Pfeature Manual

Table of Contents

1.0 Composition

In this section, we have described python functions developed for amino acid composition based feature generation. These modules can be used for feature generation for protein sequences to apply machine learning techniques for further analysis.

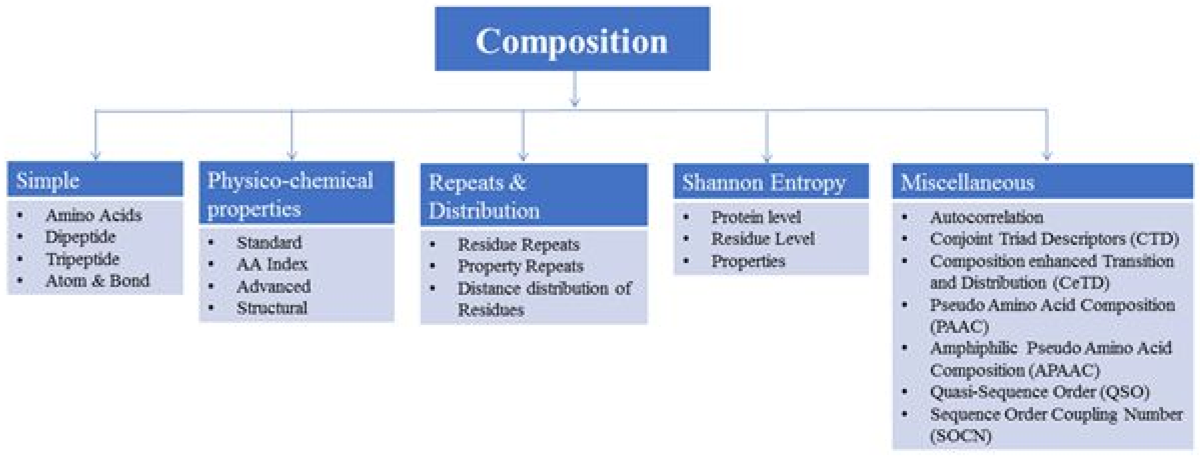

Composition
Simple
Amino Acids
Dipeptide
Tripeptide
Atom & Bond
Physico-chemical properties
Standard
AA Index
Advanced
Structural
Repeats & Distribution
Residue Repeats
Property Repeats
Distance distribution of Residues
Shannon Entropy
Protein level
Residue Level
Properties
Miscellaneous
Autocorrelation
Conjoint Triad Descriptors (CTD)
Composition enhanced Transition and Distribution (CeTD)
Pseudo Amino Acid Composition (PAAC)
Amphiphilic Pseudo Amino Acid Composition (APAAC)
Quasi-Sequence Order (QSO)
Sequence Order Coupling Number (SOCN)

Figure 1: This flowchart shows different menus/submenus for computing different type of composition-based features of protein/peptide composition.

1.1 Simple: This module describes python programs to generate simple composition based feature from a protein. This module is called simple composition as obvious composition has been computed like amino acid composition (20 features), dipeptide composition (400 features) tripeptide composition (8000 features). Pfeature web site provides dynamic web web page to compute these features, our python module have following functions to compute these features.

Function Title	**Description**
AAC	To calculate Amino acid composition of a peptide
AAC_NT	To calculate Amino acid composition of N-terminal residues defined by user
AAC_CT	To calculate Amino acid composition of C-terminal residues defined by user

AAC_rest	To calculate Amino acid composition of remaining residue from N-Terminal and C-Terminal Residues defined by user
AAC_split	To calculate Amino acid composition by splitting peptide into fragments defined by user
DPC	To calculate Dipeptide composition of a peptide
DPC_NT	To calculate Dipeptide composition of N-terminal residues defined by user
DPC_CT	To calculate Dipeptide composition of C-terminal residues defined by user
DPC_rest	To calculate Dipeptide composition of remaining residue from N-Terminal and C-Terminal Residues defined by user
DPC_split	To calculate Dipeptide composition by splitting a peptide into fragments defined by user
TPC	To calculate Tripeptide composition of a peptide
ATC	To calculate Atomic composition (% of Carbon, Hydrogen, Nitrogen, Oxygen, Sulphur content) of a peptide
ATC_NT	To calculate Atomic composition of N-terminal residues defined by user
ATC_CT	To calculate Atomic composition of C-terminal residues defined by user
ATC_rest	To calculate Atomic composition of remaining residue from N-Terminal and C-Terminal Residues defined by user
ATC_split	To calculate Atomic composition by splitting a peptide into fragments defined by user
BTC	To calculate Bond composition (% of Carbon, Hydrogen, Nitrogen, Oxygen, Sulphur content) of a peptide

BTC_NT	To calculate Bond composition of N-terminal residues defined by user
BTC_CT	To calculate Bond composition of C-terminal residues defined by user
BTC_rest	To calculate Bond composition of remaining residue from N-Terminal and C-Terminal Residues defined by user
BTC_split	To calculate Bond composition by splitting a peptide into fragments defined by user

1.1.1 Amino Acids

Description: This is a simplest feature, which is heavily used in literature for predicting function or structure of a protein. It computes the amino acid composition of each type of residue of a protein sequence. The compositions of all 20 natural amino acids were calculated using the following formula:

$$AAC_i = \frac{R_i}{L} \tag{1}$$

where $\boldsymbol{AAC_i}$ is amino acid composition of residue type $\boldsymbol{i}$; $\boldsymbol{R_i}$ and $\boldsymbol{L}$ number of residues of type $\boldsymbol{i}$ and length of sequence.

In order to compute amino acid composition of different portions of an amino acid sequence, we have developed number of python function (brief description is given below). In web server user may select portion of sequence for calculating protein features.

- **AAC**: Usage: aac_comp(input_filename)
  Description: This function will compute amino acid composition from whole sequence of a protein using Eq. 1.
- **AAC_NT**: Usage: aac_nt (input_filename, n)
  Description: This function computes the amino acid composition of residues selected from N-terminal using Eq.1, user need to provide number of N-terminal residues ***n*** to be used for calculating.
- **AAC_CT**: Usage: aac_ct (input_filename, m)
  Description: This function computes the amino acid composition of residue selected from C-terminal using Eq.1, user need to provide number of C-terminal residues ***m*** to be used for calculating.
- **AAC_REST**: Usage: aac_rest (input_filename, n, m)

Description: This function computes the amino acid composition of remaining residues of a sequence after cleaving ***n*** residues from N-Terminal and ***m*** residues from C-Terminal using Eq.1.

- **AAC_SPLIT:** Usage: aac_split (input_filename, s)
  Description: This function computes the amino acid composition of each portion after splitting sequence in ***s*** portions using Eq.1. This function is important to compute amino acid composition of different portion of a protein.

1.1.2 Dipeptide

Amino acid composition provides only number of different type of residues, no information about order of residues. Dipeptide composition is used to encapsulate the global information about each sequence, which gives a fixed pattern length of 400 (20 X 20). This representation encompassed the information about amino acid composition along local order of amino acid. Traditionally a dipeptide is made of consecutive residues (residue ***i*** and ***i+1***), In 2005, dipeptide of higher order were introduced (**J Biol Chem. 2005; 280:14427-32**). In case of higher order dipeptides, a dipeptide is made of ***i*** and ***i+2*** or ***i+3*** or ***i+4*** etc. instead of consecutive residues (See Figure 1, adapted from **J Biol Chem. 2005; 280:14427-32**).

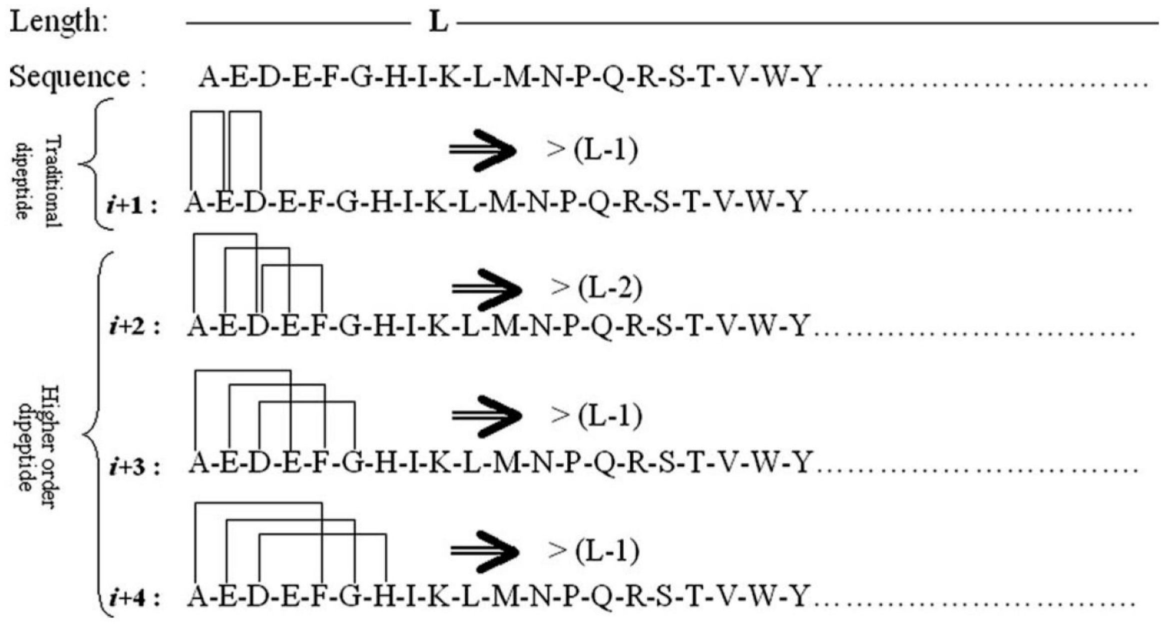

Length: L
Sequence : A-E-D-E-F-G-H-I-K-L-M-N-P-Q-R-S-T-V-W-Y............................
Traditional dipeptide
> (L-1)
i+1 : A-E-D-E-F-G-H-I-K-L-M-N-P-Q-R-S-T-V-W-Y............................
Higher order dipeptide
> (L-2)
i+2 : A-E-D-E-F-G-H-I-K-L-M-N-P-Q-R-S-T-V-W-Y............................
> (L-1)
i+3 : A-E-D-E-F-G-H-I-K-L-M-N-P-Q-R-S-T-V-W-Y............................
> (L-1)
i+4 : A-E-D-E-F-G-H-I-K-L-M-N-P-Q-R-S-T-V-W-Y............................

Figure 1: Graphical representation of traditional peptides and higher order dipeptides, figure is adapted from **J Biol Chem. 2005; 280:14427-32.**

In order to compute traditional dipeptide composition from a protein sequence following equation is used

$$DPC_i^j = \frac{D_i^j}{L-j} \qquad (2)$$

Where $\boldsymbol{DPC^j_i}$ is the fraction or composition of dipeptide of type ***i*** for jth order. $\boldsymbol{D^j_i}$ and ***L*** are the number of dipeptides of type ***i*** and length of a protein. Here higher order dipeptide $\boldsymbol{D^j_i}$ is made of residue $\boldsymbol{R_i}$ and $\boldsymbol{R_{i+j}}$ where value of ***j*** is 2 or more. In case ***j*** is equal to 1 then dipeptide is called traditional dipeptide.

We have developed number of python functions to compute traditional and higher order dipeptide composition in different portions of an amino acid sequence, we have developed number of python function. In web server user may select portion of sequence for calculating protein features. Following is brief description of these python functions.

- **DPC**: Usage: dpc (input_filename, j):
  Description: This function compute dipeptide in a sequence (input_filename), j is order or dipeptide using Eq. 2.
- **DPC_NT**: Usage: dpc_nt (input_filename, j, n)
  Description: This function computes the dipeptide composition of residues selected from N-terminal using Eq.2, user need to provide order of dipeptide ***j*** and ***n*** number of N-terminal residues.
- **DPC_CT:** Usage: dpc_ct (input_filename, j, m)
  Description: This function computes the dipeptide composition of residues selected from C-terminal using Eq.2, user need to provide order of dipeptide ***j*** and ***m*** number of C-terminal residues.
- **DPC_REST**: Usage: dpc_rest (input_filename, j, n, m)
  Description: This function computes the dipeptide composition of order ***j*** of rest of sequence after removing ***n*** residues from N-Terminal and ***m*** residues from C-Terminal using Eq.2.
- **DPC_SPLIT**: Usage: dpc_split (input_filename, j, s)
  Description: This function computes the dipeptide composition of each portion after splitting sequence in ***s*** portions using Eq.2. This function is important to compute dipeptide composition of order ***j*** of different portion of a protein.

1.1.3 Tripeptide

Three consecutive amino acids form a tripeptide which provide local order in addition to simple composition. Both previous and next residues are used to form a tripeptide. There are total 800 (20*20*20) possible tripeptides from by 20 type of natural residue.

$$TPC_i = \frac{T_i}{L-2} \qquad (3)$$

where ***TPC**$_i$* is tripeptide composition of type ***i***, out of possible 800 tripeptides. ***T**$_i$* and ***L*** are number of tripeptides of type ***i*** and length of a protein sequence. In order to compute tripeptide composition in a sequence, following functions has been developed.

- **TPC**: Usage: tpc_comp(Input_filename)
  Description: This function computes the tripeptide composition of a sequence using Eq.3.

1.1.4 Atom & Bond

All amino acids are made of atoms and bonds. In this module, we compute different type atom and bond composition. Atomic composition is fraction of Carbon, Hydrogen, Nitrogen, Oxygen and Sulphur atoms present in a protein sequence. For bond composition four types of bonds are considered total number of bonds (including aromatic), hydrogen bond, single bond and double bond. The number of values for each kind of bond is provided as bonds.csv file.

$$ATC_i = \frac{A_i}{N} \tag{4}$$

$$BTCi = \frac{Bi}{N} \tag{5}$$

where ***ATC**$_i$* is atomic composition of type ***i***, ***A**$_i$* and ***N*** are number of atoms of type ***i*** and number of atoms in a protein. Where ***BTC**$_i$* is atomic composition of type ***i***, ***B**$_i$* and ***N*** are number of atoms of type ***i*** and number of atoms in a protein.In order to compute atomic composition in a sequence, following functions has been developed.

Atomic Composition	Bond Composition
Carbon Atom	Total Bonds
Hydrogen Atom	Hydrogen Bond
Nitrogen Atom	Single Bond
Oxygen Atom	Double Bond
Sulphur Atom	

Table1: List of Atoms and Bonds included in ATC & BTC Pfeature programs.

- **ATC**: Usage: atc (input_filename)
  Description: This function computes the atomic composition of each amino acid residue of the peptide sequence using Eq.4.

- **ATC_NT:** Usage: atc_nt (input_filename, n)
  Description: This function computes the atomic composition of each amino acid residue selected from N-terminal using Eq. 4, user can give peptide sequence and the value of ***n*** number of N-terminal residues.
- **ATC_CT:** Usage: atc_ct (input_filename, m)
  Description: This function computes the atomic composition of each amino acid residue selected from C- terminal using Eq. 4, user can give peptide sequence and the value of ***m*** number of C-terminal residues.
- **ATC_REST:** Usage: atc_rest (input_filename, n, m)
  Description: This function computes the atomic composition of each amino acid composition of remaining peptide residues cleaved from both N-Terminal and C-Terminal ends using Eq.4, user can give peptide sequence and the value of ***n*** (position from N-Terminal) and ***m*** (position from C-Terminal).
- **ATC_SPLIT:** Usage: atc_split (input_filename, s)
  Description: This function computes the atomic composition of each portion after splitting sequence in ***s*** portions using Eq.4.
- **BTC**: Usage: btc (input_filename)
  Description: This function computes the bond composition of each amino acid residue of the peptide sequence using Eq.5.
- **BTC_NT:** Usage: btc_nt (input_filename, n)
  Description: This function computes the bond composition of each amino acid residue selected from N-terminal using Eq.5, user need to provide number of N-terminal residues ***n*** to be used for calculating.
- **BTC_CT**: Usage: btc_ct (input_filename, m)
  Description: This function computes the bond composition of each amino acid residue selected from C- terminal using Eq.5, user need to provide number of C-terminal residues ***m*** to be used for calculating.
- **BTC_REST:** Usage: btc_rest (input_filename, n, m)
  Description: This function computes the bond composition of each amino acid composition of remaining peptide residues cleaved from both N-Terminal and C-Terminal ends using Eq.5, user can give peptide sequence and the value of ***n*** (position from N-Terminal) and ***m*** (position from C-Terminal).
- **BTC_SPLIT:** Usage: btc_split (input_filename, s)
  Description: This function computes the bond composition of each portion after splitting sequence in ***s*** portions using Eq.5.

Atom & Bond also considered together with the above given operations (NT, CT, REST, SPLIT).

1.2 Physico-Chemical properties

The physico-chemical properties were used to represent a protein. The values of each physico-chemical property for all 20 amino acids were normalized between 0 and 1 using the standard conversion formula. The input vector has scalar values, each representing the average value of a distinct physico-chemical property of protein (**Nucleic Acids Res. 2004; 32:W414-9**).

1.2.1 Standard physico-chemical properties

This function calculates the fraction of each standard physico-chemical property in given sequences. Following properties have been incorporated in Pfeature for calculating compositional features

Table 2: List of physico-chemical properties included in Pfeature for computing features

Positively Charged	Aromaticity	Hydroxylic
Negatively Charged	Acidity	Sulphur Content
Neutral Charge	Basicity	Tiny
Polarity in residues	Neutral (pH)	Small
Non-polarity in residues	Hydrophobicity	Large
Aliphaticity	Hydrophilicity	
Cyclicity	Neutral towards water	

We used following formula to calculate these features

$$PCP_i = \frac{P_i}{L} \tag{6}$$

where $\boldsymbol{PCP_i}$ is physico-chemical properties composition of residue type $\boldsymbol{i}$; $\boldsymbol{P_i}$ and $\boldsymbol{L}$ are sum of property of type $\boldsymbol{i}$ and length of sequence. In order to compute composition of standard properties for different portions of an amino acid sequence, we have developed number of python function (brief description is given below). In web server user may select portion of sequence and type of properties for calculating protein features.

- **PCP**: Usage: pcp_comp(input_filename)
  Description: This function will compute property composition from whole sequence of a protein using Eq. 5.
- **PCP_NT**: Usage: pcp_nt (input_filename, n)
  Description: This function computes the properties composition of residues selected from N-terminal, user need to provide number of N-terminal residues ***n*** to be used for calculating.

- **PCP_CT**: Usage: pcp_ct (input_filename, m)
  Description: This function computes the properties composition of residue selected from C-terminal, user need to provide number of C-terminal residues ***m*** to be used for calculating.
- **PCP_REST**: Usage: ppc_rest (input_filename, n, m)
  Description: This function computes the properties composition of remaining residues of a sequence after cleaving ***n*** residues from N-Terminal and ***m*** residues from C-Terminal.
- **PCP_SPLIT:** Usage: pcp_split (input_filename, s)
  Description: This function computes the properties composition of each portion after splitting sequence in ***s*** portions. This function is important to properties composition of different portion of a protein.

1.2.2 Amino Acid index (AAindex)

AAindex is a database of amino acid indices, where AAindex is a set of 20 numerical values representing various physico-chemical and biochemical properties of amino acids. Current version 9.0 of database have total 566 AA indices (https://www.genome.jp/dbget/AAindex/list_of_indices). Pfeature allow user to compute composition of selected AA index via web interface or python function, using following equation

$$AAIC_i = \frac{AAI_i}{L} \tag{7}$$

where $\boldsymbol{AAIC_i}$ is AA index composition of residue type ***i***; $\boldsymbol{AAI_i}$ and ***L*** are sum of AA index value of type ***i*** and length of sequence. In order to compute composition of AA index values for different portions of an amino acid sequence, we have developed number of python function (brief description is given below). In web server user may select portion of sequence and type of properties for calculating protein features.

- **AAIC**: Usage: aaic_comp(input_filename,a)
  Description: This function will compute AA index composition of ***a*** indices from whole sequence of a protein using Eq. 7.
- **AAIC_NT**: Usage: aaic_nt (input_filename, n,a)
  Description: This function computes AA index composition of ***a*** indices from N-terminal, user need to provide number of N-terminal residues ***n*** and list of AA indices ***a*** to be used for calculating.
- **AAIC_CT**: Usage: aaic_ct (input_filename, m,a)
  Description: This function computes AA index composition of ***a*** indices from C-terminal, user need to provide number of C-terminal residues ***m*** and list of AA indices ***a*** to be used for calculating.
- **AAIC_REST**: Usage: aaic_rest (input_filename, n, m,a)

Description: This function computes AA index composition of ***a*** indices of remaining residues of a sequence after cleaving ***n*** residues from N-Terminal and ***m*** residues from C-Terminal.

- **AAIC_SPLIT:** Usage: aaic_split (input_filename, s,a)
  Description: This function computes AA index composition of ***a*** indices of each portion after splitting sequence in ***s*** portions. This function is important to properties composition of different portion of a protein.

1.2.3 Advanced properties

This module allow to compute composition of advanced properties like z1, z2, z3, z4 and z5 of a protein sequence. This "**Advanced**" module is similar to "**Standard**" module of computing physico-chemical properties.

1.2.4 Structural Properties

This module allow to compute composition of advanced properties like secondary structure and surface accessibility of a protein sequence. This "**Structural**" module is similar to "**Standard**" module of computing physico-chemical properties.

1.3 Repeats & Distribution

Most of composition modules describes above measures fraction of particular type of residue or residue property. One of the problem with existing features is that they do not measure repeat of particular type of residue or distribution. In this study, we introduced new features, which compute repeats of amino acids and distribution of amino acids.

Function Title	**Description**
RRI	To compute Repetitive Residue Information of amino acid of protein sequences.
RRI_NT	To compute Repetitive Residue Information of N-terminal residues defined by user.
RRI_CT	To compute Repetitive Residue Information of C-terminal residues defined by user.
RRI_rest	To compute Repetitive Residue Information of remaining residue from N-Terminal and C-Terminal residues of defined by user

RRI_split	To compute Repetitive Residue Information of amino acid by splitting peptide into fragments defined by user
DDOR	To compute Distance Distribution of residue (DDOR) of protein sequences.

1.3.1 Residue Repeats

This function calculates the Repetitive Residue Information (RRI) for a peptide/protein sequence. RRI measures number of continuous runs of a residue type in a sequence, it can be calculated using following formula.

$$RRI_i = \frac{\sum_{j=1}^{N} (R_j)^2}{\sum_{j=1}^{N} R_j} \tag{8}$$

where $\boldsymbol{RRI_i}$ **,** $\boldsymbol{N}$ and $\boldsymbol{R_j}$ are residue repeat information, maximum number of occurance and number of runs/repeats in occurrence $\boldsymbol{j}$ respectively for residue type $\boldsymbol{i}$.

Example: If a residue is a residue type occurs once at time then value of RRI will be one. For example amino acid alanine ***A*** occurs four times in following sequence "G**A**R**A**GRG**A**RDE**A**RT**A**G"; each time single run. It means ***N*** will be 5, RRI for A can be calculated using following formula

$$RRI_A = \frac{(1)^2 + (1)^2 + (1)^2 + (1)^2 + (1)^2}{1+1+1+1+1} = \frac{5}{5} = 1$$

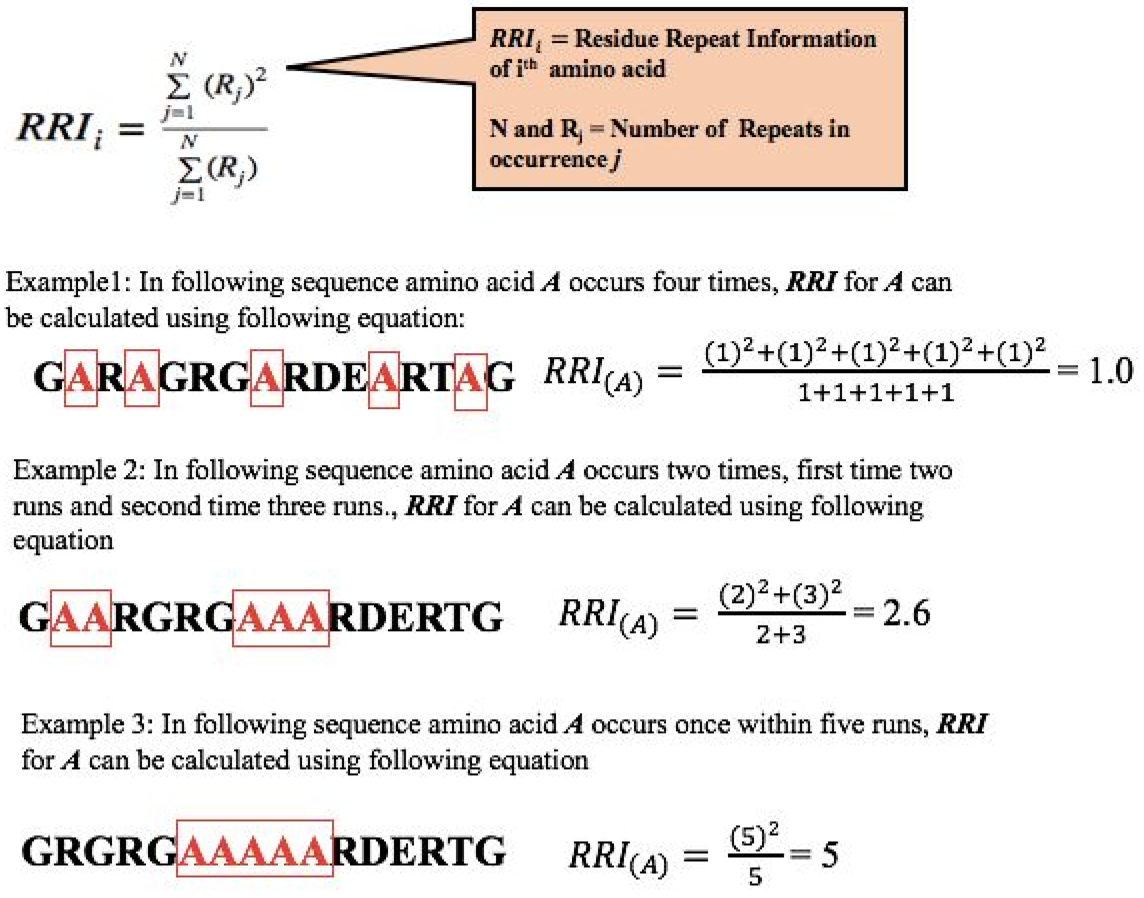
In following sequence "G**AA**RGRG**AAA**RDERTG" amino acid ***A*** occurs two times, first time two runs and second time three runs. It means $\boldsymbol{N=2, R_1=2}$ and $\boldsymbol{R_2=3}$, ***RRI*** for ***A*** can be calculated using following equation

$$RRI_A = \frac{(2)^2 + (3)^2}{2+3} = \frac{4+9}{5} = 2.6$$

In following sequence "GRGRGAA**AAA**RDERTG" amino acid ***A*** occurs once with 5 runs. It means $\boldsymbol{N=1,}$ and $\boldsymbol{R_1=5;}$ ***RRI*** for ***A*** can be calculated using following equation

$$RRI_A = \frac{(5)^2}{5} = \frac{25}{5} = 5$$

This means for a given residue type, minimum RRI will be 1 and maximum will be total number of that type of residues in sequence. This value measures multiple runs of a residue in a sequence.



$$RRI_i = \frac{\sum_{j=1}^{N} (R_j)^2}{\sum_{j=1}^{N} (R_j)}$$

RRI_i = Residue Repeat Information of i^th amino acid

N and R_j = Number of Repeats in occurrence j

Example1: In following sequence amino acid *A* occurs four times, ***RRI*** for *A* can be calculated using following equation:

GARAGRGARDEARTAG $RRI_{(A)} = \frac{(1)^2+(1)^2+(1)^2+(1)^2+(1)^2}{1+1+1+1+1} = 1.0$

Example 2: In following sequence amino acid *A* occurs two times, first time two runs and second time three runs., ***RRI*** for *A* can be calculated using following equation

GAARGRGAAARDERTG $RRI_{(A)} = \frac{(2)^2+(3)^2}{2+3} = 2.6$

Example 3: In following sequence amino acid *A* occurs once within five runs, ***RRI*** for *A* can be calculated using following equation

GRGRGAAAAARDERTG $RRI_{(A)} = \frac{(5)^2}{5} = 5$

Figure 2: Calculation of Repetitive Residue Information (RRI) for a peptide/protein sequence.

In order to compute repetitive residue information of different portions of an amino acid sequence, we have developed number of python function (brief description is given below). In web server user may select portion of sequence for calculating protein features.

- **RRI**: Usage: rri (input_filename)
  Description: This function will compute repetitive residue information from whole sequence of a protein using Eq. 8.
- **RRI_NT**: Usage: rri_nt (input_filename, n)
  Description: This function computes the repetitive residue information of residues selected from N-terminal using Eq.8, user need to provide number of N-terminal residues ***n*** to be used for calculating.
- **RRI_CT**: Usage: rri_ct (input_filename, m)

Description: This function computes the repetitive residue information of residue selected from C-terminal using Eq.8, user need to provide number of C-terminal residues ***m*** to be used for calculating.

- **RRI_REST**: Usage: rri_rest (input_filename, n, m)
  Description: This function computes the repetitive residue information of remaining residues of a sequence after cleaving ***n*** residues from N-Terminal and ***m*** residues from C-Terminal using Eq.1.
- **RRI_SPLIT:** Usage: rri_split (input_filename, s)
  Description: This function computes the amino acid composition of each portion after splitting sequence in ***s*** portions using Eq.8. This function is important to compute amino acid composition of different portion of a protein.

1.3.2 Property Repeats

This function calculates property repeat information (PRI) which gives the information of repetitiveness of each physicochemical property within a peptide/ protein sequence.

$$PRI_i = \frac{\sum_{j=1}^{N} (P_j)^2}{\sum_{j=1}^{N} P_j} \tag{9}$$

where $\boldsymbol{PRI_i}$ **,** $\boldsymbol{N}$ and $\boldsymbol{P_j}$ are property repeat information, maximum number of occurance and number of runs/repeats in occurrence ***j*** respectively for property type ***i***.

1.3.3 Distance distribution of Residues

This function distance distribution of residues (DDOR) computes the distribution of residue on the basis of distance from N-terminal, C-terminal and inter-distances between same residue within the given peptide/protein sequence.

$$DDOR_i = \frac{(R_{NT})^2 + \sum_{j=1}^{N} (R_j)^2 + (R_{CT})^2}{(L - F_i) + 1} \tag{10}$$

where, ***DDOR**$_i$* is distance distribution of residue type ***i,*** ***N*** is total number of inter-residue distances for type ***i.***

R_{NT} = Residue distance from N-terminal

R_j = Inter-distance between residue type ***i***

R_{CT} = Residue distance from C-terminal

L = Total length of protein sequence

F_i = Frequency of residue type ***i***

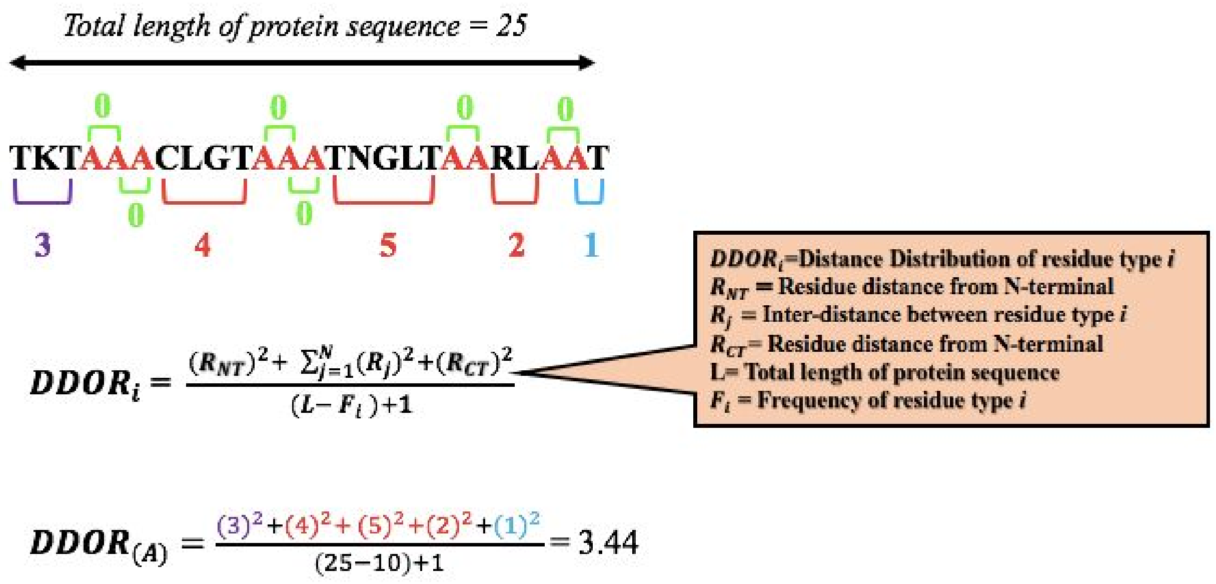

Total length of protein sequence = 25
TKTAAACLGTAAATNGLTAARLAAT
0
0
0
0
0
0
3
4
5
2
1
$DDOR_i = \frac{(R_{NT})^2 + \sum_{j=1}^{N}(R_j)^2 + (R_{CT})^2}{(L - F_i) + 1}$
$DDOR_i$=Distance Distribution of residue type i
R_{NT} = Residue distance from N-terminal
R_j = Inter-distance between residue type i
R_{CT}= Residue distance from N-terminal
L= Total length of protein sequence
F_i = Frequency of residue type i
$DDOR_{(A)} = \frac{(3)^2+(4)^2+(5)^2+(2)^2+(1)^2}{(25-10)+1} = 3.44$

Figure 3: Calculation of Distance Distribution of Residue (DDOR) for peptide/protein sequence.

1.4 Shannon Entropy

1.4.1 Protein Level

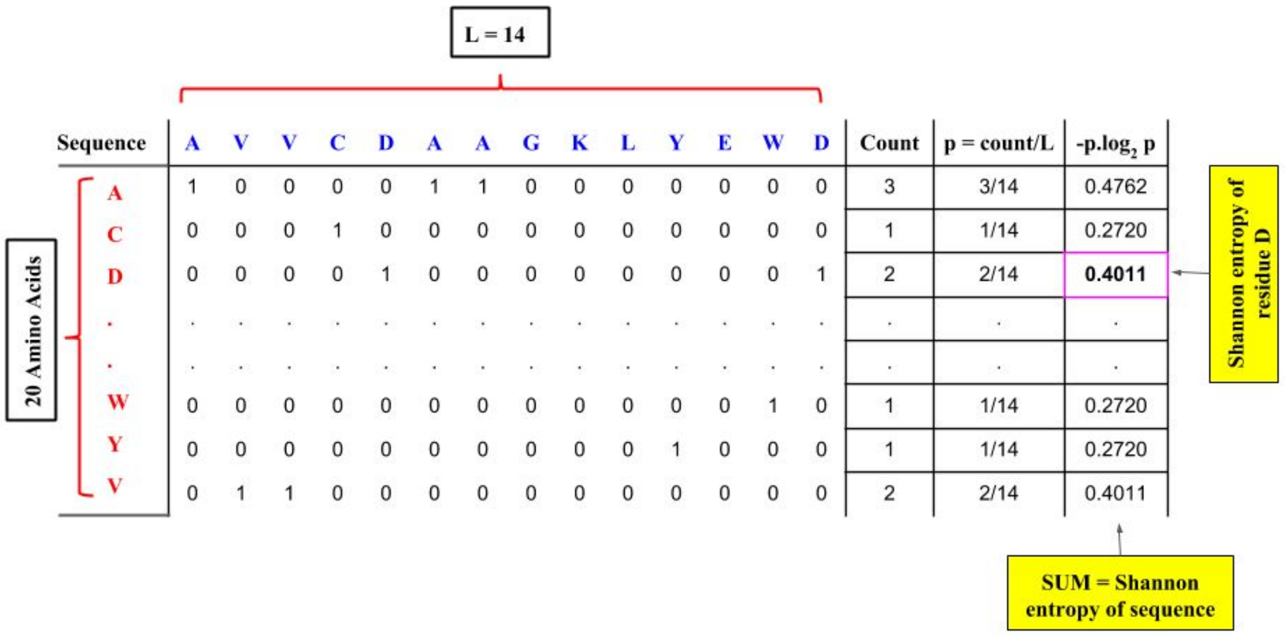
Shannon entropy for a protein/peptide sequence can be computed by the standard expression:

$$H(X) = -\sum_{i=1}^{20} p_i \log_2 p_i \tag{11}$$

Where i is the amino acid in the sequence (i=A, C, D, … ,Y) and X is any protein/peptide sequence. See figure below for more details.

Function Title	**Description**
SE	To compute Shannon Entropy of protein/ peptide sequences.
SE_NT	To compute Shannon Entropy of N-terminal residues defined by user.
SE_CT	To compute Shannon Entropy of C-terminal residues defined by user.
SE_REST	To compute Shannon Entropy of remaining residue from N-Terminal and C-Terminal residues as defined by user.
SE_SPLIT	To compute Shannon Entropy of sub-sequences by splitting protein/ peptide into fragments as defined by user.

- **SE:** Usage: SE (input_filename)
  Description: This function computes the shannon entropy of a protein sequence. Shannon entropy of all sequences were calculated using the Eq. 11
- **SE_NT:** Usage: SE_NT (input_filename, n)
  Description: This function computes the shannon entropy of the N-terminal of a protein sequence, where n is number of N-terminal residues.
- **SE_CT:** Usage: SE_CT (input_filename, m)
  Description: This function computes the shannon entropy of the C-terminal of a protein sequence, m is number of C-terminal terminal residues.
- **SE_REST:** Usage: SE_REST (input_filename, n, m)
  Description: This function computes the shannon entropy of a protein sequence by removing the n and m residues from N- and C-terminal respectively.
- **SE_SPLIT:** Usage: SE_SPLIT(input_filename, s)
  Description: This function computes the shannon entropy of the subsequences of a protein sequence after splitting sequence in s segments.


L = 14
20 Amino Acids
Shannon entropy of residue D
SUM = Shannon entropy of sequence

Sequence	A	V	V	C	D	A	A	G	K	L	Y	E	W	D	Count	p = count/L	$-p.\log_2 p$
A	1	0	0	0	0	1	1	0	0	0	0	0	0	0	3	3/14	0.4762
C	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	1/14	0.2720
D	0	0	0	0	1	0	0	0	0	0	0	0	0	1	2	2/14	**0.4011**
.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
W	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1/14	0.2720
Y	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	1/14	0.2720
V	0	1	1	0	0	0	0	0	0	0	0	0	0	0	2	2/14	0.4011

Figure 5 : Calculation of shannon entropy for a protein/ peptide sequence or sub-sequence at protein and residue levels.

1.4.2 Residue Level

- **SER:** Usage: SER (input_filename)
  Description: This function computes the shannon entropy of the residues of the peptide/ protein sequence. Here, user can input a list of sequences for which shannon entropy for every residue can be computed separately for each sequence.
  Shannon entropy of all sequences were calculated using the following formula:

$$H(X) = - p_i \log_2 p_i \tag{12}$$

  In the above shannon entropy equation, p_i is the probability of a given amino acid in the sequence. [NOTE: Replace all residues except under investigation to zero and calculate entropy iteratively for each of them.]

Function Title	**Description**
SER	To compute Shannon Entropy of all amino acids of protein/ peptide sequences.

SER_NT	To compute Shannon Entropy of all amino acids of N-terminal residues defined by user.
SER_CT	To compute Shannon Entropy of all amino acids of C-terminal residues defined by user.
SER_REST	To compute Shannon Entropy of all amino acids of remaining residue from N-Terminal and C-Terminal residues as defined by user.
SER_SPLIT	To compute Shannon Entropy of all amino acids of sub-sequences by splitting protein/ peptide into fragments as defined by user.

- **SER_NT:** Usage: SER_NT (input_filename, n)
  Description: This function computes the shannon entropy of the residues of the N-terminal of the peptide/ protein sequence. Here, user can input a list of sequences and N-terminal length for which shannon entropy of the residues can be computed separately.

- **SER_CT:** Usage: SER_CT (input_filename, m)
  Description: This function computes the shannon entropy of the residues of the C-terminal of the peptide/ protein sequence. Here, user can input a list of sequences and C-terminal length for which shannon entropy of the residues can be computed separately.

- **SER_REST:** Usage: SER_REST (input_filename, n, m)
  Description: This function computes the shannon entropy of the residues of peptide/ protein sequence by removing the N- and C-terminal of the sequence. Here, input to the function is the file having all the sequences and the size of N-terminal and C-terminal.

- **SER_SPLIT:** Usage: SER_SPLIT(input_filename, s)
  Description: This function computes the shannon entropy of the residues of subsequences of peptide/ protein sequence. Here, input to the function is the file having all the sequences and the number of splits. The output of the function is shannon entropy for residues corresponding to every split.

1.4.3 **Properties**

This function calculates the Shannon Entropy of a particular Physicochemical property in a sequence. Let the sequence be of length 'l' and has r_i instances of a property present in the sequence, then the Shannon Entropy $H_i(x)$ of a particular physicochemical property is calculated using the following formula:

$$H_i = -p_i\, log(p_i) - (1 - p_i) log\, (1 - pi) \qquad (13)$$

where p_i is r_i/l

Function Title	**Description**
SHANNON_all	To compute the Shannon Entropy of an entire protein/ peptide sequence for a physicochemical property defined by user.
SHANNON_NT	To compute the Shannon Entropy of N-terminal residues of protein/ peptide sequence for a physicochemical property defined by user.
SHANNON_CT	To compute the Shannon Entropy of C-terminal residues of protein/ peptide sequence for a physicochemical property defined by user.
SHANNON_REST	To compute Shannon Entropy of remaining residue from N-Terminal and C-Terminal residues of protein/ peptide sequence for a physicochemical property defined by user.
SHANNON_SPLIT	To compute Shannon Entropy of sub-sequences by splitting protein/ peptide into fragments for a physicochemical property as defined by user.

- **SHANNON_all**: Usage: shannon_all(input_filename,featureNum)
  Description: This function gives the Shannon Entropy of an entire sequence for a physicochemical property represented by a number. The user can input the file having all the sequences and a feature number for which the entropy needs to be calculated.

- **SHANNON_NT:** Usage: shannon_NT(input_filename, featureNum, n)
  Description: This function calculates Shannon Entropy of a physicochemical property of a residues from N terminal. User can input the file containing these sequences, featureNumber and the number of n terminal residues.

- **SHANNON_CT:** Usage: shannon_CT(input_filename, featureNum, n)

Description: This function calculates Shannon Entropy of a physicochemical property of a residues from C terminal. User can input the file containing these sequences, featureNumber and the number of C terminal residues.

- **SHANNON_REST:** Usage: shannon_rest(input_filename, featureNum, m,n)
  Description: This function calculates Shannon Entropy of a physicochemical property by removing 'm' N-terminal residues and 'n' C-terminal residues. User can input the file containing these sequences, featureNumber, number of N terminal residues and number of C terminal residues.

- **SHANNON_SPLIT:** Usage: shannon_split(input_filename, featureNum, SPLIT)
  Description: This function calculates the Shannon Entropy of a physicochemical property by splitting the entire sequence into SPLIT parts and then doing the calculation iteratively over each split part. The user can input a file containing the sequences, desired physicochemical property and number of splits to be made in each sequence.

1.5 **Miscellaneous**

Function Title	**Description**
Autocorr	To compute all three autocorrelation descriptors for the protein/ peptide sequences for the AAindex accession numbers given in 'aaindex_file', at a specific value of d given in 'dval' defined by the user.
Autocorr_NT	To compute all three autocorrelation descriptors for the protein/ peptide sequences for the AAindex accession numbers given in 'aaindex_file', at a specific value of d given in 'dval' of N-terminal residues defined by user.
Autocorr_CT	To compute all three autocorrelation descriptors for the protein/ peptide sequences for the AAindex accession numbers given in 'aaindex_file', at a specific value of d given in 'dval' of C-terminal residues defined by user.

Autocorr_REST	To compute all three autocorrelation descriptors for the protein/ peptide sequences for the AAindex accession numbers given in ‘aaindex_file’, at a specific value of d given in ‘dval’ as defined by user.
Autocorr_SPLIT	To compute all three autocorrelation descriptors for the protein/ peptide sequences for the AAindex accession numbers given in ‘aaindex_file’, at a specific value of d given in ‘dval’ of sub-sequences by splitting protein/ peptide into fragments as defined by user.
CTC	The Conjoint Triad Calculation of the Descriptors of protein/ peptide sequences.
CTC_NT	The Conjoint Triad Calculation of the Descriptors of N-terminal residues defined by user.
CTC_CT	The Conjoint Triad Calculation of the Descriptors of C-terminal residues defined by user.
CTC_REST	The Conjoint Triad Calculation of the Descriptors of remaining residue from N-Terminal and C-Terminal residues as defined by user.
CTC_SPLIT	The Conjoint Triad Calculation of the Descriptors of sub-sequences by splitting protein/ peptide into fragments as defined by user.
CeTD	To compute Composition enhanced Transition Distribution of a peptide
CeTD_NT	To compute Composition enhanced Transition Distribution of N-terminal residues defined by user
CeTD_CT	To compute Composition enhanced Transition Distribution of C-terminal residues defined by user
CeTD_rest	To compute Composition enhanced Transition Distribution of remaining residue from N-Terminal and C-Terminal Residues defined by user

CeTD_split	To compute Composition enhanced Transition Distribution by splitting peptide into fragments defined by user
PAAC	To compute Pseudo Amino acid composition of a peptide
PAAC_NT	To compute Pseudo Amino acid composition of N-terminal residues defined by user
PAAC_CT	To compute Pseudo Amino acid composition of C-terminal residues defined by user
PAAC_rest	To compute Pseudo Amino acid composition of remaining residue from N-Terminal and C-Terminal Residues defined by user
PAAC_split	To compute Pseudo Amino acid composition by splitting a peptide into fragments defined by user
APAAC	To compute Amphiphilic pseudo amino acid composition of a peptide
APAAC_NT	To compute Amphiphilic pseudo amino acid composition of N-terminal residues defined by user
APAAC_CT	To compute Amphiphilic pseudo amino acid composition of C-terminal residues defined by user
APAAC_rest	To compute Amphiphilic pseudo amino acid composition of remaining residue from N-Terminal and C-Terminal Residues defined by user
APAAC_split	To compute Amphiphilic pseudo amino acid composition by splitting a peptide into fragments defined by user
QSO	To compute Quasi-Sequence Order of a peptide
QSO_NT	To compute Quasi-Sequence Order of N-terminal residues defined by user
QSO_CT	To compute Quasi-Sequence Order of C-terminal residues defined by user

QSO_rest	To compute Quasi-Sequence Order of remaining residue from N-Terminal and C-Terminal Residues defined by user
QSO_split	To compute Quasi-Sequence Order by splitting a peptide into fragments defined by user
SOCN	To compute Sequence Order Coupling Number of a peptide
SOCN_NT	To compute Sequence Order Coupling Number of N-terminal residues defined by user
SOCN_CT	To compute Sequence Order Coupling Number of C-terminal residues defined by user
SOCN_rest	To compute Sequence Order Coupling Number of remaining residue from N-Terminal and C-Terminal Residues defined by user
SOCN_split	To compute Sequence Order Coupling Number by splitting a peptide into fragments defined by user

1.5.1 **Autocorrelation**

Autocorrelation descriptors are defined based on the distribution of amino acid properties along the sequence. The amino acid properties used here are various types of amino acid indices (http://www.genome.ad.jp/dbget/aaindex.html). Three type of autocorrelation descriptors are used here viz. Normalized Moreau-Broto, Moran and Geary autocorrelation descriptors as implemented in (**Dong, Jie, et al.** ***Journal of cheminformatics*** **10.1 (2018): 16.**)

The python functions for calculation of these descriptors are described below:

- **Autocorr:** usage: autocorr_aa(seq_file, aaindex_file, dval)
  Description: This function computes all three autocorrelation descriptors for the sequences given in ‘seq_file’ for the AAindex accession numbers given in ‘aaindex_file’, at a specific value of d given in ‘dval’.
- **Autocorr_NT:** usage: autocorr_aa_n(seq_file, aaindex_file, dval, n)
  Description: This function computes all three autocorrelation descriptors for the N-terminal of sequences given in ‘seq_file’ for the AAindex accession numbers given in ‘aaindex_file’, at a specific value of d given in ‘dval’. User has to mention length of N-terminal in ‘n’.
- **Autocorr_CT:** usage: autocorr_aa_c(seq_file, aaindex_file, dval, m)
  Description: This function computes all three autocorrelation descriptors for the C-terminal of sequences given in ‘seq_file’ for the AAindex accession numbers given in

‘aaindex_file’, at a specific value of d given in ‘dval’. User has to mention length of C-terminal in ‘m’.

- **Autocorr_REST:** usage: autocorr_aa_rest(seq_file, aaindex_file, dval, n, m)
  Description: This function computes all three autocorrelation descriptors for the remainder of sequences given in ‘seq_file’ when N-terminal and C-terminal are not considered, for the AAindex accession numbers given in ‘aaindex_file’, at a specific value of d given in ‘dval’. User has to mention length of both N-terminal and C-terminal in ‘n’ and ‘m’ variables respectively.
- **Autocorr_SPLIT:** usage: autocorr_aa_split(seq_file, aaindex_file, dval, s)
  Description: This function computes all three autocorrelation descriptors for the split parts of the sequences given in ‘seq_file’ for the AAindex accession numbers given in ‘aaindex_file’, at a specific value of d given in ‘dval’. User has to mention number of split parts in ‘s’.

  Conditions: dval<=min(L-1, 30) where L is the length of the sequence/ sub-sequence for which autocorrelation descriptors have to be calculated. Seq_file should be a new-line separated .csv file. aaindex_file should be a comma separated .csv file.

1.5.2 Conjoint Triad Descriptors (CTD)

Conjoint triad descriptors are proposed by J.W. Shen et.al. These descriptors explains the features of protein pairs based on the classification of amino acids. The 20 amino acids were clustered into several classes according to their dipoles and volumes of the side chains in the following manner (**Dong, Jie, et al. *Journal of cheminformatics* 10.1 (2018): 16.**):-

Group 1: A, G, V
Group 2: I, L, F, P
Group 3: Y, M, T, S
Group 4: H, N, Q, W
Group 5: R, K
Group 6: D, E
Group 7: C

The conjoint triad descriptors considers the property of amino acid along with its adjacent amino acids as one single unit of three amino acids. Triad of three amino acids belonging to same group are identical in nature, such as RCE and KCD are identical in nature. Protein sequence can be represented as a binary space (V, F) where, V is the vector space of the sequence features, and each feature v_i represents a triad type; F is the frequency vector corresponding to V, and f_i is the frequency of type v_i appearing in the protein sequence. For the amino acids that have been catalogued into seven classes, the size of V should be 7×7×7; thus i = 1,2, ..., 343. Long protein would have a large value of f_i as compared to small sequences thus creating problem while comparing two heterogeneous proteins. Thus, we will normalize f_i in following manner:-

$$f\_norm_i = (f_i - \min(f_1, f_2, f_3 \ldots\ldots f_{343})) / \max(f_1, f_2, f_3 \ldots\ldots f_{343})$$

The python functions for conjoint triad calculation of these descriptors are described below:

- **CTC whole sequence:** CTC(input_filename)
  Description: This function computes the descriptor value f_i corresponding to each triad group as explained above(Group 1, 2....343 ----111, 112….777) in the protein sequence.

- **CTC_NT:** CTC_NT(input_filename, n)
  Description: This function computes the descriptor value f_i corresponding to each triad group as explained above(Group 1, 2....343 ----111, 112….777) in the N-terminal of the protein sequence. N-terminal value has to be submitted by the user.

- **CTC_CT:** CTC_CT(input_filename, m)
  Description: This function computes the descriptor value f_i corresponding to each triad group as explained above(Group 1, 2....343 ----111, 112….777) in the C-terminal of the protein sequence. C-terminal value has to be submitted by the user.

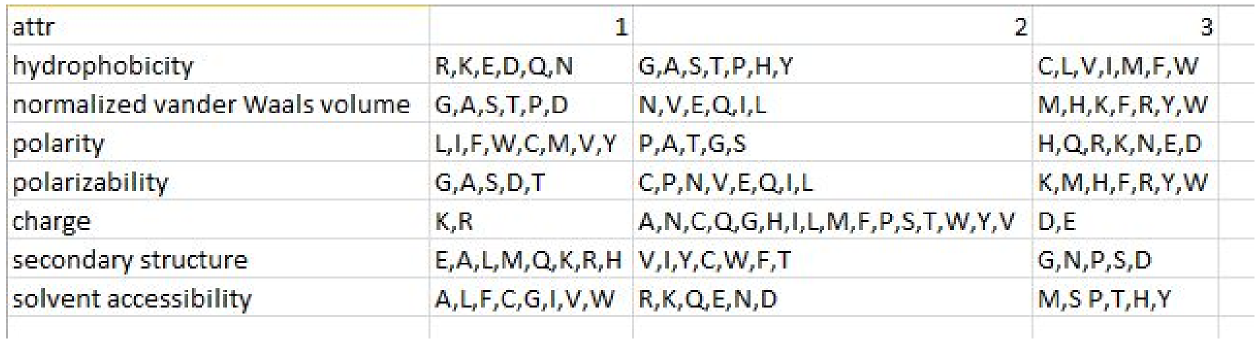
- **CTC_REST:** CTC_REST(input_filename, n, m)
  Description: This function computes the descriptor value f_i corresponding to each triad group as explained above(Group 1, 2....343 ----111, 112….777) in the remaining sequence after removing N- and C-terminal of the protein sequence. N- and C-terminal values have to be submitted by the user.

- **CTC_SPLIT:** CTC_SPLIT(input_filename, s)
  Description: This function computes the descriptor value f_i corresponding to each triad group as explained above(Group 1, 2....343 ----111, 112….777) in the sub-sequence of the protein sequence. Split value has to be submitted by the user and will generate equivalent sub-sequences from main sequence in continuous manner..

1.5.3 Composition enhanced Transition and Distribution (CeTD):

First step is to encode(convert) the peptide/protein sequence on the basis of their group value. All the values are present in aa_aatr_group.csv file. Then occurrence (composition) of each residue within should be calculated using formula:

$$Composition = \frac{Frequency\ of\ same\ Residue * 100}{Length\ of\ peptide\ sequence} \tag{14}$$



attr	1	2	3
hydrophobicity	R,K,E,D,Q,N	G,A,S,T,P,H,Y	C,L,V,I,M,F,W
normalized vander Waals volume	G,A,S,T,P,D	N,V,E,Q,I,L	M,H,K,F,R,Y,W
polarity	L,I,F,W,C,M,V,Y	P,A,T,G,S	H,Q,R,K,N,E,D
polarizability	G,A,S,D,T	C,P,N,V,E,Q,I,L	K,M,H,F,R,Y,W
charge	K,R	A,N,C,Q,G,H,I,L,M,F,P,S,T,W,Y,V	D,E
secondary structure	E,A,L,M,Q,K,R,H	V,I,Y,C,W,F,T	G,N,P,S,D
solvent accessibility	A,L,F,C,G,I,V,W	R,K,Q,E,N,D	M,S P,T,H,Y

There are 9- possibilities that two residues lying next to each other. This is called enhanced transition (E-Transition). 11,12,13,21,22,23,31,32,33 are the 9 possibilities.

Distribution is the measure of presence of particular residue in 5 quartile (0%, 25%, 50%, 75%, 100%) of the peptide sequence.

- **CeTD:** Usage: ctd(input_filename)
  Description: This function will compute the atomic composition of each amino acid residue from whole sequence of a protein using Eq.14
- **CeTD_NT**: Usage: ctd_nt(input_filename, n)
  Description: This function computes the atomic composition of each amino acid residue selected from N-terminal using Eq.14, user need to provide number of N-terminal residues ***n*** to be used for calculating.
- **CeTD_CT**: Usage: ctd_ct(input_filename, m)
  Description: This function computes the atomic composition of each amino acid residue selected from C-terminal using Eq.14, user need to provide number of N-terminal residues ***m*** to be used for calculating.
- **CeTD_REST**: Usage: ctd_rest (input_filename, n, m)
  Description: This function computes the atomic composition of remaining residues of a sequence after cleaving ***n*** residues from N-Terminal and ***m*** residues from C-Terminal using Eq.14.
- **CeTD_SPLIT** : Usage: aac_split (input_filename, s)
  Description: This function computes the atomic composition of each portion after splitting sequence in ***s*** portions using Eq.14. This function is important to compute amino acid composition of different portion of a protein.

1.5.4 Pseudo Amino Acid Composition (PAAC): This group of descriptors has been proposed by K.C. Chou. Let $H_1^o(i)$be hydrophobicity values for i = 1,2,3,......20, $H_2^o(i)$ be the hydrophilicity values for i = 1,2,3,......20, and $M^o(i)$ be the side chain masses of the 20 natural amino acids. They are converted to the following quantities by a standard conversion:

$$H_1(i) = \frac{H_1^o(i) - \frac{1}{20}\sum_{i=1}^{20} H_1^o(i)}{\sqrt{\frac{\sum_{i=1}^{20}\left[H_1^o(i) - \frac{1}{20}\sum_{i=1}^{20} H_1^o(i)\right]^2}{20}}} \tag{15}$$

Where, H_2^o (i) and $M^o(i)$ are normalized as$H_2(i)$ and $M(i)$ in the same manner.

1.5.5 Amphiphilic Pseudo Amino Acid Composition (APAAC): Amphiphilic Pseudo-Amino Acid Composition (APAAC) is described as:

$$P_c = \frac{f_c}{\sum_{r=1}^{20} f_r + w \sum_{j=1}^{2\lambda} \tau_j} \quad (1 < c < 20) \tag{16(i)}$$

$$P_c = \frac{\omega \tau_u}{\sum_{r=1}^{20} f_r + w \sum_{j=1}^{2\lambda} \tau_j} \quad (21 < \mathrm{u} < 20 + 2\lambda) \tag{16(ii)}$$

where w is the weighting factor which is set as (w= 0.5),as described in Chou's work (Chou, 2001).

1.5.6 Quasi-Sequence Order (QSO): The quasi-sequence-order descriptors are proposed by K.C. Chou, et.al. Quasi-sequence-order Descriptors obtained from the distance matrix between the 20 amino acids. Schneider-Wrede physicochemical distance matrix (Schneider and Wrede, 1994) and the chemical distance matrix by Grantham (Grantham, 1974) are used by Kuo-Chen Chou.

For each type of amino acid, a quasi-sequence-order descriptor can be described as:

$$X_r = \frac{f_r}{\sum_{r=1}^{20} f_r + w \sum_{d=1}^{nlag} \tau_d} \quad r = 1,2,....20 \tag{17}$$

where *fr* is the normalized occurrence of amino acid type *r,*and *w* is a weighting factor (*w* = 0.1), *nlag* and $\tau_d$is the same which was described above. These are the first 20 quasi-sequence-order descriptors. The other 30 quasi-sequence-order descriptors are defined as:

$$X_d = \frac{w\tau_d - 20}{\sum_{r=1}^{20} f_r + w \sum_{d=1}^{nlag} \tau_d} \quad d = 21,22,....30 + nlag \tag{18}$$

1.5.7 Sequence Order Coupling Number (SOCN): The *d*-th rank sequence-order-coupling number is described as:

$$\tau_d = \sum_{i=1}^{N-d} (d_{i,\, i+d})^2 \quad d = 1,2,3,.....,nlag \tag{19}$$

where *d i,i+d* is the number in a given distance matrix explaining a distance between the two amino acids *i*and *i+d*, nlag is the maximum value of the lag, and N denotes the length of a protein or peptide sequence.

Note: The length of the protein or peptide sequence must be not less than the maximum value of *nlag*.

2.0 Binary Profiles

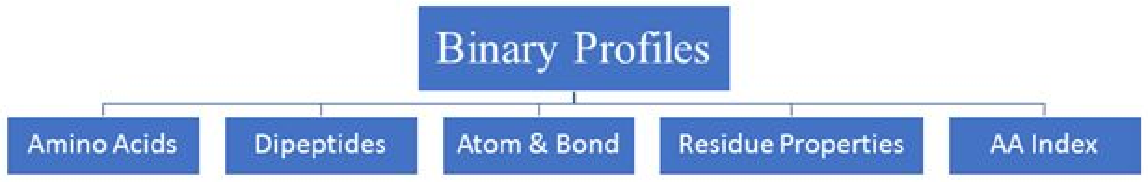


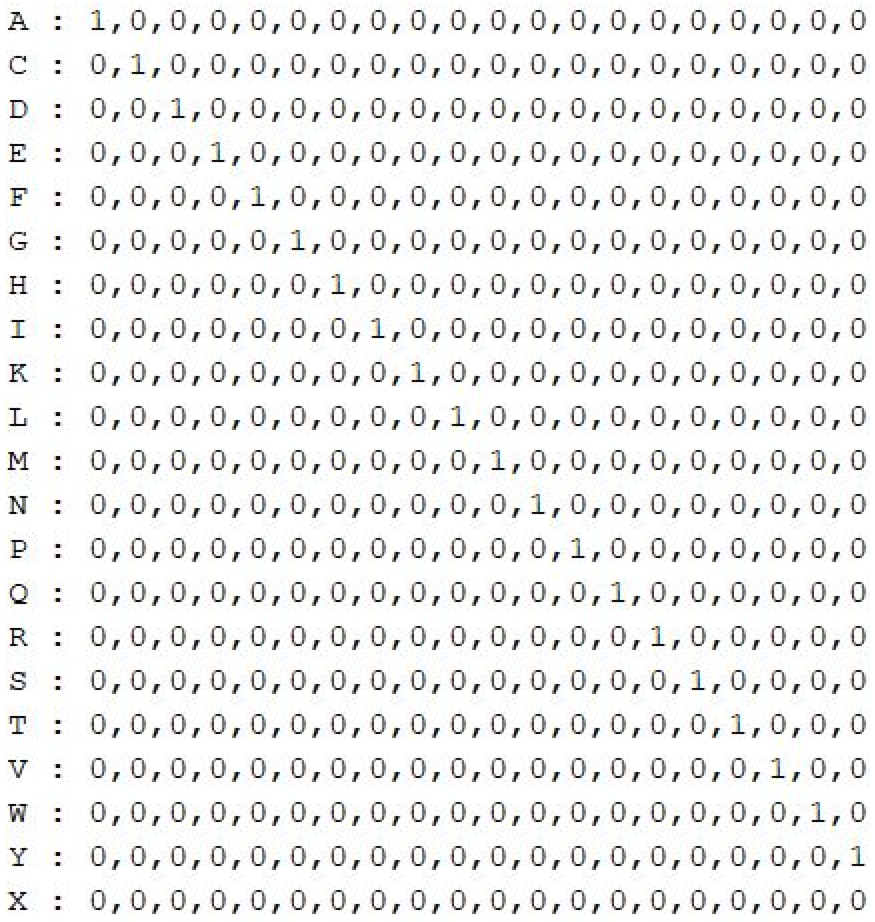
Figure 2: This flowchart shows different types of protein/peptide Binary profiles based features.

Function Title	**Description**
AABP	To calculate Amino acid Binary Profile of a peptide
AABP_NT	To calculate Amino acid Binary Profile of N-terminal residues defined by user
AABP_CT	To calculate Amino acid Binary Profile of C-terminal residues defined by user
AABP_rest	To calculate Amino acid Binary Profile of remaining residue from N-Terminal and C-Terminal Residues defined by user
AABP_split	To calculate Amino acid Binary Profile by splitting peptide into fragments defined by user
DPBP	To calculate Dipeptide Binary Profile of a peptide
DPBP_NT	To calculate Dipeptide Binary Profile of N-terminal residues defined by user
DPBP_CT	To calculate Dipeptide Binary Profile of C-terminal residues defined by user
DPBP_rest	To calculate Dipeptide Binary Profile of remaining residue from N-Terminal and C-Terminal Residues defined by user

DPBP_split	To calculate Dipeptide Binary Profile by splitting a peptide into fragments defined by user
ATBP	To calculate Atomic Binary Profile (% of Carbon, Hydrogen, Nitrogen, Oxygen, Sulphur content) of a peptide
ATBP_NT	To calculate Atomic Binary Profile of N-terminal residues defined by user
ATBP_CT	To calculate Atomic Binary Profile of C-terminal residues defined by user
ATBP_rest	To calculate Atomic Binary Profile of remaining residue from N-Terminal and C-Terminal Residues defined by user
ATBP_split	To calculate Atomic Binary Profile by splitting a peptide into fragments defined by user
BTBP	To calculate Bond Binary Profile (% of Carbon, Hydrogen, Nitrogen, Oxygen, Sulphur content) of a peptide
BTBP_NT	To calculate Bond Binary Profile of N-terminal residues defined by user
BTBP_CT	To calculate Bond Binary Profile of C-terminal residues defined by user
BTBP_rest	To calculate Bond Binary Profile of remaining residue from N-Terminal and C-Terminal Residues defined by user
BTBP_split	To calculate Bond Binary Profile by splitting a peptide into fragments defined by user

2.1 Amino Acids

This function generates binary equivalent of each residues. The following table consists of 20-vector binary profile for each residue. Peptide/protein sequences are replaced by their equivalent binary profile.

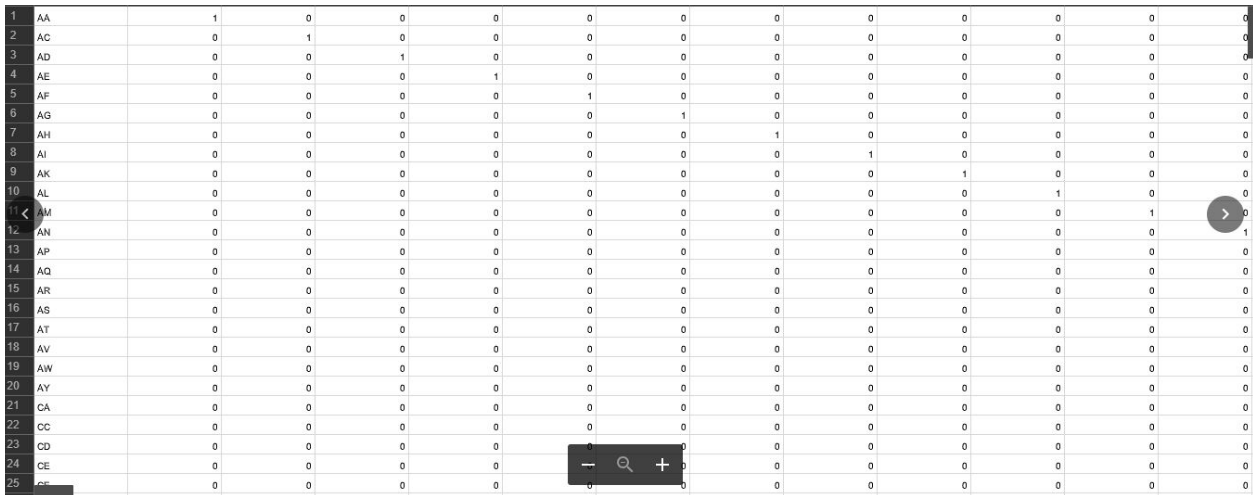


```
A : 1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
C : 0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
D : 0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
E : 0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
F : 0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
G : 0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0
H : 0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0
I : 0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0
K : 0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0
L : 0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0
M : 0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0
N : 0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0
P : 0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0
Q : 0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0
R : 0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0
S : 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0
T : 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0
V : 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0
W : 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0
Y : 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1
X : 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
```

- **AABP**: Usage: aabp (input_filename)
  Description: This function generates binary profile for residues from whole sequence of a protein.
- **AABP_NT**: Usage: aabp_nt (input_filename, n)
  Description: This function generates binary profile for residues selected from N-terminal, user need to provide number of N-terminal residues ***n*** to be used for calculating.
- **AABP_CT**: Usage: aabp_ct (input_filename, m)
  Description: This function generates binary profile for residues selected from C-terminal, user need to provide number of N-terminal residues ***m*** to be used for calculating.
- **AABP_REST**: Usage: aabp_rest (input_filename, n, m)
  Description: This function generates binary profile of remaining residues of a sequence after cleaving ***n*** residues from N-Terminal and ***m*** residues from C-Terminal.
- **AABP_SPLIT**: Usage: aabp_split (input_filename, s)
  Description: This function generates binary profile of each portion after splitting sequence in ***s*** portions. This function is important to generates binary profile of different portion of a protein.

2.2 Dipeptides

The Dipeptide binary profiles are generated by this function by replacing residues by their equivalent 400-size vector.The snapshot is as below.



1	AA	1	0	0	0	0	0	0	0	0	0	0	0
2	AC	0	1	0	0	0	0	0	0	0	0	0	0
3	AD	0	0	1	0	0	0	0	0	0	0	0	0
4	AE	0	0	0	1	0	0	0	0	0	0	0	0
5	AF	0	0	0	0	1	0	0	0	0	0	0	0
6	AG	0	0	0	0	0	1	0	0	0	0	0	0
7	AH	0	0	0	0	0	0	1	0	0	0	0	0
8	AI	0	0	0	0	0	0	0	1	0	0	0	0
9	AK	0	0	0	0	0	0	0	0	1	0	0	0
10	AL	0	0	0	0	0	0	0	0	0	1	0	0
11	AM	0	0	0	0	0	0	0	0	0	0	1	0
12	AN	0	0	0	0	0	0	0	0	0	0	0	1
13	AP	0	0	0	0	0	0	0	0	0	0	0	0
14	AQ	0	0	0	0	0	0	0	0	0	0	0	0
15	AR	0	0	0	0	0	0	0	0	0	0	0	0
16	AS	0	0	0	0	0	0	0	0	0	0	0	0
17	AT	0	0	0	0	0	0	0	0	0	0	0	0
18	AV	0	0	0	0	0	0	0	0	0	0	0	0
19	AW	0	0	0	0	0	0	0	0	0	0	0	0
20	AY	0	0	0	0	0	0	0	0	0	0	0	0
21	CA	0	0	0	0	0	0	0	0	0	0	0	0
22	CC	0	0	0	0	0	0	0	0	0	0	0	0
23	CD	0	0	0	0	0	0	0	0	0	0	0	0
24	CE	0	0	0	0	0	0	0	0	0	0	0	0
25	[illegible]	0	0	0	0	0	0	0	0	0	0	0	0

Here gap is also taken in account. If no gap is present then 0 value should passed by user and otherwise needed gap should be entered.

- **DPBP**: Usage: dpbp (input_filename)
  Description: This function will generates Dipeptide binary profile for residues from whole sequence of a protein.
- **DPBP_NT:** Usage: dpbp_nt (input_filename, n)
  Description: This function generates Dipeptide binary profile for residues selected from N-terminal, user need to provide number of N-terminal residues ***n*** to be used for calculating.
- **DPBP_CT:** Usage: dpbp_ct (input_filename, m)
  Description: This function generates Dipeptide binary profile for residues selected from C-terminal, user need to provide number of N-terminal residues ***m*** to be used for calculating.
- **DPBP_REST:** Usage: dpbp_rest (input_filename, n, m)
  Description: This function generates Dipeptide binary profile of remaining residues of a sequence after cleaving ***n*** residues from N-Terminal and ***m*** residues from C-Terminal.

- **DPBP_SPLIT:** Usage: dpbp_split (input_filename, s)
  Description: This function generates Dipeptide binary profile of each portion after splitting sequence in ***s*** portions. This function is important to generates binary profile of different portion of a protein.

2.3 Atom & Bond:

This function computes the binary profile corresponding to atomic and bond composition of each amino acid residue of the peptide sequence. Atomic composition is percentage of Carbon, Hydrogen, Nitrogen, Oxygen and Sulphur atoms present in a peptide sequence. These five atoms form a size 5 binary vector. Their combinations form binary profile of each residue. For example residue of Alanine(A) contains 13 atoms in total. Thus binary profile of 'A' will be of size 13*5=65.

- **ATBP**: Usage: atbp (input_filename)
  Description:This function generates Atomic binary profile from whole sequence of a protein.
- **ATBP_NT:** Usage: atbp_nt (input_filename, n)
  Description: This function generates Atomic binary profile for residues selected from N-terminal, user need to provide number of N-terminal residues ***n*** to be used for calculating.
- **ATBP_CT:** Usage: atbp_ct (input_filename, m)
  Description: This function generates Atomic binary profile for residues selected from C-terminal, user need to provide number of N-terminal residues ***m*** to be used for calculating.
- **ATBP_REST:** Usage: atbp_rest (input_filename, n, m)
  Description: This function generates Atomic binary profile for remaining residues of a sequence after cleaving ***n*** residues from N-Terminal and ***m*** residues from C-Terminal.
- **ATBP_SPLIT:** Usage: atbp_split (input_filename, s)
  Description: This function generates Atomic binary profile of each portion after splitting sequence in ***s*** portions.

Bond binary profile is made based upon canonical smile (from PubChem) for each Amino Acid. Four kinds of bond considered c(cyclic), benzene ring(b), single bond(-) and double bond (=). Corresponding to these bonds binary vector is created.

- **BBP**: Usage: bbp (input_filename)
  Description:This function generates Bond binary profile from whole sequence of a protein.
- **BBP_NT:** Usage: bbp_nt (input_filename, n)
  Description: This function generates Bond binary profile for residues selected from N-terminal, user need to provide number of N-terminal residues ***n*** to be used for calculating.
- **BBP_CT:** Usage: bbp_ct (input_filename, m)

Description: This function generates Bond binary profile for residues selected from C-terminal, user need to provide number of N-terminal residues ***m*** to be used for calculating.

- **BBP_REST:** Usage: bbp_rest (input_filename, n, m)
  Description: This function generates Bond binary profile for remaining residues of a sequence after cleaving ***n*** residues from N-Terminal and ***m*** residues from C-Terminal.
- **BBP_SPLIT:** Usage: bbp_split (input_filename, s)
  Description: This function generates Bond binary profile of each portion after splitting sequence in ***s*** portions.

2.4 Residue Properties

This function outputs a binary profile of each sequence which convey where a particular physicochemical property is present in a sequence.

- **Binary Profile of entire sequence:**
  Usage: bp_phychem_all(input_file, featureNum)
  Description: This function calculates the binary profile of a particular physicochemical property for each input sequence. The user can enter the input file containing these sequences and the feature number.
- **Binary Profile of N-terminal residues:**
  Usage: bp_phychem_NT(input_file,featureNum,n)
  Description: This function outputs the binary profile of desired physicochemical property by considering only 'n' N Terminal residues.
- **Binary Profile of C-terminal residues:**
  Usage: bp_phychem_CT(input_file,featureNum,n)
  Description:This function outputs the binary profile of desired physicochemical property by considering only 'n' C-Terminal residues.
- **Binary Profile of rest residues:**
  Usage: bp_phychem_rest(input_file,featureNum, m, n)
  Description:This function outputs the binary profile of desired physicochemical property by removing 'n' N Terminal residues and 'm' C Terminal residues and considering only the residues left after these removals.
- **Binary Profile of split subsequences:**
  Usage: bp_phychem_split(input_file,featureNum,SPLIT)
  Description:This function splits the sequence into 'SPLIT' parts and then iteratively gives binary profile of each split subsequence.

2.5 AA Index

Usage: phychem_AAI(input_file, AAIndices)

This function gives the binary profile of input AA Indices. If normalised score of AAIndex value of a particular residue is negative, the function assigns ‘0’ to that residue otherwise assigns ‘1’. The user can enter the input file along with a comma separated file containing multiple desired AA Indices from https://www.genome.jp/dbget/AAindex/list_of_indices. This hyperlink lists out all the indices and the same can be input into the function.

- **AA Index**:phychem_AAI(input_file, AAIndices)
  Description:This function gives the binary profile of input AA Indices.

3.0 Evolutionary Information

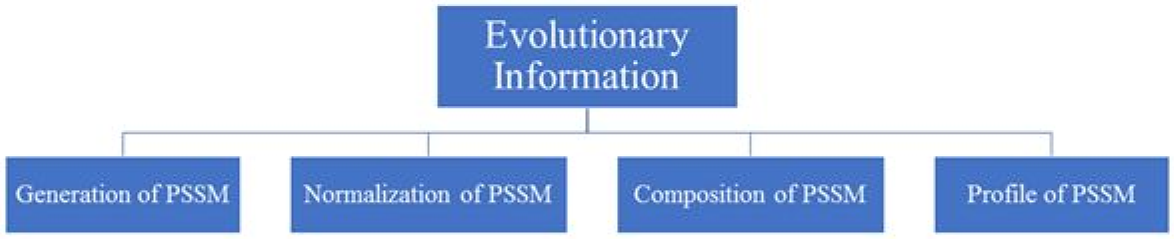


Figure 3: This flowchart shows different types of protein/peptide Evolutionary Information based features.

3.1 Generation of PSSM

This matrix is generated by using psi-blast against databases (nr or swissprot). The resultant matrix consists information of evolutionary conservation of elements of type x(i,j), where j is a residue at position ‘i’.

3.2 Normalization of PSSM

Various Normalization techniques are there to normalize the PSSM profile.

- **pssm_n1** : Due to the large number of variation in the value of PSSM matrix, it is necessary to normalize it. Each element of matrix is normalized by 1/(1+e-x).
- **pssm_n2** : This is the second technique to normalize the elements of PSSM matrix using the formula (num - min)/(max - min).
- **pssm_n3** : This is the third technique to normalize the matrix using the formula (num - min)*100/(max - min).
- **pssm_n4** : This is the fourth technique to normalize the PSSM profile using the formula 1/(1+e-(x/100).

3.3 Composition of PSSM

This function results the vector of 400 size. It calculates the frequency of amino acid composition corresponding to residue of peptide/protein sequence. Each column consists of 20 values.

3.4 Profile of PSSM

This matrix is generated by using psi-blast against databases (nr or swissprot). The resultant matrix consists information of evolutionary conservation of elements of type x(i,j), where j is a residue at position ‘i’.

In order to generate PSSM profile of different portions of an amino acid sequence, we have developed number of python function (brief description is given below). In web server user may select portion of sequence for calculating protein features.

- **PSSM_PROFILE**: Usage: pssm_profile(input_filename)
  Description: This function will generate PSSM profile from whole sequence of a protein.
- **PSSM_NT**: Usage: pssm_nt (input_filename, n)
  Description: This function will generate PSSM profile of residues selected from N-terminal. User need to provide number of N-terminal residues ***n*** to be used for calculating.
- **PSSM_CT**: Usage: pssm_ct (input_filename, m)
  Description: This function will generate PSSM profile of residue selected from C-terminal. User need to provide number of C-terminal residues ***m*** to be used for calculating.
- **PSSM_REST**: Usage: pssm_rest (input_filename, n, m)
  Description: This function will generate PSSM profile of remaining residues of a sequence after cleaving ***n*** residues from N-Terminal and ***m*** residues from C-Terminal.

4.0 Structure

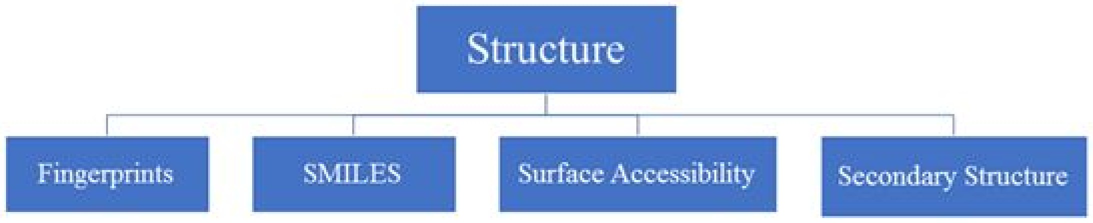


Figure 4: This flowchart shows different types of protein/peptide Structure based features.

4.1 Fingerprints:

This module was developed to calculate different types of fingerprints descriptors. The fingerprints were calculated using PaDEL software, which is java based software. PaDEL software provides 10 different types of fingerprints types which in total provide 14,532 fingerprint values. These fingerprints are calculated using mainly The Chemistry Development Kit (CDK).
Along with CDK, other fingerprints present are Pubchem fingerprints, MACCS fingerprints, Klekota-Roth fingerprints. Fingerprints have been used as an important type of feature in various prediction methods developed previously in literature.

Usage: Here user needs to upload its molecular structure in PDB file format for calculating the fingerprints.

4.2 SMILES

SMILES stands for Simplified Molecular Input Line Entry System. It is a type of line notation for representing various molecules and reactions. It contains the same information as the extended data tables consists of. One of the advantage of using it is that it is easy to understand since it is a linguistic construct rather than a computer data structure. Also, the SMILES format takes 50-70% less space in comparison to other way of representing the information as well as required lesser time for processing the information. SMILES notation is represented by series of characters and no spaces are present in between the characters.
SMILES notation follows five simple rules required for its encoding which are corresponding to atoms, bonds, branches, ring closures and disconnections. Detailed description of the SMILES notations can be obtained at http://www.daylight.com/dayhtml/doc/theory/theory.smiles.html .

Usage: Here, SMILES format were generated using openbabel software, where users are required to upload their structure in PDB file format in the SMILES page of pfeature in order to get the desired output.

4.3 Surface Accessibility

Accessible molecular surface or solvent-exposed area is defined as the area of an atom which can be touched by water molecule. Contact surface area and atoms chemical properties play an important role in modeling side chain conformations in proteins, structure and functional annotation of biological molecules. Here we have developed a module, which calculates the Relative Accessibility Area (RSA) using NACCESS software. The software requires PDB structure as an input and calculates relative accessible area. The output provided by the software shows value in percentage. In general values ranges between 0-100%; however, for some residues values go beyond 100%. In general, residues showing value less than 20% are said to buried whereas residues showing value above 20% are said to be exposed

Usage: Here, user needs to upload their structure in PDB file format and the server will calculate the relative accessibility area as an output.

4.4 Secondary Structure

Secondary structure refers to the interaction of hydrogen bond donor and acceptor residues of the repeating peptide unit. It plays an important role in protein structure prediction and protein folding. The two most important element of secondary structure are alpha helix and beta sheet. However coils are also considered as an important type of secondary structure in many cases. There are many software present in the literature which has been developed to predict the type of secondary structure. Since secondary structure elements represents an important type of feature, we have developed a module which calculates the percent average secondary structure element present in the input structure file. We have used DSSP software, which assigns the secondary structure state of the residue.

Usage: In order to calculate the percent average secondary structure element, user needs to upload the PDB file on to the server.

5.0 Pattern

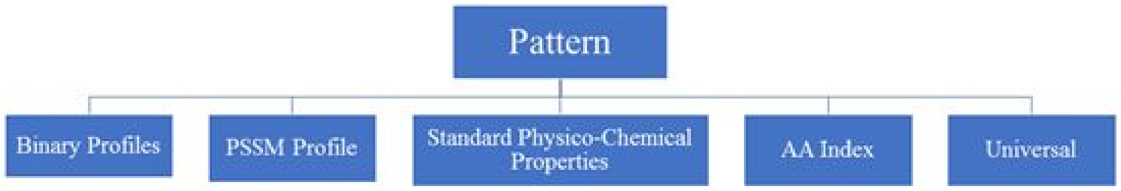


Figure 5: This flowchart shows different types of protein/peptide Pattern based features.

5.1 Binary Profiles

This function is used to compute binary profile for the patterns of protein and peptide sequences. The patterns generated are in different window size. The window size will always be an odd number to generate equal size of the patterns. An extra 'X' is added in the starting and end of the sequence to make equal size patterns. The binary pattern is generated for the pattern generated.

5.2 PSSM Profile

This function is used to compute PSSM for patterns of protein and peptide sequences. Here the patterns are generated from PSSM matrix in different window size. The window size will always be an odd number to generate equal size of patterns. Here extra 'X' is added in the starting and end of the sequence to make the equal size patterns, so the vector size will be 21.

5.3 Standard PhysicoChemical Properties

This function generates patterns of desired length within sequences and then calculates the standard physicochemical properties(refer to section 1.2.1) of each generated pattern.

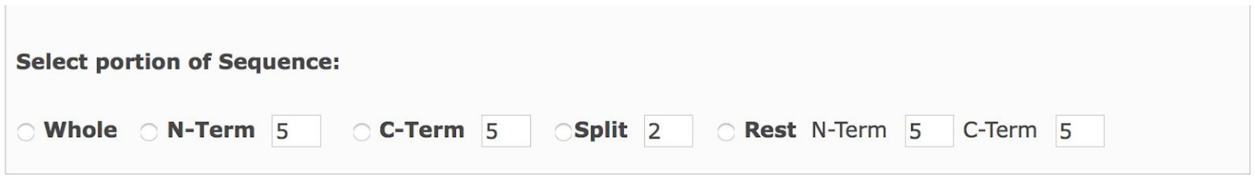
5.4 AA Index

This function generates patterns of desired length and then calculates average desired AA Index value for each generated pattern.

5.5 Universal

This function will generate patterns for any type of string like secondary structure, surface accessibility. These patterns are generated in sliding window manner and are of defined length. Additional 'X' is added on both the sides of the peptide sequence which results in the generation of equal length pattern.

6.0 Portion of a Sequence


Select portion of Sequence:

○ **Whole** ○ **N-Term** 5 ○ **C-Term** 5 ○ **Split** 2 ○ **Rest** N-Term 5 C-Term 5

6.1 Whole amino acid sequence

This option allow users to compute features of a protein from whole sequence. This option is important for user when user wish to understand overall property of a protein or peptide. Most of methods developed in past use whole amino acid sequence of a protein.

6.2 N-Terminal

It has been observed in past that N-terminal of a protein is responsible for its function. For example most of classical secretory proteins contain a signal peptide. A short peptide (16-30 amino acids) present at the N-terminus of the majority of proteins that are destined towards the secretory pathway. Signal peptides are not only found in N-terminal of secretory proteins but found in number of other class of protein. Pfeature allow user to compute wide range of features in selected region (N-terminal) of a protein. One of the advantage of in selecting region is that user can generate both composition as well as binary profile as length of selected region is fixed **(BMC Bioinformatics 2007, 8:263 & BMC Bioinformatics 2010, 11:S19)**.

6.3 C-Terminal

It has been observed in past that C-terminal of a protein is responsible for its function. Normally. N-terminus of a protein often contains targeting signals, the C-terminus can contain retention signals for protein sorting. The most common endoplasmic reticulum retention signal is the amino acid sequence KDEL or HDEL at the C-terminus. This keeps the protein in the endoplasmic reticulum and prevents it from entering the secretory pathway. Pfeature allow user to compute wide range of features in selected region (C-terminal) of a protein. One of the advantage of in selecting region is that user can generate both composition as well as binary profile as length of selected region is fixed (BMC Bioinformatics 2007, 8:263 & BMC Bioinformatics 2010, 11:S19).

6.4 Split

One of the major problem with composition based features is that that it present protein by limited features, it give only average features of whole sequence. In order to increase number of features to capture more information from a protein, split amino acid composition (SAAC) has been introduced (**J Biol Chem. 2006;281:5357-63)**. In this concept, amino acid is splitted in two or more than two portions then features of each portion is computed separately. For example is number of split is three then sequence will be divided in three portions (each portion have nearly same length). If whole protein have 20 (composition) features then splitted composition provides 60 (20 X 3) features.

6.5 Rest

As shown in above section both terminals (N- & C-) have important information so pfeature have provision to compute feature of N-terminal or C-terminal. In order to capture information or generating feature from remaining portion of proteins (after removing N-terminal and C-terminal residues). In case of rest option user need to select number of residues from N-terminal and C-terminal to be removed from protein for calculating features from rest of protein.