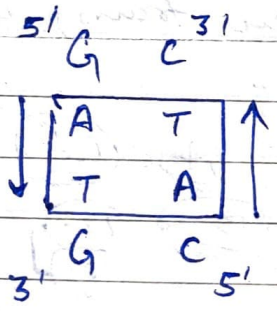
→ 10,000 → PCs → PCs

Feature

→ Features :- 28 Structural feature + 3 energy features

Values of these 28 str. features obtained by analysis of B-DNA crystal structure and values of energy features obtained from in-house softwares.

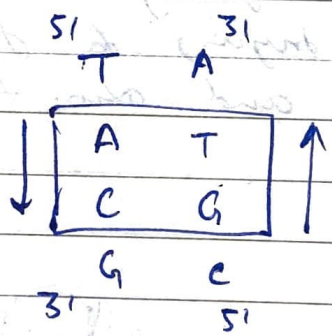
- Values for all 16 unique dinucleotides were obtained and averaged to get a particular value.
- Unique according to us need to be 16, however they are 10 because :-



AT = AT

↓

AT is one of the 10 unique dinucleotide.



AC = GT

↓

So instead of AC & GT to be 2 different dinucleotide they both are equivalent and hence together are 1 of the unique dinucleotide step.

all 10 are :-

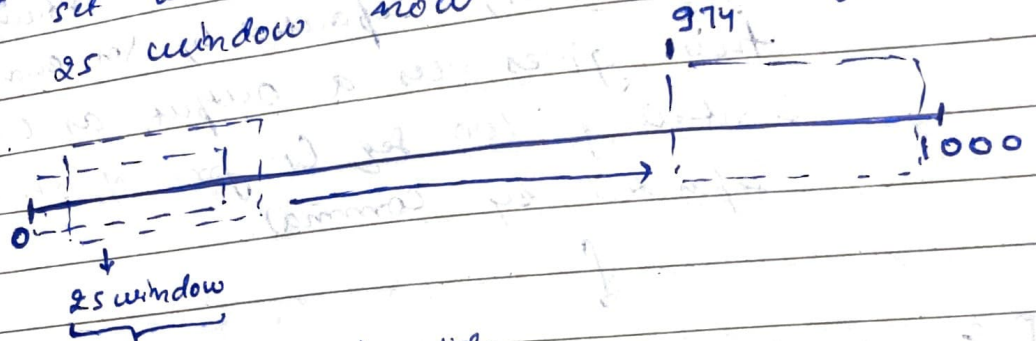
AA/TT, AT, AC/GT, AG/CT, TA, CG, CC/GG, GA/TC, CA/TG, GC.

→ Training dataset for TSS :- 12880 Seq.

Test dataset for CDS :- 6281 Seq.

All these Seq. were converted to 31 numeric sequences based upon the values of 10 dinucleotide steps.

→ each sequence of both TSS & CDS were of length 1001 nt., on this 1001 nt a window of 25 was set at index starting 0 and ending at 974 this 25 window now have 24 different dinucleotide steps.

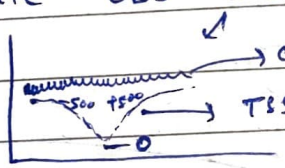


have pairs 24 dinucleotide.

fig S1 & S2

In TSS sequence

[S1 shows that all 31 parameters have some changes at TSS while CDS shows no change.



* all 31 parameters have discriminative power, however there scale were not same and also some had increasing effect other had decreasing (which might show (bias on observation) and also they were some-what correlated, hence a PCA need to be done. (after scaling/Normalisation).

Code language :-

→ We have say a input of 100 Seq.

→ from these 100 Seq., we choose 1st.

→ in this 1st Seq. we set a window of 25, for this window 31 parameters are calculated for 24

dinucleotide steps.

- These 31 parameters are then normalised (0 to 1).
- Now window shift towards right (so that can cover all 1000 nucleotide). i.e. the shifting & step 3 to end is repeated 975 times.

main.py Module

(1) → Read Seq. ** It input a .txt file containing 100 sequences, separated by \n character this gives us a output as list containing 100 seq. (in form of strings) separated by comma).

↓

i.e

['Seq 1', 'seq 2', 'Seq 3' ------ 'Seq 100']

↓

Now this list is then converted to a map/dict whose structure is:-

```
{ 0 : 'Seq 1   '
  1 : 'Seq 2   '
  2 : 'Seq 3   '
  |
  |
  |
 99 : 'Seq 100 ' }
```

Possible optimisation :-

* python might have a inbuilt function that can directly convert an input file to a dict.

FINAL OUTPUT :- SEQUENCE MAP (NCE)

(2) getParameterdetails :-

→ takes sequencemap as input

→ it goes to each seq. in sequencemap and perform following functions :-

(a) Calculate-parameter

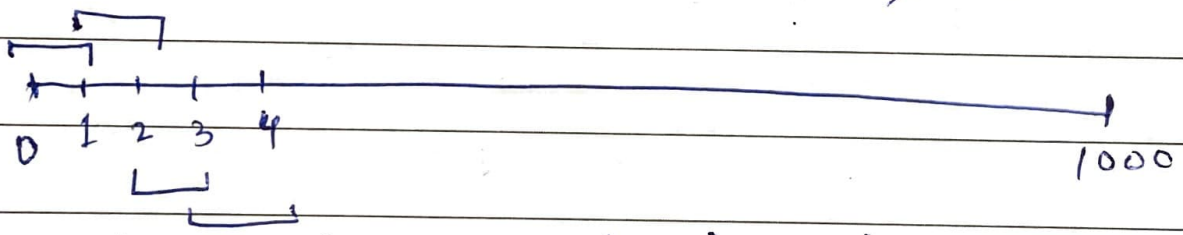
DOUBT :- Why we are considering a 25 length window. Why can't we just proceed with the whole sequence taking dimensionless steps.

if we take 25 windows then total values are 975 X 25 X 31, but if we take dimensionless step then we have 998 X 31.

① calculate parameters

- is run for each sequence (100)
- first it assigns scores / values to each ~~dict~~ dinucleotide & base pair.

for example: (A+T) will have diff score.



in this way, keeping in mind the 10 different ~~dicts~~ dinucleotide steps, we assign the scores to 31 parameters.

the result of this function is a param_map of the structure: (1000) for each 1000 bp.

```
{'a': [_, _ _ . . . . . ,]
 'b':
 .
 .
 'ae':
}
```

(2) calculateMovingAverages(param_map)

for every parameter in params_map:
say 'a': [[1000 values for each bp.]]

we take a window of 25 and take average of 1st 25 values for parameter 'a'.

then we move right by 1 step & repeat the above step.

'a': [sum of 0 to 24 / 25, sum of 1 to 25 / 25, sum of 2 to 26 / 25, ..., sum of 975-999 / 25]

'a': [975 values (averaged for a window of 25 upto 1000)]

doubt: why is this step needed?

this function returns moving_params_map of the structure:

975 X 25 X 31, but if we take
we have 998 X 31.

[@ calculate parameter, ⑥ calculate moving averages].
↳ for other

ⓒ normalise moving averages

input : { a : [975 values]
b
:
ae : [975 values] }
→ Moving params map.

→ It is just normalising all 975 values for each parameter.

★ optimisation :- python may have a inbuilt normalising function

output :- same as above, but all values (975) are in range 0 to 1 (after normalisation)

out put is named as Normalised map

*Optimisation :- In calculate-parameter function we are assigning values to every individual B-P i.e. (1000 BP for each sequences), we are extracting those values from supple table which unique codon-number step and values of them corresponding to all 31 parameters.

We can used this table in our optimisation.

→ Now we are going for PCA-regression Module.

Input is the following.

{ 0 : { a : [975 normalised value]
b :
ae : [ ]}

975-400-1
574

100 : { a : [975 Nor value]
ae : [975 Normed value] }

0 to 100 are the sequences
a to ae are the parameters.

* Extra task :- Calculate correlation for 31 parameters

* In the PCA the main motive is to reduce the no. of features / Parameters to some PCs.

for this there are 2 pipeline one is with regions upstream & downstream of TSS & other is with motif combination.

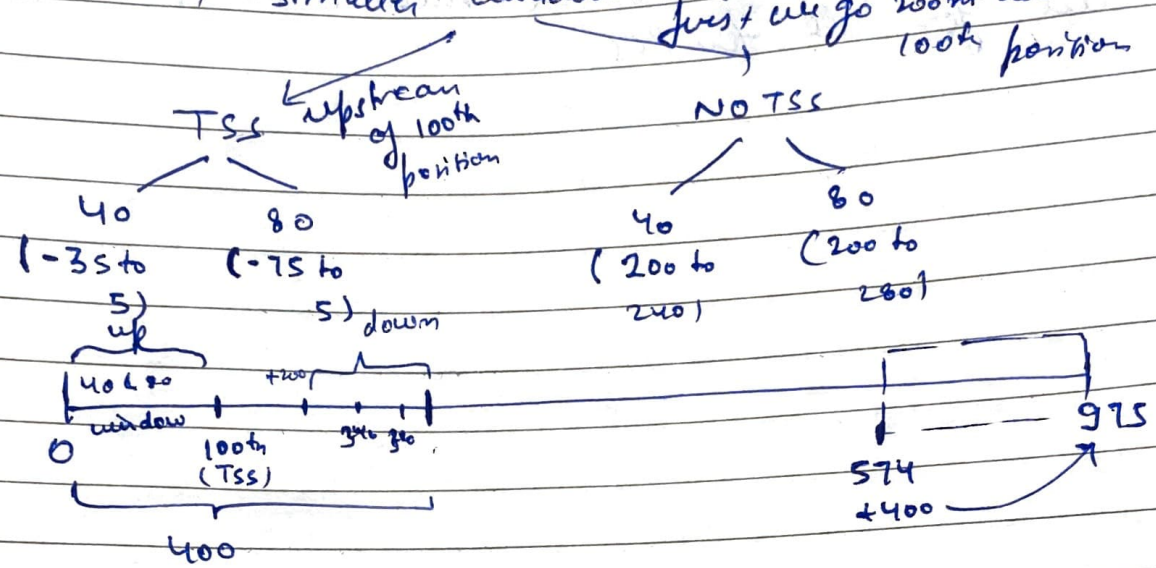
① We first discussion the 1st pipeline:

→ Here we input our sequence 1st !-

→ for this sequence ~~for [illegible]~~ all the 31 params have 975 values.

→ In this 975 value we are considering a 400 window

↳ in this 400 window, our TSS is at 100th (assumption), and then we divide this 400 window into 4 smaller windows.



→ this is repeated for 574 times (i.e. the whole sequence get covered).

↳ for this we have :-

for each window exhaust window function is called, this functions averages all the values of each 40/80 window, into TSS & NO TSS region into 31 parameters.

↳ ~~for 975, consider a 400 window, in this window~~ we assume that TSS lies on 100th position, from TSS,

for each sequence we get output in form of

Seq_40_map : { 0 : [31 para for TSS-40 window] - [31 para for NO TSS 40 window]

574 : [ ] [ ] }

SAME is for Seq_80_map & Seq_100_map

DOUBT :- Why there is a log-loss-map?

① Now all these 3 maps are sent for PCA & with Predict-PCA.

Training

↳ Now for all sequence we need to have one value for all the PCs.