Parameter Reference for FStitch

Michael Gohde

December 6, 2017

Abstract

FStitch is a tool that learns and generates annotations for regions of transcription on GRO or ChIP-seq data. This document exists to provide a quick reference on how to invoke and properly use the new interface for FStitch.

1 Training Parameters

In order to train a new model, FStitch needs, at minimum, the following information:

1. *A training input file.* FStitch needs to learn how to segment an input dataset by analyzing existing segments and regions of interest.
2. *A BedGraph file.* FStitch needs an input dataset to be able to make inferences about what each labeled region represents.
3. *An output file.* Once a model is generated, it needs to be stored so that FStitch can be applied to segment input data.

Below is a table of all parameters corresponding to the above list of required information.

Parameter	**Description**
-r *read bedgraph*	This parameter allows the user to specify a bedgraph file for FStitch to use as its input dataset.
-o *output file*	This parameter allows the user to specify a training output file.
-a *Annotation file name*	This parameter allows the user to specify a training input file.

There are a number of additional parameters that present options relating to how the input data is to be specified and processed:

Parameter	**Description**
–posfile *positive strand data* –negfile *negative strand data*	These options can be used in place of the -r parameter to allow the user to split the input dataset into positive and negative strand data.
–onfile *on training examples* –offfile *off training examples*	These options can be used in place of the -a parameter to allow the user to split the input training file into on and off regions.
-chip	This parameter allows the user to use ChIP-seq (Chromatin ImmunoPrecipitation sequencing) input data instead of the default GRO-seq (Genomic Run-On sequencing).

The training process itself can be altered through the use of input parameters. Below is a table of such parameters and their default values.

Parameter	**Default value**	**Description**
-cm *value*	100	Maximum learning iterations
-ct *value*	0.001	Convergence threshold
-lr *value*	0.4	Learning rate
-reg *value*	1	Regularization
-ms *value*	20	Maximum seed value
–strand *'+' or '-' or 'both'*	both or +, depending on inputs	Which strand to attempt to train on (if applicable).

2 Segmentation Parameters

In order to segment an input dataset based on a trained model, FStitch needs, at minimum, the following information:

1. *A trained model*
2. *An input dataset*
3. *An output file*

Below is a table of all parameters corresponding to the above list of required information:

Parameter	**Description**
-w *model or "weights" file*	This parameter allows the user to specify a training input file.
-r *bedgraph file*	This parameter allows the user to specify a bedgraph file for FStitch to use as its input dataset.
-o *output file*	This parameter specifies the name of the output annotations bed file that FStitch will write.

There are a few additional parameters that present options relating to how input and output data are to be processed:

Parameter	**Description**
–report *'on' or 'off' or 'both'*	Whether to only report "on" regions, "off" regions, or both "on" and "off" regions.
–strand *'+' or '-' or 'both'*	Which strand to attempt to train on (if applicable).

3 Parameters Common to Both Training and Segmentation

There are two classes of common parameters:

1. Parameters that change the operation of FStitch regardless of the command used.
2. Parameters that behave similarly regardless of the command used.

The following table documents the first class of common parameters:

Parameter	**Description**
-v	This option enables verbose logging.
-np *number of processors*	This option allows FStitch to use up to the number of CPU threads specified to perform its tasks.

The following table documents the second class of common parameters:

Parameter	**Description**
-r *read bedgraph*	Both ‘train’ and ‘segment’ need an input bedgraph file, so this parameter is the same for both.
-o *output file*	Despite producing different kinds of outputs, both ‘train’ and ‘segment’ require the specification of an output file.
-chip	This parameter is necessary for both training and segmentation due to differences in how various sequencing protocols operate.
–strand	Which strand to attempt to train on (if applicable).