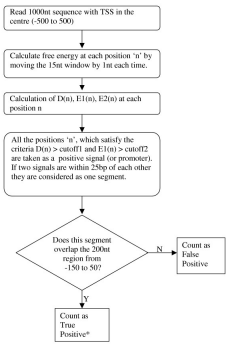
Predicción de promotores mediante el algoritmo Nearest Neighbor ΔG calculator

Ciro Ramírez Suástegui, José Damián Martínez Reyes

05/03/2016

El modelo Nearest Neighbor (NN) aproxima la estabilidad molecular de un ácido nucleico en disolución a partir de su secuencia, que se analiza como una secuencia de dinucleótidos solapantes de interacciones aditivas, cuyas energías se han determinado experimentalmente (SantaLucia, 1998; Breslauer et al., 1986).[1]

El siguiente diagrama muestra el flujo del programa, solo que tendrá algunas modificaciones:



Algoritmo de Kanhere & Bansal (2005), que se basa en calcular la diferencia de ΔG entre dos ventanas en torno a una posición n.

El código

El archivo del programa estará aquí.

Agregamos las librerías y las opciones del programa:

```
use strict; # Para restringir construcciones no seguras
use warnings; # Advertencias del código
use Getopt::Long; # LTomamos los parámetros y datos en el prompt
my %opts = ();
# Agragamos las llaves y sus valores para los parámetros del prompt
GetOptions (\%opts, 'h'); # Tomamos las intrucciones
if(($opts{'h'})){
   &PrintHelp();
}
```

Para llamar al programa `program_name [-h] file`

Ahora definimos las variables y parámetros globales:

```
# Varaibles globales
my $T           = 37; # temperatura(C)
my $windowL     = 15;  # tamaño de la ventana
```

```
# http://www.biomedcentral.com/1471-2105/6/1
my %NNparams    = (
  # SantaLucia J (1998) PNAS 95(4): 1460-1465.
    # [NaCl] 1M, 37C & pH=7
    # H(enthalpy): kcal/mol , S(entropía): cal/kmol
    # definimos los valores para cada dinucleótido
    'AA/TT' , {'H',-7.9, 'S',-22.2}, 'AT/TA' , {'H',-7.2, 'S',-20.4},
    'TA/AT' , {'H',-7.2, 'S',-21.3}, 'CA/GT' , {'H',-8.5, 'S',-22.7},
    'GT/CA' , {'H',-8.4, 'S',-22.4}, 'CT/GA' , {'H',-7.8, 'S',-21.0},
    'GA/CT' , {'H',-8.2, 'S',-22.2}, 'CG/GC' , {'H',-10.6,'S',-27.2},
    'GC/CG' , {'H',-9.8, 'S',-24.4}, 'GG/CC' , {'H',-8.0, 'S',-19.9},
    # Costo de inicio de la secuencia
    'G'     , {'H', 0.1, 'S',-2.8 }, 'A'     , {'H', 2.3, 'S',4.1  },
    # Corrección de simetría
    'sym'   , {'H',   0, 'S',-1.4 } );
my %seqs; # Secuencias
my $g_deltaG; # Para el dG
my $n; # Contador
```

Leemos el archivo de la secuencia:

```
# verificamos que exista el archivo y tomamos los cutoff
my $infile = $ARGV[0] || die "# usage: $0 <promoters file>\n";
#my $cut1 = $ARGV[1] || die "# usage: $1 <cut-off for Dn>\n";
#my $cut2 = $ARGV[2] || die "# usage: $2 <cut-off for E1>\n";
# Visualizamos las condiciones
print "# parameters: Temperature=$T gC Window=$windowL\n\n";
# Abrimos nuestro archivo y comenzamos
open(SEQ, $infile) || die "# cannot open input $infile : $!\n";
print("Reading sequences...\n");
print("Calculating dGs...\n");
while(<SEQ>)
{
  if(/^(b\d{4}) \\ ([ATGC]+)/)
    {
        # Tomamos el identificador y la secuencia (459 nts)
        my ($name, $seq) = ($1, $2);
        #printf("sequence %s (%d nts)\n",$name, length($seq));

        # Construyendo la colección de secuencias
        $seqs{$name}{'seq'} = $seq;
        # Posteriormente %seqs tendrá otro hash con las ventanas y sus dGs
    }
}
close(SEQ);
```

Subrutinas:

Secuencia complementaria

- Obtiene la secuencia complementaria
- Parámetro: Secuencia de ADN
- Devuelve: Secuencia complementaria

```
sub complement{ $_[0] =~ tr/ATGC/TACG/; return $_[0] }
```

ΔG del duplex de ADN

- Calcula la energía libre de un dúplex de ADN
- Parámetros: 1) Secuencia de ADN; 2) Temperatura Celsius
- Revuelve: Escalar de energía libre
- Usa el hash global %NNparams

```
sub duplex_deltaG
{
     my ($seq, $tCelsius) = @_;
    my $tK = 273.15 + $tCelsius;
    my ($DNAstep, $nt, $dG, $total_dG) = ('','',0,0);
    my @sequence = split(//, uc($seq));

    # Agregar dG de dinucleótidos sobrelapados
    for($n=0; $n<$#sequence; $n++)
    {
        # Construimos el identificador concatenando los dinucleótidos
        $DNAstep = $sequence[$n].$sequence[$n+1].'/'.
            complement($sequence[$n].$sequence[$n+1]);

        if(!defined($NNparams{$DNAstep})) # Pregunta si está el dinucleótido
        {
            $DNAstep = reverse($DNAstep); # Si no, obtiene la reversa
        }

        $dG = ((1000*$NNparams{$DNAstep}{'H'})-
                ($tK*$NNparams{$DNAstep}{'S'}))
                / 1000 ; # Calculating the dG value

        $total_dG += $dG;
    }

    # Agregar corrección para inicio de secuencia
    $nt = $sequence[0]; # Primer par; pregunta si está
    # Si no, usa el complementario
    if(!defined($NNparams{$nt})){ $nt = complement($nt) }
    $total_dG += ((1000*$NNparams{$nt}{'H'})-
                    ($tK*$NNparams{$nt}{'S'}))
                    / 1000;

    $nt = $sequence[$#sequence]; # Último par; pregunta si está
    # Si no, usa el complementario
    if(!defined($NNparams{$nt})){ $nt = complement($nt) }
    $total_dG += ((1000*$NNparams{$nt}{'H'})-
                    ($tK*$NNparams{$nt}{'S'}))
                    / 1000;

    # Corrección de simetría
    $seq = join('', @sequence);
    my $fseq = substr($seq, 0, $windowL/2); # Definimos una cadena para
```

```
    # mejor manejo y sacamos la reversa complementaria
    my $rseq = complement(reverse(substr($seq, $#sequence-$windowL/2, $windowL)));
    if ($fseq eq $rseq)
    {
        $total_dG += ((1000*$NNparams{'sym'}{'H'})-
            ($tK*$NNparams{'sym'}{'S'}))
            / 1000;
    }

    return $total_dG;
}
```

Predictor de TSS

- Calcular E1, e2 y D para cada uno de los n
- Predecir los promotores respecto a valores de corte de e1 y D
- Parámetros: 1) %hash de secuencias 2)
- Devuelve: hash de predicciones

```
sub predict
{
  my $seqs = shift;
    my $cut1 = shift;
    my $cut2 = shift;
    # Definimos las variables que vamos a usar
    my $e1;
    my $e2;
    my $Dn;
    my %preds;
    my $ntposition;

    # Volvemos a recorrer todas nuestras secuencias
    foreach my $name (keys(%seqs))
    {
        # Recorremos toda la secuencia
        for($ntposition = 0; length($seqs{$name}{'seq'}) >= $ntposition+199+$windowL; $ntposition++)
        {
            # Para E1; recorremos la longitud, sumamos y dividimos
            $e1 = 0;
            for($n = $ntposition; $n!=$ntposition+50; $n++)
            {
                $e1 += $seqs{$name}{$n};
            }
            $e1 /= 50;
            $e2 = 0;
            # Para E2; recorremos la longitud, sumamos y dividimos
            for($n = $ntposition+99; $n!=$ntposition+199; $n++)
            {
                $e2 += $seqs{$name}{$n};
                #print "Segundo ", $ntposition, "\t", $e2,  "\tLimite: ", length($seqs{$name}{'seq'})-2
            }
            $e2 /= 100;
            # Obtenemos la diferencias entre E2 y E1: D(n)
```

```
            $Dn = 0;
            $Dn = $e2 - $e1;
            # Guardamos los valores para cada posición
            $preds{$name}{$ntposition}{'Dn'} = $Dn;
            $preds{$name}{$ntposition}{'e1'} = $e1;
            $preds{$name}{$ntposition}{'e2'} = $e2;
            # Verificamos si pasan el umbral
            # Y reportamos si es positivo o negativo
            if($Dn > $cut1 and $e1 > $cut2)
            {
                $preds{$name}{$ntposition}{'result'} = "positivo";
            }else
            {
                $preds{$name}{$ntposition}{'result'} = "negativo";
            }
        }
    }
    return(%preds);
}
```

Programa principal

Obtener las ventanas y sus dG

```
foreach my $name (keys(%seqs))
{
  # Recorremos toda la secuencia
    for(my $ntposition = 0; $ntposition <= length($seqs{$name}{'seq'})-$windowL; $ntposition++)
    {
        # Tomamos por ventanas hasta llegar al número del largo de la secuencia
        # menos el tamaño de la ventana
        my $subseq = substr($seqs{$name}{'seq'}, $ntposition, $windowL);
        # Llamamos a la subrutina que calcula todos los dG de cada ventana
        $seqs{$name}{$ntposition} = duplex_deltaG($subseq, $T);
    }
}
```

Definir los parámetros de corte y llamar al predictor

Nos fijamos en la siguiente tabla.

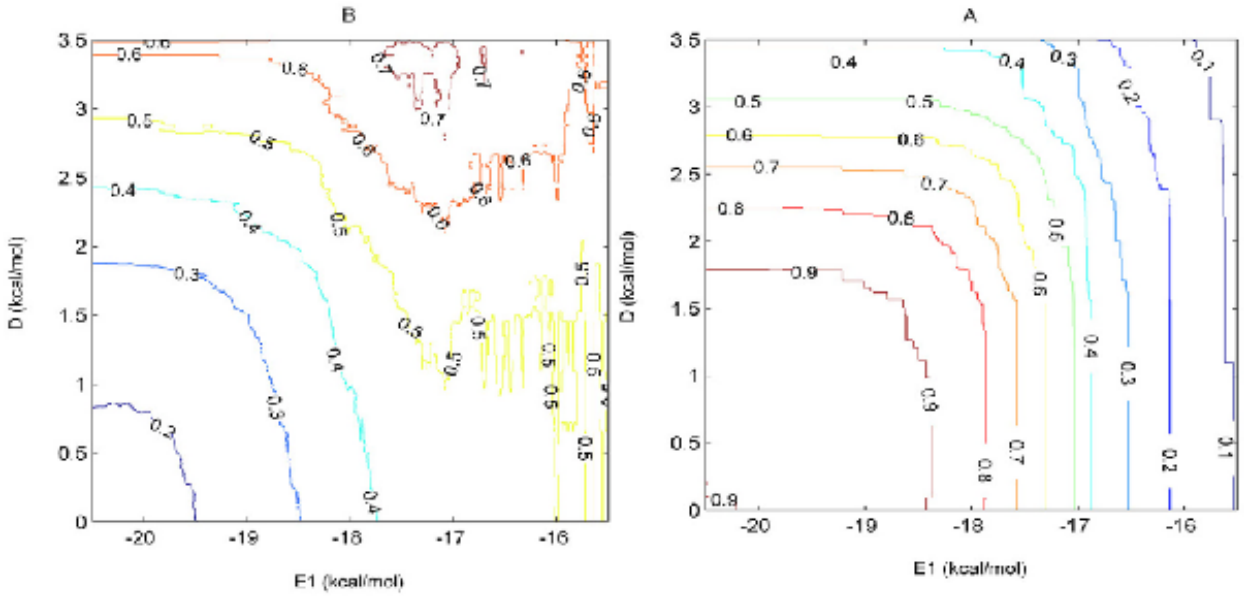
Sensitivity	Cut-off for D	Cut-off for E1 (kcal/mole)	Frequency of false positives	
			FP (1/nt)[a]	FP (1/nt)[b]
0.13	3.4	-15.99	1/16214	1/261000
0.22	3.4	-16.7	1/11350	1/130500
0.32	3.3	-17.1	1/8407	1/65250
0.40	3.3	-17.55	1/6486	1/29000
0.50	2.76	-17.53	1/3914	1/13737
0.60	2.45	-17.64	1/2467	1/7250
0.70	2.35	-18.07	1/1621	1/2747
0.81	1.9	-18.15	1/1086	1/1878
0.90	0.97	-18.37	1/572	1/967

[a] The false positives in the 1000 nt fragments, with TSS at the centre (-500 to +500).
[b] The false positives in the 1000 nt fragments extracted from the centre of ORFs with length more than 2000 nt.

Decidimos tomar como parámetros de corte a:

D(n)	E1	sensibilidad
2.35	-18.07	0.7

Esto porque quisimos situarnos en el centro de las gráficas de sensibilidad y precisión.




```
print("Cut-off for\nDn = ");
my $cut1 = <STDIN>;
print("Cut-off for\nE1 = ");
my $cut2 = <STDIN>;

my %predictions = predict(\%seqs, $cut1, $cut2);
```

Creación del reporte de sitios predichos

```
my $report = "predicted_TSS.txt";
open(my $fh, '>', $report) or die "Could not open file '$report' $!";
foreach my $name (keys(%predictions))
{
  # Recorremos las posicipones de las secuencias
    for(my $ntposition = 0; length($seqs{$name}{'seq'}) >= $ntposition+199+$windowL; $ntposition++)
    {
        # Solo guardamos las que tenemos como positivas
        if($predictions{$name}{$ntposition}{'result'} eq "positivo")
        {
            print $fh ($name, "\t", $ntposition-150, "\t");
            print $fh ($predictions{$name}{$ntposition}{'Dn'}, "\t");
            print $fh ($predictions{$name}{$ntposition}{'e1'}, "\t");
            print $fh ($predictions{$name}{$ntposition}{'e2'}, "\n");
        }
    }
}
close $fh;
print "Done\n";
```

Ayuda del programa

```
#########################################################################
```

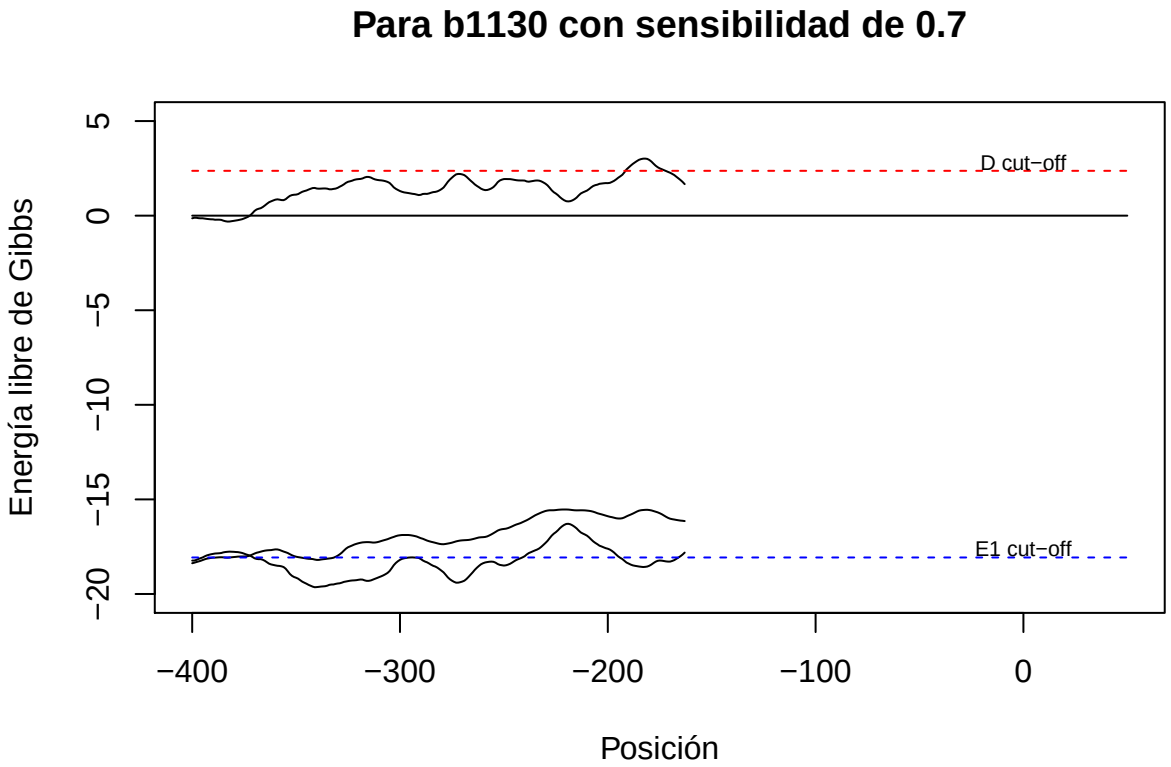
```
#### Despliega la ayuda en linea con opcion -h ###################
######################################################################
sub PrintHelp {
   system "pod2text -c $0 "; # Convert POD data to formatted ASCII text
   exit();
}
```

Resultados

Resultamos con un archivo de nombre *predicted_TSS.txt*. Sin embargo pudimos obtener varios valores de corte rápidamente, dada las características del programa.

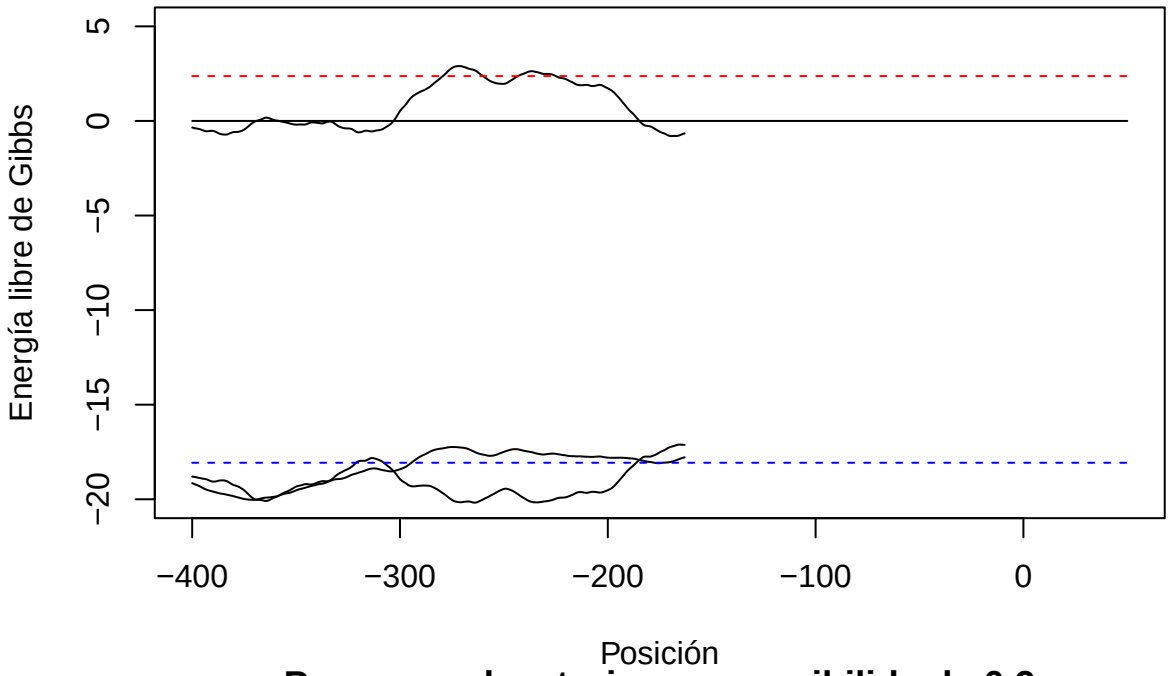
```
##      id  pos           D        E1        E2   result
## 1 b0708 -400 -0.03512410 -18.51230 -18.54743 negativo
## 2 b0708 -399 -0.00491720 -18.58128 -18.58620 negativo
## 3 b0708 -398  0.01229710 -18.63200 -18.61970 negativo
## 4 b0708 -397  0.00438165 -18.66316 -18.65878 negativo
## 5 b0708 -396 -0.00851315 -18.69157 -18.70008 negativo
## 6 b0708 -395  0.00072180 -18.74628 -18.74555 negativo
```

Podemos graficar $D(n)$, E1 y E2 para la secuencia "b1130". Procesada con una sensibilidad de 0.7 se puede observar un valor de $D(n)$ sobre el umbral, lo que nos indica que podría estar allí el TSS. En este caso tiene sentido debido a que la posición en la que cae es un potencial lugar.

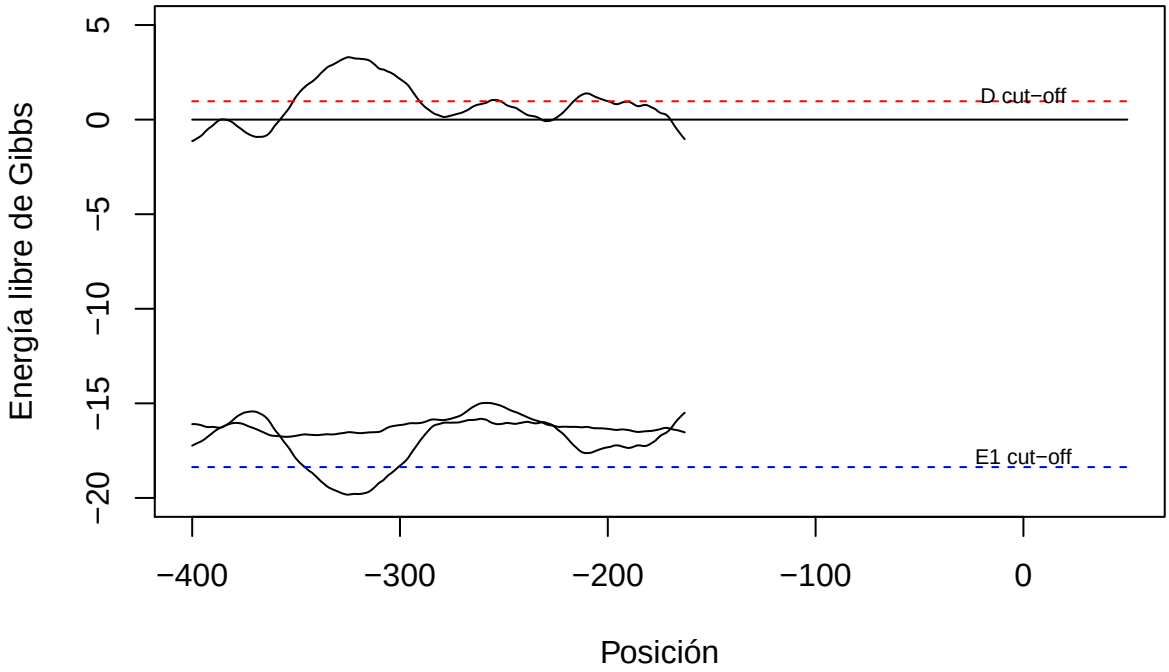


En esta sección podemos **graficamos $D(n)$, E1 y E2 para una posición n al azar** para ver más predicciones de TSS de las demás secuencias.

Para un n alaeatorio con sensibilidad de 0.7

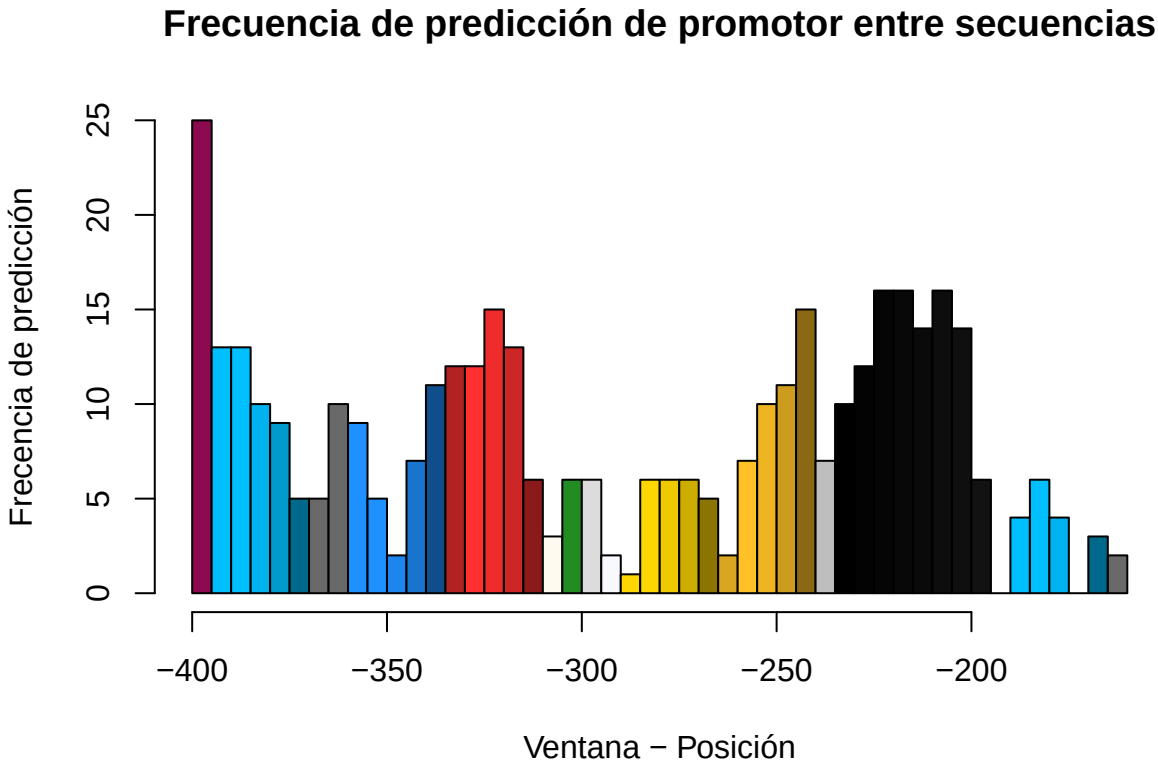


Para un n alaeatorio con sensibilida de 0.9

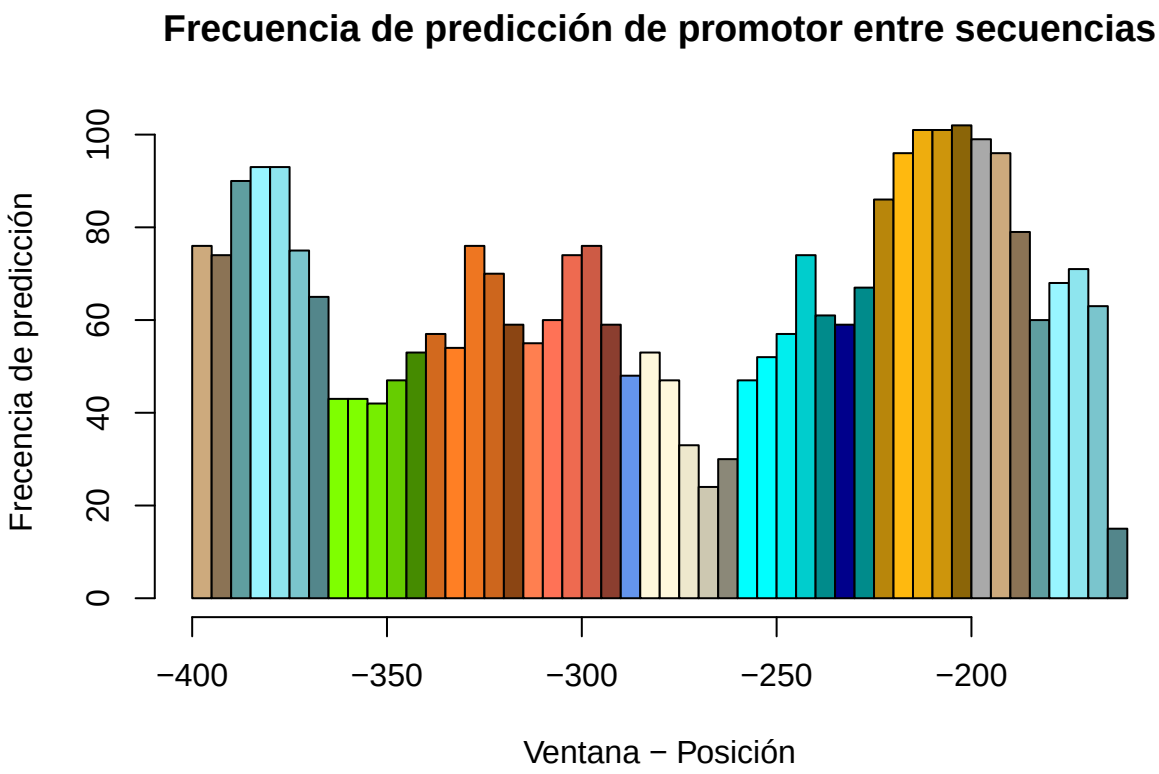


Frecuencias

Para una sensibilidad de 0.7



Para una sensibilidad de 0.9



Observamos que al conseguir más sensibilidad la cantidad de predicciones aumenta debido a que se presenta aún más falsos positivos y esto se refleja en un histograma de distribución más uniforme. Sin embargo no logra apreciarse bien una zona de predicción, por lo que podríamos inferir que los datos son muy variables.

Conclusión

Podemos concluir que, al menos en particular con este conjunto de secuencias, no se puede indentificar una zona donde caigan frecuentemente los TSS. Podríamos decir que con base en la energía libre de la secuencia completa no es muy fiable la predicción; es por eso que se recurre a zonas más puntuales con este tipo de enfoques de Nearest Neighbor ΔG. Sin embargo, este conjunto de datos presentó una gran variabilida en cuanto a las predicciones, por lo que debería revisarse bien el procedimiento de obtención o, más convenientemente, cambiar de enfoque. Por ejemplo podemos encontrar directamente la secuencia donde se pegan los promotores debido a que presentan una conservación mayor.
