**Bio Informatics Computing**

MUKHASIR SHAH SYED

109137268

Task: median string and motif finding.

Steps of operation:

1. Read input fasta sequence file "HMP-617.fa" (arbitary number of sequences and arbitary length each) and store that in a list(data structure).
2. Find the median string; using a priority queue of size 5, keep the 5 best median strings and their total\_distances; searching space is K^L, with 6-mer and DNA data, 4096 choices. Searching space optimization techniques have not be used.
3. Using 5 best median strings, find the corresponding motif consensus strings/score/positions; The original appearing order of the DNA sequences should be kept in the reported positions, i.e., s=(s1....st), where s1 is the starting position of motif in DNA sequence1 and so on where st is starting position of motif in DNA sequence t.

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Date : Mar-14-2015 Author: Mukhasir Shah Syed

\* Program written will be used to find median string and motif from given set of DNA sequences and

the length of the motif. We would report the best 5 median strings and their corresponding motif

consensus strings and positions.

\* INPUT:

\* A set of DNA sequences (arbitary length each) from "HMP-617.fa" file and motif length (L-mer)

\* of size 6.

\* OUTPUT:

\* We will display 5 best median strings(with total distance), corresonding motif consensus

\* strings(with conseensus scores), and motif positions (for each motif string).

\* Firstly, we would make list which contains all different sequences of median strings(4^L[L=6]).6-mer's

\* Secondly, we will read all DNA sequences from "HMP-617.fa" and place them into a list of key value pair.

\* As now we have got all median strings and all sequences from file, we will now loop through DNA

sequences for each Median string.

\* We will loop through the each substring of length as 6 in each DNA sequence and find out hamming

score and place median string value and score into a Dictonary.

\* After placing the value into dictionary we will sort disctionary based on the score which is placed

in value of key value pair in ascending order.

\* After sorting is completed, we can find best 5 median strings for which motif consensus string,

motif consensus score and motif position in each DNA sequences with less hamming score.

\* TotalDistance() method is used to get all median strings and their hamming score for each substring

of size "6" in each DNA sequence provided. We will pass median value and DNA sequence and dictionary

to store Median Values and their corresponding score. Score would be update with addition

of new score to old score. We check for duplicacy of Median Value and keep only unique value.

\* FindMotifPositions() method will find one motif position in each DNA sequence with low hamming score

and place that value and its position in sequence into list. We would pass median value from top 5

\* median values selected, DNA sequences and List of key value pair.

\* FindConsensus() method is used to find motif consensus score and motif consensus score.

\* To complie/run the code we have to create a Console Application using Visual Studio 2013

and compile/build this application which will create an executable file then run the

".exe" application to execute the code.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

using System;

using System.Collections.Generic;

using System.IO;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace MotifFind\_Median

{

class Program

{

static void Main(string[] args)

{

//Variables usd in the program further.

string motifCon\_String = string.Empty;

string FinalValue = string.Empty;

string HMP\_Label = string.Empty;

string HMP\_Seq = string.Empty;

//As we consider 6-mers to use as median, we will select length of string as 6.

int length = Convert.ToInt32(Math.Pow(4,6));

//List to place 4096 median values.

List<string> medians = new List<string>();

//Linq method which will create random strings.

Random random = new Random();

bool add = true;

int Size = 6;

string input = "ACGT";

StringBuilder builder = null;

char ch;

int i =0;

int j = 0;

//Loop through and create 6-Mer median values.

for (j = 0; j < length; j++)

{

add = true;

builder = new StringBuilder();

for (i = 0; i < Size; i++)

{

ch = input[random.Next(0, input.Length)];

builder.Append(ch);

}

//We will check for duplicacy, if present then we will not add.

foreach(string v in medians)

{

if(v == builder.ToString())

{

j = j - 1;

add = false;

break;

}

}

//If not duplicate then we wil add the value to list.

if(add)

{

medians.Add(builder.ToString());

}

}

//Sort the list such that it starts from AAAAAA

medians.Sort();

//Initialize KeyValuePair List for storing database sequences.

List<KeyValuePair<string, string>> HMP\_Data = new List<KeyValuePair<string, string>>();

//Loop throug database file to store the label and bases respectively in key-value pair format.

foreach (string Database in File.ReadLines(@"E:\Bio Informatics\Assignment 5\HMP-617.fa"))

{

if (Database.Contains(">"))

{

if (HMP\_Seq != string.Empty)

{

HMP\_Data.Add(new KeyValuePair<string, string>(HMP\_Label, HMP\_Seq));

HMP\_Seq = string.Empty;

}

HMP\_Label = Database;

}

else

{

HMP\_Seq += Database;

}

}

HMP\_Data.Add(new KeyValuePair<string, string>(HMP\_Label, HMP\_Seq));

//Initialize Dictionary for storing Median Values along with their respective hamming score.

Dictionary<string, int> MedianDistanceData = new Dictionary<string, int>();

//Loop thru each DNA sequence.

foreach(KeyValuePair<string,string> HMPMOtif\_Seq in HMP\_Data)

{

//Loop thru each median value of size 6.

foreach(string V in medians)

{

//This method is used to traverse through each median with each sequence and each substring of

//size "6" in the sequence. This will provide us a list with median value and their corresponding score.

TotalDistance(V, HMPMOtif\_Seq.Value, ref MedianDistanceData);

}

}

//Linq query to sort data present in disctionary based on value(which is score) in ascending order.

var items = from pair in MedianDistanceData

orderby pair.Value ascending

select pair;

//Initialize KeyValuePair List for storing median values with their score.

List<KeyValuePair<string, int>> med = new List<KeyValuePair<string, int>>(items);

//Variables used to store and display values.

string median\_string = string.Empty;

string motifps = string.Empty;

string cValueFind = string.Empty;

//Initialize KeyValuePair List for storing motif data and their positions.

List<KeyValuePair<string, int>> MotifPositionData = null;

//Loop through 5 best median values.

for (int counter = 0; counter < 5; counter++)

{

motifps = "";

MotifPositionData = new List<KeyValuePair<string, int>>();

//Loop through DNA Sequences

foreach (KeyValuePair<string, string> b in HMP\_Data)

{

//Build a string value with median string and its total distance

median\_string = "median string: " + med[counter].Key + " (tot\_dist = " + med[counter].Value + ")";

//This method is used to find motif string and its position in DNA sequence which have low

//hamming distance.

FindMotifPositions(med[counter].Key, b.Value, ref MotifPositionData);

}

cValueFind = string.Empty;

//Loop through all motif strings for a particular median string and form a string to display.

foreach (KeyValuePair<string, int> motifSeq in MotifPositionData)

{

//Concatinate motif strings into one string so that it could be passed to FindConsensus to find

//motif consensus string and motif consensus score.

cValueFind += motifSeq.Key + ";";

//Concatinate position and string to diplay in output.

motifps += motifSeq.Value + "(" + motifSeq.Key + "), ";

motifps = motifps.Trim();

}

//Build format and concatinate final value into the string to be displayed.

motifps = "motif positions/string s=(s1..st): " + "\n\t" + motifps.Substring(0,motifps.Length-1);

//This method is used to find out motif consensus string and motif consensus score by passing

//all motif strings found above.

motifCon\_String = FindConsensusScore(cValueFind);

//Concatinate all the string to display the final outpout format.

FinalValue += median\_string + "\n" + motifCon\_String + "\n" + motifps + "\n\n";

}

//Write final result into the text file.

File.WriteAllText(@"E:\Bio Informatics\Assignment 5\Result.txt", FinalValue);

}

/// <summary>

/// This method is used to find the Motif Consensus Score and Motif Consensus String from all

/// motif strings retrieved for each median string.

/// </summary>

/// <param name="sequences"></param>

/// <returns></returns>

public static string FindConsensusScore(string sequences)

{

string modifyString = string.Empty;

int score = 0;

string consensus\_string= string.Empty;

//We will split string based on splitter ";".

string[] motifstrings = sequences.Split(';');

//Loop through each character present in each string.

for (int j = 0; j < 6; j++)

{

//Loop through each motif string to get character at "i" position to form a string.

for (int i = 0; i < motifstrings.Length; i++)

{

if (motifstrings[i] != "")

{

modifyString += motifstrings[i][j].ToString();

}

}

//We will add occurence of character in each column and get numeric number and add the sum of all value to get consensus score.

score += modifyString.GroupBy(x => x).OrderByDescending(x => x.Count()).First().Count();

//We will get maximun occured character and concatinate all values to get consensus string.

consensus\_string += modifyString.GroupBy(x => x).OrderByDescending(x => x.Count()).First().Key.ToString();

modifyString = "";

}

//Form a value in string to be displayed in output.

string mcs = "motif consensus string: " + consensus\_string + " (consensus\_score = " + score.ToString() + ")";

//return final string.

return mcs;

}

/// <summary>

///

/// </summary>

/// <param name="medianValue"></param>

/// <param name="SequenceValue"></param>

/// <param name="MotifPositions"></param>

public static void FindMotifPositions(string medianValue, string SequenceValue, ref List<KeyValuePair<string,int>> MotifPositions)

{

//Variables used in this method.

bool setValue = true;

string v1 = string.Empty;

int v2 = 0;

int k = 0;

int score = 0;

int substring\_position = 0;

//Loop through DNA sequence and get substring of sequence size "6" and find out hamming score.

for (k = 0; k <= (SequenceValue.Length - medianValue.Length); k++)

{

score = 0;

//Get substring of size of l(L=6)

string tempValue = SequenceValue.Substring(k, medianValue.Length);

//Convert median and substring into character array for comparing each character at same position.

char[] median = medianValue.ToArray();

char[] tmpvalue = tempValue.ToArray();

//Loop through character array to find score of mismatch.

for (int l = 0; l < medianValue.Length; l++)

{

if (median[l] != tmpvalue[l])

{

score = score + 1;

}

}

int maxscore = score;

//Below written if statement will set minimun hamming score.

if (maxscore != substring\_position)

{

if (setValue)

{

substring\_position = maxscore;

setValue = false;

}

}

//We will check if the new hamming score value is less than the old hamming score value then set new value

//into variables so that that can be update into list.

if (maxscore < substring\_position)

{

substring\_position = maxscore;

v1 = tempValue;

v2 = k;

}

//If hamming score value is zero/same value then we use the below code set last string as value to below

//variables.

if(maxscore==substring\_position)

{

v1 = tempValue;

v2 = k;

}

}

//Add motif strings and their corresponding score into list.

MotifPositions.Add(new KeyValuePair<string, int>(v1, v2));

}

/// <summary>

/// This method is used to find out median string and its score. We will add new score to the existing score if that

/// median has already been placed into dictionary.

/// </summary>

/// <param name="medianValue"></param>

/// <param name="SequenceValue"></param>

/// <param name="searchValue"></param>

public static void TotalDistance(string medianValue, string SequenceValue, ref Dictionary<string, int> searchValue)

{

//Variables to be used in this method

bool setValue = true;

int k = 0;

int score = 0;

int substring\_position = 0;

//Loop through DNA sequence till the last substring string has size "6" and find out hamming score.

for (k = 0; k <= (SequenceValue.Length - medianValue.Length); k++)

{

score = 0;

//Get substring of size of l(L=6)

string tempValue = SequenceValue.Substring(k, medianValue.Length);

//Convert median and substring into character array for comparing each character at same position.

char[] median = medianValue.ToArray();

char[] tmpvalue = tempValue.ToArray();

//Loop through character array to find score of mismatch.

for (int l = 0; l < medianValue.Length; l++)

{

if (median[l] != tmpvalue[l])

{

score = score + 1;

}

}

int maxscore = score;

//We will check if the new hamming score value is less than the old hamming score value then set new value

//into variables so that that can be update into list.

if (maxscore != substring\_position)

{

if (setValue)

{

substring\_position = maxscore;

setValue = false;

}

}

//get the score for each substring.

if (maxscore <= substring\_position)

{

substring\_position = maxscore;

}

}

//We will check for duplicacy and if that string is not present then we will add to that list.

//if we find that then we will add that new score to existing score.

if (searchValue.ContainsKey(medianValue))

{

//Add new score to existing score for particular sustring from sequence.

searchValue[medianValue] = searchValue[medianValue] + substring\_position;

}

else

{

//Add substring from sequences and its score.

searchValue.Add(medianValue, substring\_position);

}

}

}

}

**OUTPUT:**

median string: GAAAAA (tot\_dist = 309)

motif consensus string: GAAAAA (consensus\_score = 3393)

motif positions/string s=(s1..st):

194(GAAAAA),2034(GAAAAA),221(GATTAA),876(GAAAAA),188(GAAAAA),124(GTAAAT),482(GAAAAA),384(GAAAAA),46(GAAAAA),12(GAAAAA),297(GAAATA),203(GAAAAA),170(TAAATA),876(GAAAAA),188(GAAAAA),170(GAATAA),12(GAAAAA),46(GAAAAA),500(GAAAAT),408(GAAAAA),609(GAAAAA),662(GAAAAA),819(GAAAAA),237(GAAGAA),361(CAAAAA),423(GAAAAA),162(GAAATA),718(CAAAAA),113(GAAACA),102(GAAATT),300(GAAAAA),954(GAAAAA),216(GAAAAA),322(GAAAAA),158(GATAAA),1168(GAAAAA),111(GAAAAA),200(GAAATA),249(GATAAA),371(GAAAAT),250(GAATAA),139(GAAAAA),984(GAAAAA),2079(GAAAAA),491(GAAAGA),479(GAATAA),42(GAAAAT),579(GAAAAA),194(CAAAAA),223(GTAAAA),108(GAAAAA),770(GAAATA),328(GAAAGA),401(GAAAAA),599(GAATAA),247(GAAAAA),726(GTAAAA),186(GACAAA),749(GAAAAA),317(GAAAAA),240(GAAAAC),818(GAAAAA),145(CAAAAA),797(GAAAAA),1191(GAAAAA),1032(GAAAAA),253(GAAAAG),23(GAAAAA),125(GAAAAA),115(GCAAAA),1014(GAAAAA),356(GAATAA),159(GAAAAA),3(GAAAAA),317(GAAAAA),122(GCAAGA),860(AAAAAA),779(GAAAAA),801(GAAAGA),2571(GAAAAA),108(GAAAAA),557(GAACAA),362(GAAATA),2(GAAAAA),487(GAAAAA),17(GAAAAA),89(AAAAAA),746(GAAGAA),2409(GATAAA),2(GAAAAA),2(GAAAAA),393(GAAACA),66(GAAAAA),33(GAACAA),129(GATAAA),2(GATAAA),138(GAAAAT),296(CAAAAA),336(GAAAAA),749(GAAAAA),1124(GAAAAA),678(GAAAAA),780(GAAAAA),335(GAAAAA),784(GAATAA),1310(GAGAAA),804(GACAAA),177(GAAAAA),2199(GAAAAC),42(GAAAAA),243(GAAAAA),525(GAAAAA),255(GAAAGA),251(GAAAAA),18(GAAAAA),407(GAAAAA),760(CAAAAA),116(GAAAAA),273(GAAAAA),181(GAAACA),25(GAAAAC),197(GAAACC),140(GAAAAC),732(GAAAAA),465(GAGAAG),47(GAAAAG),230(GCAAAA),425(GAAAAA),248(GAACCA),1509(GAAAAA),2(GAAAAC),75(GAAAAA),255(GAGAAA),123(GATTAA),36(GAACAA),326(GAAAAA),217(GAAAAA),703(GAAAAA),10(GAAAAA),388(GAAAAA),175(GCAAAA),363(GAAAAA),102(GAAAAA),66(GAAAAA),44(GAAAAA),186(AAAAAA),555(GAAAAA),288(GAAAAA),527(GAAAAA),254(AAAAAA),1013(GAAAAA),42(GAAAAA),161(GAAATG),598(GAAAAT),944(CAAAAA),7(GAAAAA),244(GAAAAA),401(GAAAAA),1652(GAAAAA),138(GAAATA),257(GAAGAA),567(GAAAAA),1002(GAAAAA),2684(GAAAAA),344(GAAAAA),214(GAAAAA),444(GAAAAA),59(GAAAAA),878(GAAAAA),4959(GAAAAA),3(GAAAAC),523(GAAAAA),78(GAAAAC),551(GAATAA),28(GAAAAA),225(GAAAAA),141(GAAAAG),366(GAAAAA),64(CAAAAA),231(GAAAAG),36(GAAAAT),166(GAACAA),168(GATTAA),146(CAAAAA),107(GAAAAA),1585(GAAAAA),174(GAAAAT),24(GAAAAA),387(AAAAAA),1416(GAAAAA),256(GCAAAA),354(GAAAAA),922(GAAAAA),49(GAAAAA),291(AAAAAA),223(GTAAAA),129(GAAAAA),108(GAAAAA),122(GACAAA),363(GAAAAA),8(GAAAAA),264(GAAAAC),282(GAATAA),257(GAAAAA),331(GAAAAA),39(GAAAAA),192(GAAAAA),161(CAAATA),603(GAAAAA),827(GAAGCA),119(CAAGAA),42(GCAAAA),200(GAACAC),8(GACAAA),192(GAAAAA),197(GAAAAA),95(GAAAAA),1851(GAAAAA),226(GAAAAA),8(CAAAAA),66(GAAAAA),54(GAAAAA),90(GCAAAA),36(GAAAGA),615(GAAAAG),1530(GAAAAA),182(GAAAAA),194(GAAAAC),587(GAAAAA),2(GAAAAA),201(GTAACA),566(GAAAAA),78(GAAAAA),762(GAAAAC),309(GAAAAA),627(GAAAAT),38(GAAAAA),365(GAAAAA),291(GAAAAA),647(GAAAAA),375(GAAAAC),207(GAAGAA),68(GAATAA),525(GAAAAA),45(GACAAA),251(GAAAAA),18(GAAAAA),407(GAAAAA),760(CAAAAA),535(GAAAAC),116(GAAAAA),273(GAAAAA),181(GAAACA),73(GAACAG),317(GAAAAA),2(GAACAA),1754(GAAAAA),2(GAAAAC),695(GAACAA),562(GAAAAC),261(GAAGAG),280(GAACAA),169(GAAACA),180(GACAAA),68(GAATAA),804(GAAAAA),131(GAAAAA),51(GAAAAA),466(AAAAAA),792(GAAAAA),245(GAAAAA),241(GAAAAA),1029(GAAAAA),1924(GAAAAA),786(GAAAAA),621(GAAAAA),245(GAAACA),225(GAAACA),22(GAAAAC),57(GAAAAA),201(GTAAAA),246(GAAAAA),1445(GAAAAA),32(GAAAAA),1149(GAAAAA),588(GAAAAA),1962(GAAAAA),732(GAAAAA),42(GAAAAC),263(GAAAAA),145(TAAAAA),521(GAAAAA),570(GAAAAA),699(GATAAA),499(GACAAA),300(GAAAAG),3723(GAAAAA),236(GAAAAA),105(GAAACA),1044(GAAAAA),707(GAAAAA),449(GAAAAA),562(GAAAAT),749(GAAAAA),607(GAAAAA),245(GAAAAA),624(GATAAT),459(GAAACA),537(GAAGAA),46(GAAAAA),676(GAAAAA),303(GATAAA),1049(GAAAAA),1613(GAAAAA),310(CAAAAA),2(GAAAAA),477(GAAAAA),261(GAAATA),74(GAAAAC),147(GAAAAA),195(GAAAAA),432(GAAAAT),276(GAAAAA),2(GAAACG),2312(GAAAAA),924(GAAAAC),498(GAAAAA),2(GAAAAA),660(GAAAAT),1139(GAACAA),927(GAAAAA),582(GAATAA),174(AAAAAA),96(GAAAAA),383(GAAAAT),1238(GAAAAA),1100(GAAGAA),133(GAAAAG),716(GAAAGA),537(GAAAAA),364(GAAATA),112(GAAAAA),1397(GAAAAA),203(CAAAAA),128(GAAAAG),228(GAAAAA),2796(GAAAAA),482(GAAAAA),174(GATAAA),665(GAAAAA),632(GAAAAT),557(GAAAGA),139(CAAAAA),23(AAAAAA),929(GAAAAA),3177(GAAAAA),201(GAAAAA),18(GAAAAA),561(GAAAAT),83(GAAAAA),921(GAAAAA),1145(GAAAAA),878(GAAAAA),540(GAAAAA),596(GAAAAA),702(GAAAAA),222(GAAAAA),93(GATAAA),833(CAAAAA),234(GATAAA),1803(GAAGAA),321(CAAAAA),338(GAAAAA),309(GAAAAT),747(GAAAAT),437(GAAAAA),1245(GAACAA),2(GAAAAA),13(GAAAAA),617(GAAAAC),275(GAAAAA),2733(GAAAAA),615(GAAAAA),459(GAAAAA),741(GAAAAA),1388(GAAAAA),843(GAAAAA),275(GAAAAA),535(AAAAAA),1575(GAAAAA),677(GAAAAA),1185(GAAAAA),789(GAAATA),153(GAAAAA),957(GAAAAA),474(GAAAAA),1515(GAAAAA),81(CATTAA),18(GTAAAT),891(GATAAA),120(GCTTAA),1705(GCAAAA),159(GAAAAA),22(GAATAA),687(GAAAAA),363(GAAACA),870(GAAGAA),357(CAAAAA),1493(GAAAAA),1095(GAAAAA),1224(GATAAA),144(CAAAAA),1269(GAACAA),102(GAAAAA),885(GAAAAA),1405(GAAAAA),537(GAAAAA),819(GAACAA),1191(GAAAAA),1097(GAAAGA),830(GAAACA),516(GCAAAA),1602(GAAAAA),204(GAAAAA),129(GAAAAA),36(GAAAAA),609(GAAAAA),330(GAAAAA),950(AAAAAA),1302(GAAAAC),261(GCAAAA),600(GAAAAA),482(GAAAAC),791(GAAAAC),1359(GTAAAA),93(GAAAAA),2040(GAAAAA),1472(GAAATA),1415(GAAGAA),1623(GAAAAA),2559(GAAAAA),282(GAAGAA),2(GAAAAA),6(GAAAAA),413(TAAAAA),181(GAAAAA),2223(GAAAAA),399(GATAAA),641(GAAACA),903(GAAAAA),41(GAAAGA),62(GAAAAA),1125(GTAAAA),285(GAAGAA),854(GACAAA),702(GAAAAA),774(GAAGAA),1052(TAAAAA),591(GAATAA),525(GAAAAA),498(GAAACA),2547(GAAAAA),675(GCAAAA),547(TAAAAA),459(GACAAA),1320(CAAAAA),489(GAAAAA),435(GAAAAA),678(AAAAAA),164(GAAAGA),2459(GAAAAA),1350(GAAAAA),119(GAAAAG),1673(GAAATA),375(GAAAAG),880(GTAAAG),1964(GAGAAA),108(GAAAAA),1359(GCAAAA),186(GAAAAA),516(GAAACA),555(GAAAAC),476(GAAAAT),1287(GAAAAA),1322(GAAAAA),800(GAAAAC),366(GAAAAA),176(GAAAAA),1032(GAAAAA),33(GAAAAA),699(GAAGAA),573(GAAAAA),696(GAAAAA),234(GAAAAA),1092(GAAAAA),27(GAAAAA),559(GTAAAA),1233(GAAAAG),8(GAAAAA),249(GAAAAA),940(GCAAAA),351(GAAAAA),720(GAAACA),447(GAAAAA),513(GAAAAA),3231(GAAAAA),375(GAAAAA),2125(TAAAAA),327(CAAAAA),1283(GAAAAC),99(GAAAAA),144(GAAAAA),500(GAATAA),417(GAATAA),570(GAAAAA),617(GAAAAC),89(CAAAAA),348(AAAAAA),927(GAAAAA),51(GAAAAA),290(GAAAAT),78(GAAAAA),566(GAATAC),1002(GTAAAA),558(CAAAAA),408(GAAAAA),490(GAAAAA),771(GATAAA),272(GAAAAA),1331(GAAAAA),620(GAAAAA),430(GAAAAA),254(GAAAAA),698(GAAGAA),552(GAAAAA),798(GAAAAA),369(GAAAAA),1344(GAAAAA),2469(GAAAAA),1404(GAACAA),258(GAAAAA),2379(GAAAAA),45(GAAAAA),1110(GAAAAA),506(GAAAAA),1254(GAAAAA),753(GAAAAA),1073(GAAAAA),348(GAAACA),1299(GAAAAC),11(CAAAAA),441(GAAAAT),77(GAAAAT),110(TAAAAA),445(GCAAAA),1554(GAAAAA),4137(GAAAAA),443(GAAAAA),358(GAAAAA),150(GAAAAA),151(TAAAAA),546(GAAAAA),723(GAAAAG),441(GAAAAC),1359(GAAAAA),558(GAAAAA),390(GAAAAA),732(GCAAAA),225(GATAAA),243(GATAAA),167(GAAAAA),302(AAAAAA),55(GAAAAA),1595(GAAAGA),176(GAAAAA),1029(GAAAGA),342(GCAAAA),1127(GAAAAA),50(GAAAAA),884(GAAAAA),6(GAAAAA),1227(GAAAAA),258(GTAAAA),521(GAAAAA),786(GAAAAC),84(GAAAAA),765(GAAAAA),802(GAAAAA),1707(GAAAAA),180(GAAGAA),303(GAAACA),546(GAATAA),224(GAATAA),831(GCAAAA),820(GAAAAA),596(TAAAAA),1508(GAAAAG),520(GAATAA),351(GAAAAA),387(GAATAA),402(GAAAAA),466(TAAAAA),311(GAAAAA),172(TAAAAA),521(GAGAAA),209(GAAAAG),500(GCAAAA),473(GAAAAA),617(GAAAAA),318(GAAAAA),378(GAAAAA),800(GAAAAC),75(GAAACA),444(GAAAAC),440(GAAAAA),312(GAAAAC),225(GCATAA)

median string: CTGGCA (tot\_dist = 345)

motif consensus string: CTGGCA (consensus\_score = 3357)

motif positions/string s=(s1..st):

606(CTGGCT),1677(CTGGCA),114(CTGGCA),937(CTGGCA),249(CTGGCA),106(CGGGCA),829(CTAGCA),574(CTGGAA),94(CTGGCA),99(CAGGCA),99(CTGGCA),200(CTGGAA),63(CGGGCA),937(CTGGCA),249(CTGGCA),132(CTGGCA),99(CAGGCA),105(CCGGCC),668(CTGGTA),385(CTGACC),158(CTGGCA),570(CTGGCA),701(CTGGCA),180(CTGGTA),357(CGGGCA),349(CTGGCA),159(CTGGAA),307(CTGGCA),155(CTGGCA),146(CCGTCA),669(CTGGCA),951(CTGGAA),186(CTGGCA),333(CCGGCA),331(CTGGAA),1158(CTGGCG),182(CTGGAA),72(CTGGAA),257(CTGTCA),657(CTGGAA),404(CTGGTA),222(CTGTCA),1446(CTGGGA),1544(CTGGCA),188(CTGTCA),504(CTGGCA),170(CTGGTA),738(CTGGCA),335(CTGGCC),257(CTTTGA),1935(CTGGCA),105(CTGGCA),246(CTGGCA),295(GTGGCA),758(CTGGCA),725(CTGGCG),734(CTGGCA),252(ATGGCA),888(CTGGCA),146(CTGGCT),249(CTGACA),222(CTGGCA),48(CTGGCA),642(CTGGCA),1446(CTGGCA),378(CTGGCA),275(CTGGTT),546(CTGGCG),27(CTGACA),21(CTGGCA),1191(CCGGCA),396(CTGGAA),25(ATGGCA),95(CCGGCA),146(CTGGCT),169(CTGGCC),855(ATGGCA),439(CTGGCA),492(TTGGCA),2096(CCGGCA),649(CTGTCA),312(CTGGTA),403(CTGGCA),505(CTGACA),244(CTGGTA),367(CTGACA),0(GTGGCG),701(CTGGCA),635(CTGGCA),63(ATGGCA),445(CTGACA),247(ATGGCA),270(CTGGCA),140(CTGGCA),97(CTGATA),162(CTGACA),128(CTGGCA),206(CTGGCG),105(CTGGCA),718(ATGGCA),1105(CTGGAA),399(CTGGCA),282(CTGGCA),816(CTGGCA),849(CTGGTA),497(CTGGCA),620(CTGGAA),824(ATGGCA),2078(ATGGCA),9(CTGGCT),324(CCGGCA),830(CTGGCA),168(CTGGCG),351(CTGGCA),192(CTGGAA),33(CTGGCA),756(CTGGCA),149(CTGGCA),288(CTGGCA),172(AGGGCA),33(CTGGCA),225(CTGGCA),220(CGGGGA),1245(GTGGCA),501(CTGCCA),145(CTGTCA),219(CTGGCA),333(CTGGGA),215(CTGGAA),1365(CTGGCA),348(CTGGCA),614(CTGGCA),312(CTGGCT),111(CGTGCA),18(CTGGCA),283(CTGACA),192(CTGGAA),659(CTGGCA),20(CTGGCA),294(CTGGCA),205(CAGGCA),207(CTGGAA),43(CTGCCA),176(CTGGGA),180(CTGGCA),227(CTGGCA),514(ATGGCA),673(CTGGCG),1030(CTGGCA),506(CTGGCA),341(CTGGCA),283(CTGACA),215(GTGGCA),138(CTGGCA),60(CTGGCA),239(CTGGCC),241(CAGGAA),465(CCGGCA),1683(CTGGCA),98(CTGGCA),147(CTGGCA),516(CTGGGA),1165(CTGGCA),1827(CTGGCA),671(CTGGGA),550(CTGACA),589(CTGGCT),18(CTGGCA),830(CTGGCA),1500(CTGGCA),503(CTGGCA),495(CTGGCA),149(CTGGCA),563(CTGGTA),77(ATGGCA),232(CTGCGA),138(CCGGAA),647(CTGGCA),0(ATGGCA),131(CGGGCA),113(CTGGGC),281(TTGGCA),102(CTGGCG),119(CTGGCG),669(CTGGTA),1605(CTGACA),163(GTGACA),67(CTGGCA),336(CTGGGA),1368(CTGGCA),129(CTGGCA),163(CTGCCA),60(CTGGCA),371(CTGTCA),240(CTGGAA),257(CTTTGA),270(CTGGCA),48(CTGACA),224(CTGGGA),1008(CTGGCT),256(CTGACA),251(GTGGCA),254(CGAGCA),189(CTGCCA),51(CGGGCA),132(CTGGCA),696(CTGGTA),92(CTGCCA),125(CTGGCA),1104(TTGGCA),83(CTGCCA),104(CTGGAA),228(CTGGCA),60(CTGCCA),696(CTGGTA),252(GTTGCA),1234(CCGGCA),1816(CTGTCA),223(CTGGAA),101(CTGGCA),270(CTGGCA),146(CTGGCA),87(CCGGCA),370(CTGGCC),470(CTGGCA),1572(CTTGCA),835(CTGGCA),83(CTGGCA),1011(CTGGCC),212(CTGGCT),56(CTGGCA),147(CTGGCA),333(CTGGCT),768(CGGGCA),189(CTGGCA),535(CTGGGA),148(CTGCCA),667(CTGACA),288(CTGGAA),1257(CTGGCC),324(CCGGCA),215(CTGGCA),97(CCGGCA),830(CTGGCA),168(CTGGCG),351(CTGGCA),192(CTGGAA),33(CTGGCA),756(CTGGCA),107(CTGGCA),149(CTGGCA),288(CTGGCA),172(AGGGCA),271(CTTGCA),237(CTGGGA),164(CTGGAA),1365(CTGGCA),222(CTGGCA),596(CTGGCA),489(CTGGCA),35(CTGGCA),238(CAGGCA),185(GTGGCG),408(CTGGCA),111(CTGGCA),813(CTGGCT),80(CTGGCC),213(CTGGAA),408(CTGGCA),399(CTGGCA),159(CTGGCA),207(CTGGGA),2516(CTGGCA),2273(CTGGGA),1015(CTGGCA),426(CTGGCA),259(GTGGCA),114(CTGGCA),108(CGGGAA),207(CTGGAA),191(CTGGAA),634(CAGGCA),1143(CTGGCA),464(CTGGCA),1061(CTGGCA),599(CTCGCA),1494(CTGGCA),464(CTGGCA),132(CTGGAA),294(CTGGCT),501(CTGGCA),590(CTGGCA),1209(CAGGCA),812(CTGGCA),246(CTGGCA),326(CTGGAA),3716(CAGGCA),509(CTGGCA),277(CTGGCC),819(CTGGCA),779(CTGGCG),261(CTGGCC),471(CTGGCC),914(CTGGCA),542(CTGGCA),1199(CTGCCA),539(ATGGCA),378(CTGGCA),577(CTGGTA),981(CGGGCA),864(CTGGCA),361(CTGGCA),888(CTGGCA),1398(CTGTCA),396(CCGGCA),141(CTGGCA),417(CTGGAA),160(CTGACA),242(CTGGCT),508(CTGGCG),627(CTGCCA),477(CTGGTA),195(CTGGAA),58(CGGGCT),1559(CTGGCA),618(CTGGCA),1197(CTGGCA),200(CTGGCA),197(CTGGCA),1230(CTGGCA),705(CTGGCA),264(CTGGCA),498(CTGGCG),185(CTGGCA),123(CTGGCG),1770(CTGGCA),880(CTGGCG),163(CTGGCC),1041(CTGGCC),35(CTGGCA),380(CGGGCA),593(CTGGCA),1206(CTGGAA),255(CTGGCT),120(ATGGCA),537(CTGGCA),2711(CTGGCA),392(CTGGCA),432(CCGGCA),627(CTGGCA),903(CTGGCG),607(CTGGCA),91(TTGGCA),105(TTTGCA),498(CTGGCA),2608(CTGGCA),210(CCGCCA),359(GTGGCA),515(CTGTCA),849(CTGGCG),632(CTGGCA),540(CTGGCA),1049(CTGGCG),1502(CTGGAA),486(CTGGAA),636(CTGGCA),1258(ATGGCA),231(CTGGAA),1313(CGGGCA),179(CTGGCA),1790(CTGGCA),395(CTGGGA),806(CCGGCA),226(CTGGCG),789(CTGGCG),483(CTGGCA),1121(CTGGCA),1359(CTGGCA),578(CTGGCA),903(CTGGCA),619(ATGGCA),2103(CTGGCA),1542(CTGGCA),686(CGGGCA),449(CTGGCA),1665(CCGGCA),308(CTGGCA),180(CTGGCA),626(CTGGCA),963(CTGGCA),545(CTGGCA),1451(CTGGCA),632(CTGGCA),222(CTGGCA),571(CTGGCA),912(CTGGCA),1496(CTGGCG),77(CCAGCA),34(ACGGCA),869(CTGGCA),111(CTGCCA),870(CTGGCA),373(CAGGCA),19(CTGGAA),912(CTGGCA),371(CTGGCA),786(CCGGCA),111(CTGCCA),1753(GTGGCA),957(CTGGCA),963(CTGGCA),1059(CTGGCA),834(CTGGCA),939(CTGGCA),641(CTGGCA),1347(CTGGCA),909(CTGGCG),593(CTGGCA),876(CTGGCA),763(CTGGCG),909(CTGGCA),314(CTGGCA),2334(CTGGCA),266(CTGGGA),173(CTGGAG),729(CTGGCC),501(GTGGCA),466(CTGGCA),624(CTGGCA),1136(CTGGCA),284(CGGGCA),775(CTGGCA),411(CGGGCA),62(CTGGCA),939(CTGGCA),117(CTGGCA),1737(CTGGCA),1005(CTGGCA),1278(CTGGCA),983(CTGGCA),330(CTGGCA),164(CTGGCA),300(CTGCCA),846(CTGGCT),429(CTGGCC),302(CTGGCA),2231(GTGGCA),520(TTGGCA),84(CTGGCA),873(CTGGAA),590(CTGGCA),380(TTGGCA),1098(CTGGCG),299(CTGGCG),429(CTGGCA),201(CTGGCA),558(CTGGCA),64(CTGGCA),550(CTGGCC),574(CTGCCA),540(CTGGCA),1974(CTGGCA),405(CTGGCA),39(CTGGCA),405(CTGGCG),1139(CTGGCA),812(CTGGCA),448(CTGGCT),636(CTGGCT),135(CTGGCA),1578(CTGGCA),2082(CTGGCA),57(CTGGCG),1119(CTGGCA),733(ATGGCA),626(CTGGCA),789(CTGGCA),1176(CTGGCA),1356(CTGGCA),267(CTGGAA),575(TTGGCA),194(CTGGCA),237(CTGGCA),32(CTGGCA),1371(CTGGCA),1011(CTGGCA),363(CTGGAA),724(ATGGCA),2345(CTGGCA),779(CTCGCA),684(CTGGCT),304(CTGGCA),264(CTGGCA),522(CTGGCA),1185(CTCGCA),656(CTGGCC),566(CTGGCA),1320(CTTGCA),2426(CTGGTA),39(CTGGCA),935(CTGGCG),319(CTGGCG),714(CTGGCG),1008(CTGGCT),579(CTGGGA),3328(CTGGGA),240(CTGGCA),1358(CTGGCA),234(CTGGCT),1277(CTGGCA),169(CTGGCG),136(CTGTCA),527(CTGGCA),354(CTGGCA),585(CTGTCA),903(CTGGCA),265(CTGGGA),563(CTGGCG),702(CTGGCA),693(CTGGAA),360(CAGGCT),591(CTGGCA),48(CTGGCA),983(CTGGCT),525(CTGGCA),783(CTGGCT),898(CTGTCA),760(CCGGCA),365(CTGGCA),1176(CTGGCA),128(CTGGCA),285(CTGGCA),308(CTGGCA),723(CGGGCA),609(GTGGCA),738(CTGACA),213(CTGGCA),1148(CTGGCA),2886(CTGGCA),1098(CTGGCA),639(CTGGCA),2193(CTGGCA),675(CTGGCG),983(CTGGCA),240(CTGGCA),1176(CTGGCC),680(CTGGCA),1181(CTGGCA),68(CTGGCA),1382(CTGGTA),203(CTGGCA),438(CTGGAA),105(CCCGCA),25(GAGGCA),472(CTGGCA),821(CTGGCA),2126(CTGGCA),91(CTGACA),196(ATGGCA),105(CTGGCA),113(CTGGCT),830(CTGGCA),515(CTGGCA),438(CTGGAA),2085(CTGGCA),457(CTGGCA),707(CTGGCA),705(CTGGCA),250(CTGGAA),199(CTGGCG),692(CTGGCA),284(CAGACA),94(CTGGAA),1359(CTGGCA),246(CTGGCA),819(CTGGCA),143(CTGGAA),924(CTGGCA),459(CTGGCA),1193(CTGGCA),135(CTGGCG),843(CTGGCA),351(CTGGAA),729(CTGGCA),910(CTGGCA),195(CTGGCA),57(CTGGCA),1220(CTGGCA),1332(CTGGCA),161(CTGGCA),555(CTGGCA),539(CTGGCG),218(CGGGCT),738(CTGGCA),9(CTGGCA),519(CTGGCA),591(CTGGCA),411(CTGGAA),378(CTGGCA),360(CTGGCG),672(CTGGCA),259(CTGGCA),744(CTGGCA),187(CTGGCA),344(CTGGCC),203(CTGGAA),225(CTGGCA),878(CTGGCA),711(CTGGTA),463(CTTGCA),69(CTGGCA),404(CTGGCA),557(CTTGCA),537(CTGGCA),397(ATGGCA),375(CTGCCA),205(ATGGCA)

median string: CTGGCG (tot\_dist = 348)

motif consensus string: CTGGCG (consensus\_score = 3354)

motif positions/string s=(s1..st):

475(CTGGCG),525(CTGGCG),114(CTGGCA),1437(CTGGAG),507(CTGACG),68(CTGCCG),922(TTTGCG),397(CTGGAG),216(CTGACG),66(CTGGAG),182(CTTGCG),564(CTTGCG),25(CTGCCG),1437(CTGGAG),507(CTGACG),66(CTGGCG),156(CTCGCG),124(CTGTGG),555(CTGGCT),173(CTGGTG),69(CTGGCG),740(CTGCCG),701(CTGGCA),246(CTGGGG),188(CTGGCG),585(CTGACG),159(CTGGAA),676(CTGGAG),155(CTGGCA),36(CTGGGG),692(CGGGCG),663(CTGGCG),312(CTGGGG),281(CTGGCG),230(CTGGCC),1158(CTGGCG),149(CTGGCC),159(CTCGCG),181(CTGGTG),187(GTGGCG),337(CAGGCG),126(CTGCCG),420(CTGGCG),1140(CTGGCG),523(GTGGCG),513(CTGGCG),159(CTGGTG),1122(CTGGCG),335(CTGGCC),202(TTGCCG),696(CTGGCG),66(CTGGCG),273(CTGCCG),21(CTGGCG),758(CTGGCA),725(CTGGCG),472(CTGGCG),252(ATGGCA),651(CTGGCG),225(CTGGAG),161(CTGGCG),600(CTGGCG),48(CTGGCA),1385(CTGGCG),1610(CTGGCG),745(CTGGCG),275(CTGGTT),546(CTGGCG),21(CTGGCG),201(CTGGCC),161(CTGGCG),390(CTGGTG),180(ATGGCG),73(CTGTCG),225(CTGGAG),169(CTGGCC),447(CTGGGG),439(CTGGCA),705(ATGGCG),1434(CTGGCG),225(CCGGCG),655(GTGGCG),554(GTGGCG),432(CTGGGG),685(GTGGTG),433(CTGACG),0(GTGGCG),701(CTGGCA),745(CTGGCG),445(CTGACG),433(CAGGCG),357(CAGGCG),413(CTGTCG),236(CTGGCC),139(ATGTCT),116(CTGCCG),212(CTGGTG),206(CTGGCG),258(CTGGCG),375(CTGGCT),987(CTGGCG),741(CTGGCT),468(CTGGCG),859(CTGGCC),866(GTGGCG),1285(CTGGGG),784(ATAGCG),591(CTGGCG),1813(CTGGCG),9(CTGGCT),202(CTGGCG),951(GTGGCG),168(CTGGCG),127(CTGGCG),132(CTGGTG),276(CTGGCG),153(CTGGCG),111(CTGGCG),393(CAGGCG),138(CAGGCG),72(CTGGCG),248(CTGTCG),191(CCGGCG),551(CTGGCG),132(CTGGCG),85(CTGACG),116(CTGGCG),119(CTGGCG),206(CTGGCG),1157(CTGGCG),354(CTGGCG),102(CTGGCG),79(CTGGCG),75(CAGGCG),135(CTGTCG),211(CTGGCT),171(CTGGGG),659(CTGGCA),136(CTGTCG),334(CTGTCG),205(CAGGCA),166(ATGGCG),182(ATGGAG),176(CTGGGA),180(CTGGCA),278(CTGGCG),546(CTGGTG),673(CTGGCG),1230(CAGGCG),506(CTGGCA),2546(CTGGCG),399(ATGGCG),33(CAGGCG),30(CTGGCG),968(CTGGCG),294(CTTGCG),282(GTGGGG),543(GTGGCG),1869(CTGGCT),744(GTGGCG),189(CTGGCG),429(CTGGTG),372(CTGGCG),2433(CTGGCG),650(CTGGGG),462(ATGGCG),514(CTGGCG),180(CTGGCG),924(CTGGCG),4755(CTGGCG),276(CTGGCG),156(CTGGCG),194(CTGCCG),482(ATGGCG),51(CCGGCG),212(CTGGAT),56(TTGGCG),1011(CTGGCT),230(ATGGCG),126(CTGGCC),48(CTGACG),281(TTGGCA),102(CTGGCG),119(CTGGCG),567(CTGGCG),1112(ATGGCG),187(ATGGAG),241(CTGACG),317(CTGGCG),233(CTGGCG),93(CTGGCG),360(CGTGCG),917(ATGGCG),579(CTGGGG),221(CTGGCG),202(TTGCCG),270(CTGGCA),36(GTGGCG),171(CTGGCG),453(CTGGCG),171(CTGACG),193(CTGGCG),110(ATGGGG),152(GTGGCG),344(CGGGCG),312(CTGGCG),212(CTGGCG),108(CTGACG),221(CTGGCG),879(CTGGCG),60(ATGGCG),165(CCGGTG),255(CTGGTG),76(CTGACG),212(CTGGCG),154(CTGCCG),1150(CTGACG),1688(CTGGCG),104(CTGGTG),263(GTGGCG),378(CTGGCG),242(CTGGCC),67(CTGCCG),478(CTGACG),724(CTGTCG),1032(CTGGCG),835(CTGGCA),255(CTGGGG),327(CTGGCG),317(CTGCCG),171(CGGGCG),701(CTGGCG),337(CTGCCG),749(CTGGCG),231(CTGGCG),126(CTGGGG),186(CTGTCG),572(CTGGTG),351(CTGTCG),1001(CTGGCG),202(CTGGCG),230(CTGGCT),137(CTGTCG),961(ATGGCG),168(CTGGCG),387(CTGCCG),132(CTGGTG),276(CTGGCG),153(CTGGCG),301(CTGGCG),111(CTGGCG),393(CAGGCG),138(CAGGCG),223(CTGGCG),325(GTGGCG),155(CTGGCG),1157(CTGGCG),348(CTGGCG),614(CTGGCG),548(CTGGCT),411(GTGGCG),182(CTGGCG),185(GTGGCG),408(CTGGCA),141(CTGGCG),60(CTGGCG),80(CTGGCC),23(CTGGCG),408(CTGGCA),135(CTGGCG),159(CTGGCA),305(ATTGCG),2516(CTGGCA),1681(CTGGCG),1057(CTGGCC),976(CTGGCG),41(CAGGCG),198(CTGGAG),96(CTGGAG),207(CTGGAA),141(CTTGCG),580(CTGTCG),1921(TTGGCG),539(CTGGCG),1122(CTGGCT),504(CTGGCC),1461(CTGGCG),639(CTGGCG),8(CTGGAG),294(CTGGCT),377(CTGGCG),202(CTGGCG),1191(CAGGCG),975(CTGGAG),478(CTGGCG),306(CTGCCG),1571(CTGGCG),539(CTGGAG),17(CTGGCG),819(CTGGCA),779(CTGGCG),383(CTGGAG),504(CCGGCG),1096(CTGGCG),542(CTGGCA),1009(CTGGCG),445(CTGGAG),400(GTGGCG),577(CTGGTA),910(ATGGCG),1415(CTGGCT),412(CTGGCG),1207(CCGGCG),566(CTGGCG),418(CTGTCG),133(CTGGCG),570(CAGCCG),267(ATGGTG),200(CTGGCG),508(CTGGCG),317(CTGGCG),277(TTGGCG),134(CTGTCG),58(CGGGCT),2085(CTGGCG),522(CTGGCG),771(CTGGCG),149(CTGGCG),507(CTGGCG),957(CTGGCG),882(CTGGCG),375(CAGGCG),498(CTGGCG),453(CTGCCG),123(CTGGCG),1831(CTGCCG),880(CTGGCG),163(CTGGCC),945(CTGGCG),837(CCGGCG),380(CGGGCA),8(CTGGCG),852(CTGGAG),121(CTGGCG),88(CAGGCG),516(CTGGCG),1217(CTGGCG),392(CTGGCA),328(CTGGCG),690(CTGGCG),903(CTGGCG),690(CTGGAG),91(TTGGCA),72(CCGCCG),855(CTGGCG),2877(CTGGCG),192(TTAGCG),322(CTGCCG),509(CGGGCG),849(CTGGCG),1012(CTGGCG),1100(CTGGCG),1049(CTGGCG),1389(CTGGCG),453(CTGACG),768(CTGGCG),1143(CTGGCG),219(CTGGGG),1284(CTGGCG),474(CAGGCG),1083(CTGGCG),465(CTGGAG),210(CTGGCG),226(CTGGCG),789(CTGGCG),906(CTGGCG),477(CTGGCG),1838(CTGGCG),565(CTGGCG),849(CTGGCG),504(CTGGCG),2553(CTGGCG),1198(CTGGCG),39(CTGGCG),1336(GTGGCG),303(CTGGCG),674(CTGGCG),584(CTGGCG),626(CTGGCA),1383(CTGGCG),771(CTGGCG),1494(CTGGCG),1063(CTGGCG),137(CTGGCG),1230(CTGGCG),660(CTGGCG),1496(CTGGCG),50(TTTGCG),13(CTGTGG),884(CGGGCG),107(CTGGCT),1518(CTGGCG),448(CTGGTG),91(ATGGCT),507(CTGGCG),371(CTGGCA),546(CTGGCG),239(ATGGCG),1098(CTGGCG),1284(CTGGCG),252(CTGGCG),810(CTGGCG),1296(CTGGCG),639(CTGGCG),498(CTGGCG),1234(CTGGCG),909(CTGGCG),72(CTGGCG),981(CTGGCG),763(CTGGCG),825(CTGGCG),95(CTGGCG),2607(CTGGCG),251(CTGGCT),173(CTGGAG),241(CTGGCG),585(CTTGCG),732(CTGGCG),1071(CTGGCG),609(CTGGCG),427(ATGACG),817(CGGGCG),366(CTGGCG),729(CTGGCG),1134(CTGGCG),418(CTGGCG),2655(CTGGCG),1135(CTGGCG),1359(CTGGCG),534(CTGGCG),2427(CTGGCG),244(CTGGCG),770(CGGGCG),550(CTGGCG),429(CTGGCC),1303(CTGGCG),1422(CTGGCG),510(CTGCCG),353(CTGGCT),669(CTGGCG),711(GTGGCG),104(CTGGCG),1098(CTGGCG),299(CTGGCG),729(CTGGCG),843(CTGGCG),582(CTGGCG),1228(CTGGTG),271(CTGGCG),392(CTGGAG),540(CTGGCA),2100(CTGGCG),187(CTGGCG),436(CTGGCG),405(CTGGCG),1059(CTGGCG),860(CTGGCG),108(CTGGCG),139(CTGGCG),303(CTGGCG),2223(CTGGCG),1101(CTGGCG),57(CTGGCG),2094(CTGGCG),558(CTGGCG),678(CTGGCG),1602(CTGGCG),742(CTGGCG),1380(CTGGCG),204(CTGGTG),561(CTGGCG),251(CTGGCG),471(CTGGCG),1356(CTGGCC),903(CTGGCG),978(CTGGCG),318(CTGGAG),778(CTCGCG),2481(CTGGCG),402(CTGGCG),693(CAGGCG),46(CTGGCG),603(CTGGCG),522(CTGGCA),957(CTGGCG),656(CTGGCC),405(CTGGCG),373(CTGGCG),805(CTGGCG),457(CTGCCG),935(CTGGCG),319(CTGGCG),714(CTGGCG),951(CTGGCG),411(CTGGCG),2411(CTGGCG),870(CTGGCG),1764(CTGGCG),363(CTGCCG),849(CTGGCG),169(CTGGCG),174(CTGGTG),414(CTGGCG),354(CTGGCA),628(CTGGTG),849(CTGGCG),156(CTGGCG),563(CTGGCG),1174(CTGGCG),656(CTGGCG),208(GTGGCG),591(CTGGCA),616(CAGGCG),705(CTGGCG),234(CTGGCG),621(CTGGCG),745(CTCGCG),375(CTGGCG),415(CTGGCG),1039(CTGGCG),466(CTGACG),678(CAGGCG),405(CTGGCT),719(CTGGCG),600(CTGGTG),491(CTGGCG),213(CTGGCA),255(CTGGCG),2849(CTGGCG),677(CTGGCG),639(CTGGCA),2595(CTGGCG),675(CTGGCG),165(CTGGCG),474(CTGGTG),957(CTGGCG),960(CTGGCG),1740(CTGTCG),219(CTGGCC),849(CTGGCG),203(CTGGCA),279(GTGGCG),69(ATGGCG),22(ATGGAG),494(CTGGCG),1263(CTGGCG),3507(CTGGCG),318(CTGGAG),417(CTGGGG),105(CTGGCA),113(CTGGCT),1026(CTGGCG),398(CTGGCG),368(CTGGCT),2424(CTGGCG),457(CTGGCA),483(CTGGCG),705(CTGGCA),261(GCGGCG),199(CTGGCG),306(CTGGCG),234(CTGACG),238(AAGGCG),1638(CTGGCG),738(CTGGTG),1038(CTGGTG),31(CTGGCG),873(CTGGCG),486(CTGGAG),1156(CTGGCG),135(CTGGCG),960(CTGGCG),357(CCGGCG),772(CTGGCG),631(CTGGCG),870(CTGGCG),724(CTGGCG),1220(CTGGCA),1573(CTGGCG),210(CTGACG),425(CTGGCG),539(CTGGCG),232(TTCGCG),1221(CTGGCT),461(CTGGCG),739(CCGGCG),718(CTGGCG),165(CTGGAG),574(CAGGCG),360(CTGGCG),1120(CTGGCG),425(CTGGCC),744(CTGGCA),301(CTGCCG),594(CTGACG),126(CTGACG),555(CTGGGG),878(CTGGCA),574(CAGGCG),541(ATGGTG),170(CGGGCG),555(ATGGCG),70(CTGGCG),477(CTGGCG),423(CTGCCG),325(CTGTCG),179(CTGGCT)

median string: CTGGAA (tot\_dist = 353)

motif consensus string: CTGGAA (consensus\_score = 3349)

motif positions/string s=(s1..st):

610(CTGTAA),690(CTGGAA),231(CTGGAA),1506(CTGGAA),351(CTGGAA),121(ATGGTA),198(CTGGAA),574(CTGGAA),94(CTGGCA),84(CTGGAC),133(CAGGAA),200(CTGGAA),185(CTGTTA),1506(CTGGAA),351(CTGGAA),42(CTGGAA),84(CTGGAC),43(ATGGAA),347(CTGGAA),270(CTTGAA),665(CTGGGA),678(CCGGAA),402(CTGGAA),23(CTGGAA),318(ATGGAA),648(CTTGAA),159(CTGGAA),690(CTGAAA),155(CTGGCA),100(CTGAAA),206(CTGGAA),951(CTGGAA),474(CTGGGA),454(CCGGAA),331(CTGGAA),1320(GTGGAA),182(CTGGAA),72(CTGGAA),285(CTGGAT),657(CTGGAA),404(CTGGTA),222(CTGTCA),35(CTGGAA),2058(CTGGAA),479(CTGGAT),595(CTGTAA),170(CTGGTA),738(CTGGCA),240(CCGGAA),234(ATCGAA),2093(CTGGAA),729(CCGGAA),432(CTGGAA),781(CTGTAA),758(CTGGCA),99(CTGGAA),734(CTGGCA),252(ATGGCA),746(CTGGAA),314(GTGGAA),387(CAGGAA),986(ATGGAA),137(CTGGAT),1019(CTGGAA),654(CTGGAA),1053(CTGGAT),184(CCGGAA),303(CTGGAA),93(CAGGAA),33(CCGGAA),1038(CTGGAA),396(CTGGAA),96(CGGGAA),0(GTGGAA),314(GTGGAA),199(CTGTGA),508(CTGAAA),602(CTGCAA),799(CTGAAA),1904(CTGGAA),444(CTGAAA),609(CCGGAA),517(CGGGAA),235(CAGGAA),662(ATGGAA),285(CTGAAA),108(CTCAAA),701(CTGGCA),1523(CTGGAA),114(CAGGAA),349(CAGGAA),418(CGGGAA),507(CTGGAA),140(CTGGCA),97(CTGATA),162(CTGACA),144(CAGGAA),322(GGGGAA),789(CTGAAA),663(CTGAAA),1105(CTGGAA),942(CTGGAA),840(CTGGAC),332(CTGGAA),465(CTGGAA),1300(CGGGAA),620(CTGGAA),1088(CGGGAA),1117(CTGGAG),132(GTGGAA),372(CCGGAA),879(CTGGAA),21(CGGGAA),404(CTGGAA),192(CTGGAA),405(CTGAAA),767(CCGGAA),417(CTGGAC),270(CTGGAA),179(CAGAAA),111(CCGGAA),225(CTGGCA),214(CTGAAA),1226(CTGTAA),162(CTGGAA),70(CTGGAA),219(CTGGCA),399(CCGGAA),215(CTGGAA),1683(CTGGAA),366(CTGGAT),531(CTGGAA),56(CTGGAA),90(CAGGAG),51(CTGGAA),252(CGGGAA),192(CTGGAA),809(GTGGAA),20(CTGGCA),375(CAGGAA),155(CTGTAA),207(CTGGAA),239(CTTGAT),111(CTGGAA),180(CTGGCA),267(CCGGAA),495(CTGGAG),428(CTGGGA),1281(CGGGAA),506(CTGGCA),2090(CTGGAA),231(CTGGAT),42(CTGGAA),530(CTGGAT),914(CTGGGA),275(CTGGAT),241(CAGGAA),534(CTGGAG),1127(CTGGAA),495(CTGGAT),254(CTGGAA),60(CTGGAA),1236(CTGGAA),2873(GTGGAA),77(CTGGAA),102(CTGGAA),671(GTGGAA),378(CCGGAA),245(CTGGAA),3456(CTGGAA),621(CTGGTA),312(CTGGAA),75(CTGGAA),563(CTGGTA),25(CGGGAA),212(CTGGAT),138(CCGGAA),999(CTTGAA),248(CAGGAA),174(CGGGAA),16(CTGAAA),163(ATGGAA),47(CAGGAA),23(CTGGTA),387(CTGGAA),1677(ATGGAA),53(CGGGAA),393(CTGGGA),343(CCGGAA),1143(CTGGAA),290(CGGGAA),293(CGGGAA),807(CTGGAA),667(CTGTAA),240(CTGGAA),234(ATCGAA),270(CTGGCA),155(CCGGAA),224(CTGGGA),504(CTGGAA),177(CTGGAA),0(ATGGAA),206(ATGGAA),63(CTGGAT),404(CAGGAA),252(CTGGAT),696(CTGGTA),69(CTGGAA),600(CTTGAA),1094(ATGGAA),54(CTGGAG),104(CTGGAA),228(CTGGCA),109(CGGGAA),696(CTGGTA),233(CTGGAC),1062(CTGGAA),660(CTGGAA),223(CTGGAA),269(CAGGAA),507(CTGGAA),146(CTGGCA),97(CGGAAA),208(CTGAAA),353(CTGGAA),876(CTGGAA),1080(CCGGAA),267(CTGGAT),867(CTGGAA),134(CTGGAG),56(CTGGCA),533(CTGGAA),222(CTGCAA),30(CTGGAA),699(CTGGAT),535(CTGGGA),279(CTTGAA),850(CTGAAA),288(CTGGAA),711(CTGGAA),372(CCGGAA),204(CTGGAA),253(ATGGAG),879(CTGGAA),299(CCGGAC),404(CTGGAA),192(CTGGAA),405(CTGAAA),767(CCGGAA),408(CTGCAA),417(CTGGAC),270(CTGGAA),179(CAGAAA),271(CTTGCA),237(CTGGGA),164(CTGGAA),1683(CTGGAA),257(CTGGAT),531(CTGGAA),387(CTGGAA),456(CTGGAC),270(CCGGAA),166(CGGGAA),386(CTGGAA),240(CTTGAA),147(CTGGAA),129(CTGAAA),213(CTGGAA),408(CTGGCA),686(CGGGAA),302(CAGGAA),321(ATGGAA),1116(CTGGAA),2004(CTGGAA),1035(CTGAAA),599(CTGGAA),201(CTGGAT),198(CTGGAG),108(CGGGAA),207(CTGGAA),191(CTGGAA),462(CTGGAA),2167(CTGGAA),144(CTGGAA),1077(CTGAAA),464(CAGGAA),258(CTGGAA),606(CGGGAA),132(CTGGAA),346(CTGAAA),511(TTGGAA),68(CTGGAA),761(CTGGAA),905(CTGGAA),645(ATGGAA),326(CTGGAA),2952(CTGGAA),521(CTGGAA),272(CTGGAC),1071(CTGGAA),158(CTGGAA),446(CCGGAA),108(CTGGAA),618(CTGGAA),626(CTGTAA),990(CTGGAA),621(CTGGAT),410(CTGGAA),31(CTGGAA),611(CTGGAA),1419(CTGGAA),361(CTGGCA),803(CTGGAA),998(CTGGAA),376(CTGCAA),486(CTGGAA),417(CTGGAA),282(CAGGAA),171(CAGGAA),465(CCGGAA),504(CTGGAA),471(CTGGAA),195(CTGGAA),46(CAGGTA),1551(CTGGAA),921(CTGGAA),285(CTGGAA),200(CTGGCA),384(CTGGAA),987(CTGGAA),924(CTGGAA),504(CTGGAA),461(CTGGAT),441(CTGGAA),381(CTGAAA),1713(CTGGAA),420(CTGGAA),123(CTGGTA),228(CTGGAA),558(CTGGAA),361(ATGGAA),1048(CTGGAA),1206(CTGGAA),40(CTGAAA),195(CTCGGA),765(CTGGAA),2505(CTGGAA),456(CTGGAA),420(CTGGAA),876(CTGGAA),164(CTGGAA),690(CTGGAG),123(CTAAAA),94(CGGTAA),51(CTGGAA),1959(CTGGAA),199(ATGAAA),195(CTGGAA),470(CTGGAA),147(CTGGAA),1320(CTGGAA),981(CTGGAA),902(GTGGAA),1502(CTGGAA),486(CTGGAA),155(CTGGAA),762(CTGGAA),231(CTGGAA),1005(CTGGTA),501(CTGGAA),1587(CTGGAA),465(CTGGAG),690(CTGGAA),307(ATGAAA),519(CTGGAA),900(CTTGAA),918(CTGGAA),1521(CTGGAA),606(CTGGAA),591(CTGGAA),114(CTGGAA),2760(CTGGAA),1899(CTGGAA),567(CTGGAA),1469(CTGGAA),1056(CTGGAA),489(ATGGAA),12(CTGGAA),626(CTGGCA),1269(CTGGAA),332(CTGGAA),1296(CTGGAA),1000(CTGGAA),441(CTGGAA),782(CTGGAA),348(CTGGAA),720(CTGGAA),81(CATTAA),40(ATGGAT),909(ATGGAA),111(CTGCCA),1593(CTGGAA),432(GTGGAA),19(CTGGAA),696(CTGGAA),371(CTGGCA),729(CTGGAA),312(CTGCAA),326(CTGGAA),1092(CTGGAA),564(CTGGAA),1059(CTGGCA),489(CTGGAA),749(CTGGAA),882(CTGGAA),1218(CTGGAA),903(CTGGAA),748(CTGGAG),945(CAGGAA),744(CTGGAA),909(CTGGCA),267(CTGGAA),2517(CTGGAA),210(CTGGAA),173(CTGGAG),717(CTGGAT),606(CAGGAA),90(CTGGAA),1095(CTGGAA),1260(CTGGAA),171(CTGGAA),765(CTGGAA),157(CTGGAA),585(CTGGAT),648(CTGGAA),318(CTGGAA),2340(CTGGAA),1308(CTGGAA),699(CTGGAA),1740(CAGGAA),1971(CTGGAA),234(CTGGAA),743(GTGGAA),834(CTGCAA),111(CTGGTA),1403(CTGGAA),2048(GTGGAA),537(CTGGAC),639(CTGAAA),873(CTGGAA),590(CTGGCA),57(CTGGAG),654(CTGGAA),137(CTGGAA),645(CTGCAA),753(CTGGAA),537(CTGGAA),1207(ATGGAA),537(CTGGAT),392(CTGGAG),540(CTGGCA),1343(CTGGAA),348(CTGGAA),459(CTGGAT),282(CTGGAT),1194(CTGGAG),656(CTGGAA),375(CTGGAA),318(CTGAAA),135(CTGGCA),2357(CTGGAA),2354(CTGGAC),117(ATGAAA),474(CTGGAA),222(CTGGAA),861(CTGGAT),1579(CCGGAA),1005(CTGGAA),345(CTGGAA),267(CTGGAA),513(CTGGAA),210(CTGGAA),516(ATGGAA),279(CTGGAA),1131(CTGGAA),345(CTGGAA),363(CTGGAA),501(CTGGAG),2262(CTGGAA),659(CTGGAC),429(CTGGAA),411(CTGGAA),576(CTGGAA),522(CTGGCA),1089(CTGGAA),218(CTGGAA),837(CTGGTA),675(CTGGAA),408(CTGGAA),246(CTGGAA),987(CTGAAA),252(CTGGAA),729(CCGGAA),1131(CTGGAG),510(CTGGAA),3312(CTGGAA),42(CTGGAA),1908(CTGGAA),369(TTGGAA),987(CTGGAA),219(CCGGAA),0(GTGGAA),507(CTGGAA),414(CGGGAA),402(CTGGAA),591(CTGGAA),265(CTGGGA),241(TTGGAA),1566(ATGGAA),693(CTGGAA),287(CTCGAA),543(CTGGAA),576(CTGGTA),990(CAGGAA),525(CTGGCA),681(CTGGAA),789(GTGGAA),597(CGGGAA),600(CTGGAA),1277(CTGGAA),618(CTGAAA),611(CCGGAA),435(CTGGAA),564(ATGGAA),585(CTGGTA),474(CTGGAA),378(CTGGAA),1245(CTGGAA),3078(CTGGAA),45(CTGGAA),639(CTGGCA),2784(CTGGAA),441(CTGGAA),1238(CTGGAC),444(GTGGAA),873(CTGGAA),362(CTGGAA),135(CTGGAA),363(CTTGAA),381(CTGGAA),203(CTGGCA),438(CTGGAA),118(CTTCAA),131(CCGTAA),507(CTGGAA),2054(CTGGAA),2552(CTGGAA),461(CTGGAT),395(CTCGAA),217(ATGGAA),184(GAGGAA),887(CTGCAA),84(CTGGAA),438(CTGGAA),2208(CTGGAA),457(CTGGCA),348(CTGGAA),705(CTGGCA),250(CTGGAA),250(TAGGAA),692(CTGGCA),162(CTGGAT),94(CTGGAA),1452(CTGGAA),549(CTGGAA),573(CTGGAA),143(CTGGAA),1036(CTGAAA),479(CTGGAA),1293(CTGGAA),398(CTGGAT),1085(CTGGAA),351(CTGGAA),984(CTGGAT),1047(CTGGGA),1194(CTGGAA),1125(CTAGAA),278(CTGGAA),1917(CTGGTA),177(CCGGAA),702(CGGGAA),455(CTGGAT),15(CAGGAA),1174(CTGAAA),840(CTGGAT),519(CTGGCA),1505(ATGGAA),411(CTGGAA),398(CTGGAA),384(GTGGAA),503(CTGGAA),463(CTGTAA),438(CTGGAA),345(CTGGAT),170(CTGGAA),203(CTGGAA),354(CGGGAA),878(CTGGCA),711(CTGGTA),564(ATCGAA),539(CTGGAA),860(CTGGAT),168(CTGGAA),125(CTGGAA),234(CCGGAA),414(CTGCAA),152(CTGGAA)

median string: TGCTGG (tot\_dist = 353)

motif consensus string: TGCTGG (consensus\_score = 3349)

motif positions/string s=(s1..st):

514(TGCTTG),1777(TGCTGG),112(TGCTGG),962(TGCTGG),274(TGCTGG),104(TGCGGG),196(TGCTGG),466(TTCTGG),214(TGCTGA),82(AGCTGG),97(TGCTGG),482(TACTGG),61(TGCGGG),962(TGCTGG),274(TGCTGG),250(TGATGG),64(TGCTGG),20(TGCTCG),155(TGCTGA),174(TGGTGG),857(TGCAGG),658(TGCTGA),829(TGCTGG),244(CGCTGG),301(TGCTCG),526(TGCTGG),157(GTCTGG),674(TCCTGG),153(TACTGG),140(TGCCAG),133(TGCTGG),940(CGCTGG),354(TCCTGG),448(TGCCGG),329(CGCTGG),709(TGCTGG),180(CCCTGG),119(TGCTGT),25(TGCTGC),655(TGCTGG),402(TACTGG),127(TGCCGG),868(TGCTGG),1837(TGCTGG),388(GGCTGG),601(TGCTGG),168(CGCTGG),929(TGCTGG),284(TGCCGG),123(TGCTGT),730(TGCTGG),556(TGCTGG),475(TGCAGG),452(TGCTGG),756(CGCTGG),739(TGCCGG),70(TGCTGG),40(AGCTGG),814(TGCTGG),176(TGCAGG),219(TGCTGT),462(TGCTGG),46(TGCTGG),1383(TACTGG),1608(TTCTGG),872(TGTTGG),273(TACTGG),604(TGATGG),121(TGGTGA),192(TGCGGG),632(TGCTGG),394(TGCTGG),197(TGTTGT),67(AGCTGG),176(TGCAGG),192(TGGTGC),866(TGATGG),767(TGCAGG),406(TGCTGG),2065(TGCTGG),376(TGCCGG),653(TGGTGG),530(TGCTGA),332(TGCTGG),683(TGGTGG),311(TGCAGG),55(TGTTGT),699(TCCTGG),1556(TGCTGG),272(TGATGG),431(TGCAGG),404(TGCAGG),268(TGCTGG),138(TTCTGG),95(TTCTGA),124(CGCTGG),126(CGCTGG),318(TTCTGG),214(TGCTGG),448(TGATGG),883(TGCTGG),940(TTCTGG),334(TGCTGG),463(TGCTGG),937(AGCTGG),1201(TGCAGG),491(TGATGG),1136(TGCTGC),934(TGCTGG),139(TGCAAG),223(TGCTGG),322(TGCTGG),166(GGCTGG),499(TGCTGG),190(TTCTGG),274(TGCTGG),510(TGCTGG),326(TGCAGG),268(TGCTGG),148(TCCTGG),127(TGATGG),93(TGCCGG),246(CGCTGA),901(TGCTGG),427(TGCTGG),214(TGCTGC),83(TGCTGG),331(TGCTGG),240(TACTGG),1363(TGCTGG),409(TGATGG),612(TTCTGG),210(CGCTGG),113(TGCAGC),16(TGCTGG),209(CGCTGG),181(TGTTGG),752(TGATGG),139(TCGTGG),71(TGCTGG),220(TGCCTG),103(TGCTGG),237(TGCTTG),174(TACTGG),265(TCCTGG),265(TGCCGG),499(AGCTGG),457(TGCTGG),1279(TGCGGG),504(CGCTGG),1993(TGCTGG),125(TGCGGG),40(TGCTGG),589(TGCCGG),353(TGCTGG),258(TGCTGG),34(TGCTGG),532(CGCTGG),1119(TGCTGG),691(CGCTGG),235(GGCTGG),514(GGCTGG),619(TGCTGG),1066(TGCTGG),669(CGCTGG),478(TGGTGG),635(TGCCGG),196(TGCTGG),1423(TGCTGG),4390(TGCTGG),619(TGCTGG),544(TGCTTG),195(TGCCGG),427(TGCCGG),88(CGCTGG),242(TGATGC),116(TGCGGG),655(TGCTGG),224(TGCTGC),22(TGCTGG),111(GGCTGG),161(TGATGG),100(AACTGG),162(TGCTGG),667(TTCTGG),992(TGCTGG),188(TGGAGG),239(TGCTGA),1(TGCTGG),1499(TGCTGG),232(TCCTGG),325(TGCTGA),637(TGCTGG),455(TGATGG),238(AGCTGG),123(TGCTGT),268(CGCTGG),101(TGCCGG),208(TGCCGG),652(TGCTGG),175(CGCTGG),191(TGCTGG),220(TGCAGC),242(TTCGGG),268(TGATGG),365(TTCTGG),118(TGCTGG),124(TGATGG),365(TGCTGG),877(TGCTGG),52(TGCTGG),131(TGCTGA),184(TGCTGG),92(TGATGG),118(TGCTGG),267(TGGTGA),506(TGCTGG),1866(TTCTGG),242(TGCTGT),261(TGGTGG),268(TGCTGG),144(TTCTGG),85(TTCCGG),368(TGCTGG),304(TGCTGG),1574(TGCAGG),833(TACTGG),240(TGCTGG),1009(TGCTGG),304(TGGTGG),54(AGCTGG),712(TGCTGG),319(AGCTGG),123(TGCTGG),187(TGCTGG),503(TGCTCG),287(TGATGG),842(TGGTGG),286(TACTGG),1039(TGCTGG),223(TGCTGG),202(TGCTGG),251(TGATGG),322(TGCTGG),166(GGCTGG),499(TGCTGG),190(TTCTGG),274(TGCTGG),510(TGCTGG),299(GGCTGG),326(TGCAGG),268(TGCTGG),148(TCCTGG),190(TGCTGG),235(TGCTGG),189(TACTGG),1363(TGCTGG),262(TGCTGG),700(AGCTGG),445(TGCCGG),388(TGCTGG),59(TGCAGG),121(TGCTGA),406(CGCTGG),139(TCCTGG),580(TGCTGG),78(GGCTGG),178(TGCAGG),399(TGGTGG),622(TGCTGG),188(TGCAGG),274(TGCTTG),624(TGCTGG),2002(TGCTGG),1006(TGCAGG),982(TGGTGG),254(TGCTTG),112(TGCTGG),94(TTCTGG),129(TGGTGG),189(GTCTGG),632(TGCAGG),1646(TGCTGG),313(TGCTGG),1009(TGCAGG),377(TGCCGG),2164(TGCTGG),117(TGCCGG),130(TTCTGG),296(GGCTGG),499(TGCTGG),200(TGCTGG),1004(TGCCGG),832(TTCTGG),476(TGCTGG),307(TGCCGG),157(TGCTGG),325(TGCTGG),237(TGGTGG),1069(TTCTGG),625(TGCTGT),95(TGCTGC),469(TTCTGG),1094(CGCTGG),586(TGCGGG),1007(TGCTGG),619(TGCTGG),434(TGCCGG),518(TGATGG),100(TGCTGG),1417(GGCTGG),428(TGATGG),1027(TGCGGG),1595(TGGTGG),410(TGCTGG),484(TGCTGG),425(TTATGG),235(TACTGG),235(TGCAGG),506(AGCTGG),502(TTCTGG),469(TCCTGG),226(TGCTGA),56(TGCGGG),1871(TGCTGG),520(TGCTGG),769(TGCTGG),198(CGCTGG),435(TTCTGG),568(TGCTGG),362(TGCTGG),502(TGCTGG),409(TGCTGG),197(TGCTGG),51(TGCTGC),1171(TGCTGG),1091(TGATGG),179(TGCTGT),921(TGCTGG),400(TGCTGG),257(TGATGG),970(TACTGG),1324(TGCACG),134(TGCTGC),47(TGCGGG),685(TGCTGG),2709(TGCTGG),454(TGCTGG),418(TGCTGG),874(CGCTGG),895(TGCTGG),415(TGCTGG),89(TTTTGG),80(TGCCAG),847(TGCTGG),2938(TGCTGG),137(TAATGG),193(AGCTGG),577(TGCAGG),847(CGCTGG),1192(TGCTGG),895(GGCTGG),1003(TGCTGG),1387(TGCTGG),394(TGCTGG),887(TGTTGG),760(TGCTGG),112(TGCTGG),523(TGCTGG),499(TGCTGG),1120(TGCTGG),463(TTCTGG),741(TGCTGG),283(TGCAGG),808(TGCAGG),781(TGCTGG),1138(TGCTGG),1495(TGCTGG),625(TGCTGG),634(TGCTGG),502(TGCTGG),2530(TGCTGG),1249(TGCTGG),640(TGCTGG),1213(TGCTGG),301(TGCTGG),448(TGCTGG),582(TGCTGG),601(TGGTGG),1153(TGCTGG),235(TGCTGG),724(TGCTGG),487(TGCTGG),220(TGCTGG),550(TGCTGG),76(TGCTGG),1378(CGCTGG),52(TGCGCG),11(TGCTGT),559(TGCTGG),119(TGCTTA),769(TGCTGG),259(TGCTGG),49(TTCTGC),898(TGCTGG),325(TGCTGG),685(TGCTGG),211(TGCTGG),814(TGCTGG),1468(TGCTGG),1142(TGCTGG),1042(TGCTGG),1189(TGCTGG),937(TGCTGG),553(TGCTGG),1345(TGCTGG),802(TGCTGG),538(TGCTGG),701(TGCTGG),778(TGCTGG),907(TACTGG),563(TGCTGG),2440(TGCTGG),264(AGCTGG),122(TGCTGA),511(TGCTGG),379(TGCTGG),730(TGCTGG),676(TGCTGG),652(TGCTGG),354(GGCTGG),700(TGCTGG),526(TGCTGG),811(TGGTGG),161(TGCTGG),694(TGCTCG),2185(TGCTGG),458(TGCTGG),1261(TGCTGG),25(TGCTGG),2167(TGCTGG),319(TGCAGG),604(TGGTGG),767(TTCTGG),451(TGCTTG),1504(TGCTGG),2131(TGCTGG),535(TGCTGG),49(TGCTGG),667(TGCTGG),283(TGCTGG),298(TGATGG),1201(GGCTGG),374(TGCTTG),568(TGCTGG),751(TGCTGG),535(TGCTGG),62(TGCTGG),535(CGCTGG),554(TGCGGG),506(TGCAGG),2423(TGTTGG),707(TGGTGG),536(TGCTGG),416(TGATGG),1192(TGCTGG),921(TGCTGA),446(GGCTGG),688(TGCCGG),510(CGCTGG),2296(TGCTGG),1828(TGCTGG),55(CGCTGG),1490(TGCTGG),556(TGCTGG),562(TGCTGG),1858(TGCTGG),1156(TGCTGG),1096(TGCTGG),205(TGGTGG),274(TGCTGG),415(TGCTGG),658(TGATGG),277(TGCTGG),1369(TGCTGG),343(TGCTGG),176(TGATGG),499(TGCTGG),2479(TGCTGG),70(TGCTGG),650(TGCTGC),302(TGCTGG),574(TGCTGG),106(TGCTGG),361(TGCTGG),84(CGCTGG),403(TGCTGG),1300(TGCTGG),2306(TGCGGG),365(TGCTGT),746(TGCCGG),85(TGCTGG),230(TGCTGG),949(TGCTGG),262(TGCTGG),3121(TGCTGG),868(TGCTGG),1389(TGCTGG),1(TGCTGG),676(TGCTGG),235(TGGTGG),172(TACTGG),130(TGCTGG),352(GGCTGG),626(TACTGG),634(TGCTGG),293(TGTTGG),323(CGCTGG),1597(TGCTGG),691(CGCTGG),362(GGCTGT),341(TGCTGG),583(TGCTGG),184(TGCTGG),457(TGCTGG),340(TGCTGG),757(TGGTGG),430(TGGTGG),232(TGCTGG),700(TGCTGG),392(TGGTGG),508(TGCTGC),413(TGCTGC),691(TGGTGG),601(TGGTGG),489(GGCTGG),211(TGCTGG),322(TGCTGG),2641(TGCTGG),913(TGCTGG),514(TGCTGG),2782(TGCTGG),673(TGCTGG),1033(TGCTGG),472(TGCTGG),1051(TGCTGG),1012(TGCTGG),133(TGCTGG),361(TGCTTG),994(TGCTGG),55(TGCAGG),436(TGCTGG),70(TGGCGG),85(TGCTGA),115(TGCTGG),1215(TGCTGG),3337(TGCTGG),459(GACTGG),415(TACTGG),103(TTCTGG),171(AGCCGG),406(TGCTGG),280(TGCTGG),398(TGATGG),2206(TGCTGG),343(TGCTGG),610(TGCTGG),61(TGCTGG),248(AACTGG),197(AGCTGG),690(TTCTGG),160(CGCTGG),229(TGATGG),1636(TGCTGG),613(TGCTGG),1042(TGCTGG),172(TGCGGG),1060(TGCAGG),517(TGCTGA),1132(TGCTGG),358(CGCTGG),1045(TGCTGG),349(AGCTGG),847(CGCTGG),220(TGCTGG),775(TGCTGG),744(TGCTGG),1255(TGTTGG),860(TGCTGG),159(TCCTGG),553(TGCTGG),453(TGCTGG),124(TGCTGA),694(TGCTGG),406(TGCTGG),517(CGCTGG),1235(TGCTGG),506(TGCCGG),649(TGCAGG),16(TGCTGG),1153(TGCTGG),360(TTCTGG),742(CGCTGG),343(TTCTGG),124(TGCTGG),201(CGCTGG),193(TGCTGG),847(AGCTGG),228(TGCTGG),65(TGCTGT),221(TGCTGA),820(TGCGGG),424(TGCTGG),567(TGCTGA),424(TGCCGG),292(CGCTGG),177(TACTGG)