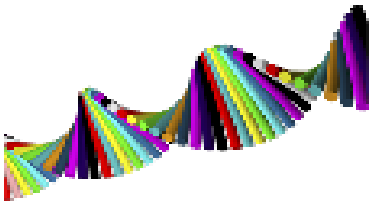
Laboratory of Natural Information Processing
DA-IICT Gandhinagar

Biospectrogram

Encodings & Transformations



BIOSPECTROGRAM

Encodings & Transformations

Laboratory of Natural Information Processing
DA-IICT, Gandhinagar, Gujarat 382007
http://www.guptalab.org/biospectrogram

Documentation version 1.0

This file last modified on December 28, 2013.

Credits & Team

Principle Investigator: Manish K. Gupta, PhD.
Key Developers: Nilay Chheda* and Naman Turakhia*
Graduate Mentor: Vandana Ravindran
Supporting Developers: Ruchin Shah and Jigar Raisinghani
Software Logo: Hiren Kangad

* Key developers contributed equally to the project

Acknowledgments

We thank Deeksha Gupta for useful discussion in the early stage of the project. All the icons are taken from the internet from the link "http://icons.mysitemyway.com/category/yellow-road-sign-icons/" These icons are free to use for personal as well as commercial use "http://icons.mysitemyway.com/terms-of-use/" MATLAB is registered trademark of Mathworks, USA. For FFT java script we thank Silvere Martin-Michiellot and also to Craig A. Lindley whose open source code has been used with modifications.

Biospectrogram Encodings

- **Indicator Encoding (E01.1.1/2/3/4 for A/C/G/T) (Vos, 1992)**

  First dialog box asks to enter character A, C, G or T. Based on the user input one of the four characters is encoded as 1 and the others are encoded as 0. If user chooses A, mapping would be,

$$\begin{array}{c|c} A \rightarrow 1 & C \rightarrow 0 \\ \hline G \rightarrow 0 & T \rightarrow 0 \end{array}$$

- **Tetrahedron Encoding (E01.2) (Silverman and Linsker, 1986)**

  This encoding does not ask for any user input. It is computed using the calculation shown below.

$$\begin{aligned}
&A \rightarrow \hat{k} = (a_r, a_g, a_b) \\
&C \rightarrow \tfrac{-2\sqrt{2}}{3}\hat{i} + \tfrac{\sqrt{6}}{3}\hat{j} - \tfrac{1}{3}\hat{k} = (c_r, c_g, c_b) \\
&G \rightarrow \tfrac{-2\sqrt{2}}{3}\hat{i} - \tfrac{\sqrt{6}}{3}\hat{j} - \tfrac{1}{3}\hat{k} = (g_r, g_g, g_b) \\
&T \rightarrow \tfrac{-2\sqrt{2}}{3}\hat{i} - \tfrac{1}{3}\hat{k} = (t_r, t_g, t_b) \\
&\forall l \in \{r, g, b\}, \\
&\chi_l[x] = a_l\chi_a(x) + c_l\chi_c(x) + g_l\chi_g(x) + t_l\chi_t(x) \\
&x \in \textstyle\sum_{DNA}^{*} \text{ and } \chi_A(x) \text{ etc. are indicator sequences} \\
&\chi_r(x) = \tfrac{\sqrt{2}}{3}\left(-\chi_c(x) - \chi_g(x) + 2\chi_t(x)\right) \\
&\chi_g(x) = \tfrac{\sqrt{6}}{3}\left(\chi_c(x) - \chi_g(x)\right) \\
&\chi_b(x) = \tfrac{1}{3}\left(3\chi_a(x) - \chi_c(x) - \chi_g(x) - \chi_t(x)\right)
\end{aligned}$$

- **Z Curve Encoding (E01.3) (Zhang and Zhang, 1994)**

  This encoding does not ask for any user input. It is computed using the calculation shown below.

$$\begin{bmatrix} x_n \\ y_n \\ z_n \end{bmatrix} = 2 \times \begin{pmatrix} 1010 \\ 1100 \\ 1001 \end{pmatrix} \times \begin{pmatrix} \chi_A(x) \\ \chi_C(x) \\ \chi_G(x) \\ \chi_T(x) \end{pmatrix} - \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$$

- **DV Curve Encoding (E01.4) (Zhang, 2009)**

  This encoding does not ask for any user input. It is computed using the mapping shown below.

$$\begin{array}{c|c} A \rightarrow \{1,1,1,1\} & C \rightarrow \{1,-1,1,1\} \\ \hline G \rightarrow \{1,-1,1,-1\} & T \rightarrow \{1,1,1,-1\} \end{array}$$

- **Complex Encoding 1 (E02.1) (Anastassiou, 2001)**

  Users are first prompt whether they want to enter complex number of their own. If user chooses no, mapping shown below is used for encoding.

$$\begin{array}{c|c} A \rightarrow 1+i & C \rightarrow -1-i \\ \hline G \rightarrow -1+i & T \rightarrow 1-i \end{array}$$

- **Complex Encoding (User choice) (E02.2)**

  Users are first prompt whether they want to enter complex number of their own. If user chooses yes, They are shown four different dialog box, two for complex number z1 and two for complex number z2. For both z1 and z2, users are asked to enter real and imaginary part of z1 and z2 in different dialog box. These inputs should be real numbers.

$$\begin{array}{c|c} A \rightarrow z_1 & C \rightarrow z_2 \\ \hline G \rightarrow \bar{z_2} & T \rightarrow \bar{z_1} \end{array}$$

- **Complex Encoding 2 (E02.3) (Rao and Shepherd, 2004)**

  This encoding does not ask for any user input. It is computed using the mapping shown below.

$$\begin{array}{c|c} A \rightarrow 1 & C \rightarrow -i \\ \hline G \rightarrow -1 & T \rightarrow i \end{array}$$

- **Random Complex Encoding (E02.4)**

  This encoding does not ask for any user input. It simply generates four random complex numbers and assign them to four nucleotides. Real and imaginary part of these random numbers are always taken from the range 0.01 to 9.99

- **Graphical Encoding 1 (E03.1) (Liao, 2005)**

  Users are prompt with two user input m & n. Users are supposed to enter a natural numbers only. After entering valid inputs, encoding is done as shown below.

$$\begin{array}{c|c} A \rightarrow (m, -n) & C \rightarrow (m, n) \\ \hline G \rightarrow (n, -m) & T \rightarrow (n, m) \end{array}$$

- **Graphical Encoding 2 (E03.2) (Yau *et al.,* 2003)**

  This encoding does not ask for any user input. It is computed using the mapping shown below.

$$\begin{array}{c|c} A \rightarrow \left(\frac{1}{2}, -\frac{\sqrt{3}}{2}\right) & C \rightarrow \left(\frac{\sqrt{3}}{2}, \frac{1}{2}\right) \\ \hline G \rightarrow \left(\frac{\sqrt{3}}{2}, -\frac{1}{2}\right) & T \rightarrow \left(\frac{1}{2}, \frac{\sqrt{3}}{2}\right) \end{array}$$

- **Random Graphical Encoding (E03.3)**

  This encoding does not ask for any user input. It simply generates two natural numbers from the range 10 to 99 and assign them to four nucleotides according to the formula used in Graphical Encoding 1.

- **Quaternion Encoding 1 (E04.1) (Brodzik and Peters, 2005; Akhtar *et al.*, 2007)**

  Users are first prompt whether they want to use Quaternion Encoding 1. If user chooses yes, mapping shown below is used for encoding.

$$\begin{array}{c|c} A \rightarrow i + j + k \equiv \{1, 1, 1\} & C \rightarrow i - j - k \equiv \{1, -1, -1\} \\ \hline G \rightarrow -i - j + k \equiv \{-1, -1, 1\} & T \rightarrow -i + j - k \equiv \{-1, 1, -1\} \end{array}$$

- **Quaternion Encoding 2 (E04.2) (Brodzik and Peters, 2005; Akhtar *et al.*, 2007)**

  Users are first prompt whether they want to use Quaternion Encoding 1. If user chooses no, mapping shown below is used for encoding.

$$\begin{array}{c|c} A \rightarrow 1 + i + j + k \equiv \{1, 1, 1, 1\} & C \rightarrow 1 + i - j - k \equiv \{1, 1, -1, -1\} \\ \hline G \rightarrow 1 - i - j + k \equiv \{1, -1, -1, 1\} & T \rightarrow 1 - i + j - k \equiv \{1, -1, 1, -1\} \end{array}$$

- **Real Value Encoding (E05.1) (Cristea, 2003; Rosen, 2006; Chakravarthy *et al.*, 2004)**

  Users are first prompt whether they want to enter real values of their own. If user chooses no, mapping shown below is used for encoding.

$A \rightarrow 1.5$	$C \rightarrow 0.5$
$G \rightarrow 0.5$	$T \rightarrow -1.5$

- **Real Value Encoding (User choice) (E05.2)**

  Users are first prompt whether they want to enter real values of their own. If user chooses yes, users are shown four different dialog box for four nucleotides. Real number inputs entered by user are directly used for mapping.

- **Electro Ion encoding (E05.3) (Ning *et al.*, 2003)**

  This encoding does not ask for any user input. It is computed using the mapping shown below.

$A \rightarrow 0.1260$	$C \rightarrow 0.1340$
$G \rightarrow 0.0806$	$T \rightarrow 0.1335$

- **Random Real Value Encoding (E05.4)**

  This encoding does not ask for any user input. It simply generates four real numbers from the range 0.01 to 9.99 and assign them to four nucleotides.

- **Quaternary Integer Mapping 1 (E06.1) (Dan Cristea, 2003)**

  This encoding does not ask for any user input. It is computed using the mapping shown below.

$A \rightarrow 2$	$C \rightarrow 1$
$G \rightarrow 3$	$T \rightarrow 0$

- **Quaternary Integer Mapping 2 (E06.2) (Dan Cristea, 2003)**

  This encoding does not ask for any user input. It is computed using the mapping shown below.

$A \rightarrow 0$	$C \rightarrow 2$
$G \rightarrow 1$	$T \rightarrow 3$

- **Protein Indicator Encoding (E07.1.1/2/3/4/5/6/7/8/9/10/11/12/13/14/15/16/17/18/19/20 for A/C/D/E/F/G/H/I/K/L/M/N/P/Q/R/S/T/V/W/Y)**

  First dialog box asks to enter one of the 20 characters. Based on the user input one of the twenty characters is encoded as 1 and the others are encoded as 0.

- **Protein Electro Ion Encoding (E07.2) (Vaidyanathan, 2005)**

  This encoding does not ask for any user input. It is computed using the mapping shown below.

$A \rightarrow 0.0373$	$C \rightarrow 0.0829$	$D \rightarrow 0.1263$	$E \rightarrow 0.0058$
$F \rightarrow 0.0946$	$G \rightarrow 0.0050$	$I \rightarrow 0.0000$	$H \rightarrow 0.0242$
$K \rightarrow 0.0371$	$L \rightarrow 0.0000$	$M \rightarrow 0.0823$	$N \rightarrow 0.0036$
$P \rightarrow 0.0198$	$Q \rightarrow 0.0761$	$R \rightarrow 0.0959$	$S \rightarrow 0.0829$
$T \rightarrow 0.0941$	$V \rightarrow 0.0057$	$W \rightarrow 0.0548$	$Y \rightarrow 0.0516$

- **Protein Real Value Encoding (User Choice) (E07.3)**

  Users are first prompt whether they want to enter real values of their own. If user chooses yes, users are shown twenty different dialog box for twenty nucleotides. Real number inputs entered by user are directly used for mapping.

- **Protein Random Real Value Encoding (E07.4)**

  This encoding does not ask for any user input. It simply generates twenty real numbers from the range 0.01 to 9.99 and assign them to twenty nucleotides.

Biospectrogram Transformations

- Fast Fourier Transform (T01)
- Hilbert Transform (T02)
- Z Transform (T03)
- Analytic Signal (T04)
- Discrete Haar Wavelet Transform (T05)
- Chirp Z Transform (T06)

For a sequence of N complex numbers $x_0, x_1, x_2, \ldots, x_{N-1}$,

Transform	Formula	Remarks
Fast Fourier Transform	$X_k = \sum_{n=0}^{N-1} x_n \cdot e^{-i 2\pi k n / N}$, $where\ k = 0,1,\ldots,N-1$	N is a power of 2. If length of sequence is not a power of 2, it is zero padded to make its length equal to the nearest power of 2
Discrete Haar Wavelet Transform	$y_{low} = (x * g)\downarrow 2$, approximation coefficients $y_{high} = (x * h)\downarrow 2$, detailed coefficients	where g is a low pass filter where as h is a high pass filter.
Hilbert transform in frequency domain	$\mathcal{F}(H(u))(\omega) = (-i\ \mathrm{sgn}(\omega)) \cdot \mathcal{F}(u)(\omega)$	where u(n)= x_n is the input sequence
Analytic Signal	$x_n + i * h_n$	where h_n is Hilbert transform of input sequence x_n in time domain
Z transform	$X(z) = \sum_{n=0}^{N-1} x_n * z^{-n}$	
Chirp Z transform	$X(z_k) = \sum_{n=0}^{N-1} x_n * {z_k}^{-n}$	If contour is a circle of radius r and z_k are N equally spaced points, then $z_k = re^{j2\pi k/N}$. Biospectrogram provides following variables: $z_k = ae^{j2\pi ku/v}$, where $k = 0,1,\ldots,m-1$. Providing u=1, v=N as input will give N equally spaced points on the circle of radius a.

References

[1] M. Akhtar, J. Epps, and E. Ambikairajah. On DNA numerical representations for period3 based exon prediction. In *Genomic Signal Processing and Statistics, 2007. GENSIPS 2007. IEEE International Workshop*, pp. 1 –4, June 2007.

[2] D. Anastassiou. Genomic signal processing. *Signal Processing Magazine, IEEE*, 18(4):8– 20, Jul 2001.

[3] A. K. Brodzik and O. Peters. Symbol-balanced quaternionic periodicity transform for latent pattern detection in DNA sequences. In *Proceedings. (ICASSP '05). IEEE International Conference on Acoustics, Speech, and Signal Processing, 2005.*, pp. 373–376. Philadelphia, Pennsylvania, USA, IEEE, 2005.

[4] N. Chakravarthy, A. Spanias, L. D. lasemidis, and K. Tsakalis. Autoregressive modeling and feature analysis of DNA sequences,. *EURASIP JASP,*, 1:13–28, 2004.

[5] P.D. Cristea. Phase analysis of DNA genomic signals. *Proceedings of the International Symposium on Circuits and Systems, 2003. ISCAS '03.*, 5:V–25–V–28, 2003.

[6] Paul Dan Cristea. Large scale features in DNA genomic signals. *Signal Process.*, 83(4):871–888, April 2003.

[7] B. Liao. A 2d graphical representation of DNA sequence,. *Chem Phys Lett,*, 401:196– 199, January 2005.

[8] J. Ning, C.N. Moore, and J.C. Nelson. Preliminary wavelet analysis of genomic sequences. In *Bioinformatics Conference, 2003. CSB 2003. Proceedings of the 2003 IEEE*, pp. 509–510, Aug. 2003.

[9] N. Rao and S.J. Shepherd. Detection of 3-periodicity for small genomic sequences based on ar technique. In *Communications, Circuits and Systems, 2004. ICCCAS 2004. 2004 International Conference on*, volume 2, pp. 1032 – 1036 Vol.2, June 2004.

[10] G. L. Rosen. Signal processing for biologically inspired gradient source localization and DNA sequence analysis,. *PhD thesis, Georgia Institute of Technology,*, Aug. 2006.

[11] B. D. Silverman and R. Linsker. A measure of DNA periodicity. *Journal of Theoretical Biology*, 118:295–300, 1986.

[12] P. P. Vaidyanathan. Genomics and proteomics: A signal processors tour. *IEEE Circuits Syst. Mag*, 4:6–29, 2005.

[13] Richard F. Voss. Evolution of long-range fractal correlations and 1/f noise in DNA base sequences. *Physical Review Letters*, 68(25):3805+, June 1992.

[14] S. T. Yau, J. Wang, A. Niknejad, C. Lu, N. Jin, and Yee-Kin Ho. DNA sequence representation without degeneracy. *Nucleic Acids Research*, 31(12):3078–3080, June 2003.

[15] R. Zhang and C. T. Zhang. Z curves, an intuitive tool for visualizing and analyzing the DNA sequences. *J. Biomol. Struct. Dyn.*, 11(4):767–782, 1994.

[16] Zhu-Jin Zhang. DV-curve: a novel intuitive tool for visualizing and analyzing DNA sequences. *Bioinformatics*, 25(9):1112–1117, March 2009.