



IBM Model II

Jean Mark Gawron

Linguistics 581

San Diego State University

gawron@mail.sdsu.edu

<http://www.rohan.sdsu.edu/~gawron>

Inner loop for IBM Model 2 Estimation

```
1  for sentence pairs  $(f_s, e_s)$ 
2     $l_e := \text{len}(e_s); l_f := \text{len}(f_s)$ 
3    for indices  $f$  in  $[1, \dots, l_f]$ 
4       $\text{total}_s(f_s[f]) := 0$ 
5      for indices  $e$  in  $[0, \dots, l_e]$ 
6         $\text{total}_s(f_s[f]) += t(f_s[f], e_s[e]) * a(e, f, l_f, l_e)$ 
7      for indices  $e$  in  $[0, \dots, l_e]$ 
8         $c := (t(f_s[f], e_s[e]) * a(e, f, l_f, l_e)) / \text{total}_s(f_s[f])$ 
9         $\text{count}(f_s[f], e_s[e]) += c; \text{d\_counts}(e, f, l_e, l_f) += c$ 
10        $\text{total}(e_s[e]) += c; ; \text{total}_d(f, l_e, l_f) += c$ 
11
15   $\text{smooth\_distortion\_counts}(\text{d\_counts}, \text{total}_d)$ 
16  for  $(f, e)$  in  $\text{domain}(\text{count})$ 
17     $t(f, e) := \text{count}(f, e) / \text{total}(e)$ 
19  for  $(i, j, I, J)$  in  $\text{domain}(\text{d\_counts})$ 
20     $a(i, j, I, J) := \text{d\_counts}(i, j, I, J) / \text{total}_d(j, I, J)$ 
```

Initializing distortion

Assuming 40 is max sentence length, and a NULL for English (= I)

```
1  for indices  $I$  in [1, ..., 41]
2       $u := 1/I$ 
3      for indices  $J$  in [1, ..., 40]
4          for indices  $j$  in [1, ...,  $J$ ]
5              for indices  $i$  in [1, ...,  $I$ ]
6                   $a(i, j, I, J) := u$ 
```

Smoothing distortion counts

Using Laplace Smoothing (add .5 but with prob weighted counts)

```
1 funct smooth_distortion_counts(d_counts, total_d)
2   laplace := 1.0
3   for (i, j, I, J) in d_counts
4     val := d_counts(i, j, I, J)
5     if val > 0 and val < laplace
6       laplace := val
7     fi
9   laplace := 0.5 * laplace
10  for (i, j, I, J) in d_counts
11    d_counts(i, j, I, J) += laplace
13  for (j, I, J) in total_d
14    total_d(j, I, J) += laplace * J
```



Your code resources

1. There is a version of IBM Model I available at

`http://bulba.sdsu.edu/ling582`.

See the code at the end of the file, under

```
if __name__ == '__main__':
```

for guidelines on how to run it. The main methods of interest are *compute_em_mod1*, *compute_em_round_slick* (which implements the IBM model I factoring trick), and *compute_em_naive* (which does not).

2. There is a version of `scoring_tools.py` available at the same location.

Your assignment: Implementing

The following assignment is due Thursday, Oct 23, 2008.

1. Implement Model 2. Your code should run n iterations of Model 1 followed by m of model 2 (try $m = 5$ and $n = 5$). Build on the provided version of IBM Model I or implement your own.
2. Use functions (or methods) in your code. At LEAST the following:
 - (a) *compute_em_mod1*: computes exactly n rounds of EM model 1. Calls *compute_em_round*.
 - (b) *compute_em_round*: computes one round. Opens and closes corpus files and recomputes *tprobs* (translation model dictionary):

$$tprobs[(f, e)] = \text{Prob}(f | e)$$

- (c) Analogously for model 2
3. Implement a version of the pseudo “viterbi” algorithm discussed in class, which USES the *tprobs* and distortion model trained in EM to align the corpus.

Assignment: Training and Testing

1. Train and test on 1000 and 10000 sentence corpora. The 10K set is:
 - /opt/corpora/mt/wmt07/alignments/short_aligned.0.10001.en
 - /opt/corpora/mt/wmt07/alignments/short_aligned.0.10001.fr
 - /opt/corpora/mt/wmt07/alignments/short_aligned.0.10001.aligned
2. For now, yes, test and train on the same corpus.
3. Turn in the code for model 1 and model 2 and the your evaluation scores.
4. Last, when you've succeeded in a test on the training set, try training and testing on different corpora. Use short_aligned.0.1001.* and short_aligned.2.1001.*.
Report.