

No title

Erik Hartman¹

Jonas Wallin²

¹ Division of Infection Medicine, Faculty of Medicine, Lund University

² Department of Statistics, Lund University

Algorithms

Algorithm 1: Simulate proteolysis of protein P .

Input: protein P , $n_{generate}$, θ_{enzyme} , θ_{gamma} , p_{endo} , p_{exo}

Output: f

```

1  $f \leftarrow \text{dict}[\text{sequence} : \text{count}]$ 
2  $f(P) \leftarrow n_P$ 
3 while  $i < n_{generate}$  do
4    $n_{generate} \leftarrow \sum_x T(x)$ 
5    $x \sim U(0, 1)$ 
6   if  $x > p_{endo}$  then
7     //exoprotease
8     sequence to chew  $s \sim f$  //sample sequence from dict with weights
9     count
10     $x \sim U(0, 1)$ 
11     $a \leftarrow \text{gamma}(\text{len}(s), \theta_{gamma})$ 
12    if  $a < x$  then
13      //accept
14       $F(P) \leftarrow F(P) + 1$ 
15       $n_{generate} \leftarrow n_{generate} + 1$ 
16    end
17  else
18    //endoprotease
19     $f_{cut}(s) \leftarrow \sum_s N_{aa}^s * \theta_{aa}$ 
20    sequence to cut  $s \leftarrow f_{cut}(s)$  //sequence to cut
21    first index to cut  $index_a \sim s(\theta_{aa}(aa_x))$ 
22     $index_2 \sim s(\theta_{aa}(aa_x)) * \text{gamma}(|index_1 - index_2|)$ 
23    left  $\leftarrow s[: \min(index_1, index_2) + 1]$ 
24    middle  $\leftarrow s[\min(index_1, index_2) + 1 : \max(index_1, index_2)]$ 
25    right  $\leftarrow s[\max(index_1, index_2) + 1 : ]$ 
26    if  $\text{len}(middle) > 5$  then
27       $f(middle) \leftarrow f(middle) + 1$ 
28       $n_{generate} \leftarrow n_{generate} + 1$ 
29    end
30    for  $s$  in [left, right] do
31       $x \sim U(0, 1)$ 
32       $a \leftarrow \text{gamma}(\text{len}(s), \theta_{gamma})$ 
33      if  $a < x$  and  $s$  is not terminal peptide in  $P$  then
34        //accept
35         $F(P) \leftarrow F(P) + 1$ 
36         $n_{generate} \leftarrow n_{generate} + 1$ 
37      end
38    end
39 end
40 Return  $f$ 

```

Algorithm 2: Estimating θ numerically. To generate a guess, simulate degradation of protein P with parameters θ to generate $n_{generate}$ peptides (see Algorithm 1).

Input: protein P , n_P , true distribution T , θ , lr_{endo} , lr_{exo}

Output: θ

```

1 for  $i$  from 0 to  $n_{endo}$  do
2   Generate guess  $G$ ;
3   Compute loss  $L \leftarrow D_{KL}(G||T) + D_{KL}(T||G)$ ;
4   for each amino acid  $aa$  in  $\theta_{aa}$  do
5      $\theta_{aa}(aa) \leftarrow \theta_{aa}(aa) + lr_{endo}$ ;
6     Generate guess  $\hat{G}$  with new  $\theta$ ;
7     Compute new loss  $\hat{L} \leftarrow D_{KL}(\hat{G}||T) + D_{KL}(T||\hat{G})$ ;
8     while  $\hat{L} < L$  do
9       Compute weighted learning rate  $lr_w \leftarrow lr_{endo} * \hat{L} - L$ ;
10       $L \leftarrow \hat{L}$ ;
11       $\theta_{aa}(aa) \leftarrow \theta_{aa}(aa) + lr_w$ ;
12      Generate guess  $\hat{G}$  with new  $\theta$ ;
13      Compute new loss  $\hat{L} \leftarrow D_{KL}(\hat{G}||T) + D_{KL}(T||\hat{G})$ ;
14    end
15     $\theta_{aa}(aa) \leftarrow \theta_{aa}(aa) - lr_{endo}$  //revert the initial parameter-change
      (before while-loop);
16  end
17 end
18 Generate guess  $G$ ;
19 Compute loss  $L \leftarrow D_{KL}(G||T) + D_{KL}(T||G)$ ;
20 for  $i$  from 0 to  $n_{exo}$  do
21    $x \leftarrow$  uniformly random from 1, -1;
22    $e \leftarrow lr_{exo} * x$ ;
23    $\theta_{exo} \leftarrow \theta_{exo} + e$ ;
24   Generate guess  $\hat{G}$  with new  $\theta$ ;
25   Compute new loss  $\hat{L} \leftarrow D_{KL}(\hat{G}||T) + D_{KL}(T||\hat{G})$ ;
26   if  $\hat{L} > L$  then
27      $\theta_{exo} \leftarrow \theta_{exo} - e$ ;
28   else
29      $L \leftarrow \hat{L}$ ;
30   end
31 end
32 Return  $\theta$ ;

```

Algorithm 3: Estimate weights in graph with gradient descent.

Input: observed distribution T , lr , λ_1 , λ_2 , $n_{iterations}$

Output: G

```

1  $G \leftarrow \{V, E, W\}$ 
2  $V_L$  denotes the original protein (bottom node)
3 for  $i$  from 0 to  $n_{iterations}$  do
4
5     //generate guess
6      $p_{generated} \leftarrow \{\}$  // this represents the output distribution if starting
       from a given node
7      $T = \{s \in V \mid \text{there exists no } (s, t) \in E \text{ for any } t \in V\}$  //terminal nodes
8     for  $t \in T$  do
9          $p_{generated}[t] \leftarrow \mathbf{1}_t$  //onehot
10    end
11    while all nodes in  $G$  is not solved do
12        solvable  $\leftarrow \{s \in V \mid t \in p_{generated} \text{ for all } (s, t) \in E\}$ ; for  $s \in V$  do
13             $p_{generated}[s] = \sum w_{s,t} * p_{generated}[t] + 1 - \sum w_{s,t} * \mathbf{1}_t$ 
14        end
15    end
16     $\hat{T} \leftarrow p_{generated}[V_L]$ 
17
18    //compute loss
19     $L_1 \leftarrow \lambda_1 \sum |w|$ 
20     $L_2 \leftarrow \lambda_2 \sum w^2$ 
21     $L \leftarrow D_{KL}(T \mid \hat{T}) + D_{KL}(\hat{T} \mid T) + L_1 + L_2$ 
22
23    //compute gradient
24     $\frac{dT}{dw} \leftarrow \hat{T}_{V_L, t}(\hat{T}_s - \mathbf{1}_s)$ 
25     $\frac{dL}{dT} \leftarrow -\frac{\hat{T}}{\hat{T}}$ 
26     $\frac{dL}{dw} \leftarrow \frac{dL}{dT} * \frac{dT}{dw}$ 
27
28    //update graph
29     $k \leftarrow 1$ 
30    for  $s \in V$  do
31         $\hat{w}_{s,t} \leftarrow w_{s,t} - lr * (\frac{dL}{dw})_{s,t}$ 
32         $d_{s,t} = \hat{w}_{s,t} - w_{s,t}$ 
33    end
34    while  $\sum_t w_{s,t} + k * d_{s,t} > 1$  do
35         $k \leftarrow k/2$ 
36    end
37    while a better graph is not found or  $k$  isn't extremely small do
38         $\hat{W} \leftarrow W + d * k$ 
39         $\hat{T} \leftarrow$  generate guess with new weights
40         $\hat{L} \leftarrow D_{KL}(T \mid \hat{T}) + D_{KL}(\hat{T} \mid T) + L_1 + L_2$ 
41        if  $\hat{L} \leq L$  then
42             $G \leftarrow \{V, E, \hat{W}\}$ 
43        else
44             $k \leftarrow k/2$ 
45        end
46    end
47 end
48 Return  $G$ 

```
