Using Python To Break Basic Ciphers: Hill Climbing Algorithms

John Hackett
jhackettga@gmail.com

December 12th 2013

Regarding Python….

Mhfk ikhidk, xskg ohgelhgqku xrqs n ilhydkf, qsrgb R bghx, R'dd vmk lkpvdnl kzilkmmrhgm. Ghx qskc snwk qxh ilhydkfm.

Excuse me?

Hint: It’s a Substitution Cipher…

- Scrambled Alphabet Substitution Cipher
  - State of the art for 1200 A.D, now used for cryptogram puzzles
  - Map plaintext alphabet into cipher alphabet (A->E, B->Q, etc.)
  - Possible keys: 26! => 88 bits (in practice, usually weakened)

- Human Attacks
  - Take small sections of the puzzle (words) and test possible keys to see if they yield English
  - Optimizations: Frequency Analysis, Target Relatively Unique Words…

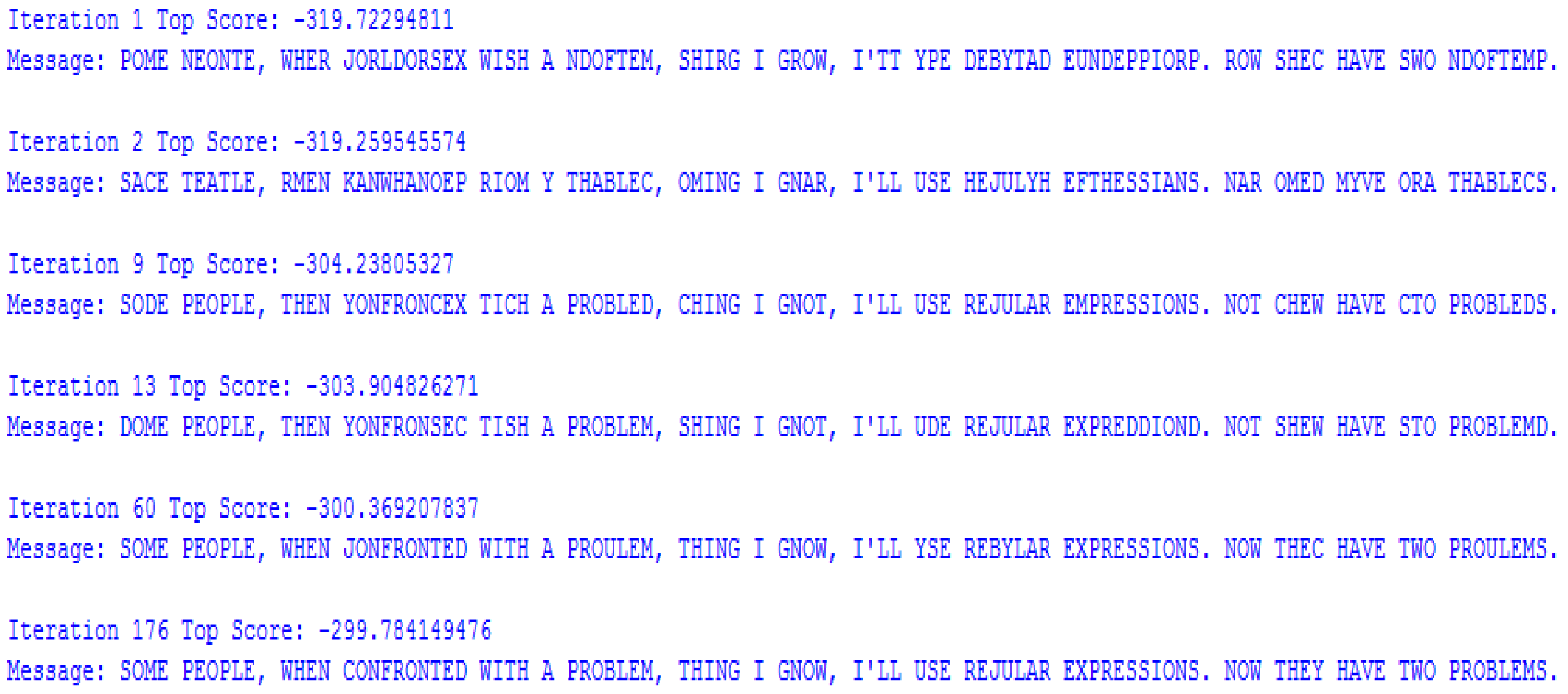
- Creating A Coded Message in Python:
  - base_alphabet = string.ascii_uppercase
  - cipher_alphabet = <shuffled version of base_alphabet>
  - string.maketrans(base_alphabet, cipher_alphabet)
  - string.translate(message_text, alpha_shift)

Automating The Attack

Generate Keys => See If It Looks Like English…

- Keys are easy:
  - Random shuffle the alphabet
  - Use the shuffled dictionary as a list..
- "Looks Like English" – statistical test using sequences of n-letters
  - Trigrams, Quadragrams, etc…
  - Does distribution of decoded message resemble english?
  - Fitness score (log probability) => how much of the key you guessed
- A process….
  - Randomly generate a key
  - Test the key – how close is the message to english?
  - If close, tweak the key (shuffle one letter) and retest
  - When your score stops improving, test another key…

The Method In Action…



```
Iteration 1 Top Score: -319.72294811
Message: POME NEONTE, WHER JORLDORSEX WISH A NDOFTEM, SHIRG I GROW, I'TT YPE DEBYTAD EUNDEPPIORP. ROW SHEC HAVE SWO NDOFTEMP.

Iteration 2 Top Score: -319.259545574
Message: SACE TEATLE, RMEN KANWHANOEP RIOM Y THABLEC, OMING I GNAR, I'LL USE HEJULYH EFTHESSIANS. NAR OMED MYVE ORA THABLECS.

Iteration 9 Top Score: -304.23805327
Message: SODE PEOPLE, THEN YONFRONCEX TICH A PROBLED, CHING I GNOT, I'LL USE REJULAR EMPRESSIONS. NOT CHEW HAVE CTO PROBLEDS.

Iteration 13 Top Score: -303.904826271
Message: DOME PEOPLE, THEN YONFRONSEC TISH A PROBLEM, SHING I GNOT, I'LL UDE REJULAR EXPREDDIOND. NOT SHEW HAVE STO PROBLEMD.

Iteration 60 Top Score: -300.369207837
Message: SOME PEOPLE, WHEN JONFRONTED WITH A PROULEM, THING I GNOW, I'LL YSE REBYLAR EXPRESSIONS. NOW THEC HAVE TWO PROULEMS.

Iteration 176 Top Score: -299.784149476
Message: SOME PEOPLE, WHEN CONFRONTED WITH A PROBLEM, THING I GNOW, I'LL USE REJULAR EXPRESSIONS. NOW THEY HAVE TWO PROBLEMS.
```

Regular Expressions are apparently not a problem..

Further Explorations…

- Stuff to Test: Potential Optimizations
  - N-gram type/source: bi-grams, tri-grams, quads, etc. Text corpus?
  - Restart criteria: how long to mutate a key before trying another?
  - Multiple Fitness Functions: word pattern dictionary (once close)
  - Different cipher-texts – some messages weaker than others

- Websites & Resources
  - Hacking secret ciphers with python (Al Sweigart) [entry level resource]
  - Overview of Classical Ciphers [entry level article of different ciphers]
  - O'Reilly: Machine Learning for Hackers – Ch 7
    - R-based, good discussion of hill-climbing optimization math
    - Link to github sample code repo
  - Practical Cyptography (site) – Multiple approaches, lots of Python code