **Hubris:** The quality that makes you write (and maintain) programs that other people won't want to say bad things about.

Larry Wall, Tom Christiansen, Randal L. Schwartz, and Stephen Potter (1996) Programming Perl 2nd Ed, O'Reilly.

# Readings

- Readings will come from a variety of freely available sources (no required textbook)
- You must read the material before class
  - I will assume you have done so
  - Programming is not (just) knowledge but a skill; we should spend our class time practicing that skill

# Homework

- Homework will comprise various practice problems aimed at familiarizing you with practical programming skills/tools.
- Some general guidelines:
  - Try to type everything on your own (at least at first), don't just copy/paste. This can be time-consuming initially, but the process will help you to remember the coding elements.
  - Try to understand the logical steps/process so you can apply them to different problems in the future.

# AI tools

- On the use of AI tools (such as ChatGPT), this course will adopt NTU's policy on the use of AI tools for coursework:
  - Give proper citations if you use any AI tool
    - 1 identify any generative AI tools used
    - 2 declare how the tools are used in submitted work
  - Some additional notes:
    - the tool may give slightly wrong information, requiring extensive debugging
    - you may end up with a working Python program that doesn't produce any output or produces plausible but incorrect output
    - if you don't understand what the code does, this will be pretty clear to me

# AI tools example 1: prompt

The screenshot shows a dark-themed AI chat application. At the top left is a sidebar with a '+ New chat' button. In the main area, a user message is shown in a dark box:

You 14:24  
write a python program to translate between words in the nltk swadesh corpus

To the right of the user message is a small yellow cartoon character icon.

Below the user message, a ChatGPT response is shown in a light blue box:

ChatGPT 14:24  
Sure! Here's a Python program that translates words between languages in the nltk Swadesh corpus:

```
```python
from nltk.corpus import swadesh
from nltk.translate import AlignedSent, Alignment, IBMModel1

def translate_word(word, source_lang, target_lang):
    # Get word translations from Swadesh corpus
    word_pairs = swadesh.entries([source_lang, target_lang])

    # Create aligned sentences for training the translation model
    aligned_sents = [AlignedSent(word_pair, word_pair)
                    for word_pair in word_pairs]
```


At the bottom of the screen, there is a footer bar with a 'Delete all chat' button and an 'Enter your message.' input field. To the right of the input field is a large blue send button with a white triangle icon.


```

# AI tools example 1: issues

The screenshot shows a dark-themed AI chat interface. On the left, there's a sidebar with a '+ New chat' button and a search bar containing the partially typed message 'write a python pro...'. Below the search bar are two small icons: a trash can and a reply arrow. At the bottom of the sidebar is a 'Delete all chat' button with a trash can icon.

**You** 14:24  
write a python program to translate between words in the nltk swadesh corpus

**ChatGPT** 14:24  
Sure! Here's a Python program that translates words between languages in the nltk Swadesh corpus:

```
```python
from nltk.corpus import swadesh
from nltk.translate import AlignedSent, Alignment,
IBMMModel1

def translate_word(word, source_lang, target_lang):
    # Get word translations from Swadesh corpus
    word_pairs = swadesh.entries([source_lang,
target_lang])

    # Create aligned sentences for training the translation
    model
    aligned_sents = [AlignedSent(word_pair, word_pair)
for word_pair in word_pairs]
```

At the bottom of the interface is a large input field with the placeholder 'Enter your message.' and a blue send button with a white right-pointing arrow. The entire interface has a clean, modern look with a yellow profile picture of a person in the top right corner.

## AI tools example 2: prompt

The screenshot shows a conversational AI interface. On the left, there's a sidebar with a "New chat" button and a message input field containing "write a python pro...". Below the input field is a "Delete all chat" button. The main area has a dark background with a light blue message card from "ChatGPT". The message is timestamped at 14:42 and contains the following Python code:

```
I Here's a Python program that uses the NLTK Swadesh corpus to find word pair equivalents between languages: ``python import nltk # Load the Swadesh corpus nltk.download('toolbox') nltk.download('swadesh') from nltk.corpus import swadesh def find_word_pairs(language1, language2): """ Find word pair equivalents between two languages using the Swadesh corpus. """ # Get the word lists for the two languages word_list1 = swadesh.words(language1) word_list2 = swadesh.words(language2) # Find word pair equivalents word_pairs = [] for word1, word2 in zip(word_list1, word_list2): word_pairs.append((word1, word2)) return word_pairs # Example usage language1 = 'eng' # English language2 = 'fra' # French word_pairs = find_word_pairs(language1, language2) print("Word pair equivalents between")
```

To the right of the message card is a "You" section with a timestamp of 14:42 and the text "write a python program that uses the nltk swadesh corpus to find word pair equivalents between languages". There's also a small yellow profile icon of a person's head.

At the bottom of the screen, there's a footer bar with a "Delete all chat" button and a "Enter your message." input field. To the right of the input field is a large grey button with a white play/pencil icon.

## AI tools example 2: errors

```
File "/Users/hiramring/Devel/lcomp/old/test.py", line 27, in <module>
    word_pairs = find_word_pairs(language1, language2)
File "/Users/hiramring/Devel/lcomp/old/test.py", line 13, in find_word_pairs
    word_list1 = swadesh.words(language1)
File "/Users/hiramring/.virtualenvs/lcomp/lib/python3.9/site-packages/nltk/corpus/reader/wordlist.py", line 21, in words
    for line in line_tokenize(self.raw(fileids))
File "/Users/hiramring/.virtualenvs/lcomp/lib/python3.9/site-packages/nltk/corpus/reader/api.py", line 218, in raw
    with self.open(f) as fp:
File "/Users/hiramring/.virtualenvs/lcomp/lib/python3.9/site-packages/nltk/corpus/reader/api.py", line 231, in open
    stream = self._root.join(file).open(encoding)
File "/Users/hiramring/.virtualenvs/lcomp/lib/python3.9/site-packages/nltk/data.py", line 334, in join
    return FileSystemPathPointer(_path)
File "/Users/hiramring/.virtualenvs/lcomp/lib/python3.9/site-packages/nltk/compat.py", line 41, in _decorator
    return init_func(*args, **kwargs)
File "/Users/hiramring/.virtualenvs/lcomp/lib/python3.9/site-packages/nltk/data.py", line 312, in __init__
    raise OSError("No such file or directory: %r" % _path)
OSError: No such file or directory: '/Users/hiramring/nltk_data/corpora/swadesh/eng'
```

## AI tools: notes

There are lots of issues with current AI systems. For an accessible introduction to the many concerns, this Scientific American article is a good start:

- AI Causes Real Harm. Let's Focus on That over the End-of-Humanity Hype

# Algorithmic Thinking

- Exercise: How to make kaya toast
- Also see: <http://www.cookingforengineers.com/>

(break)

# Environment setup, Git, Python, Homework

<https://hg2051-ntu.github.io/week1.html>