instagraal Documentation

Release 0.1.4

Lyam Baudry, Hervé Marie-Nelly

Jan 27, 2023

Contents:

1 Reference API **3**
1.1 instagraal 3

2 Indices and tables **15**

Python Module Index **17**

Index **19**

InstaGRAAL is a Hi-C based scaffolder written with large genome assemblies in mind. As such, it also provides a comprehensive framework for generating, storing and analyzing Hi-C data, as well as a number of polishing libraries for post-scaffolding use. The present documentation is dedicated to exposing the various APIs to users who wish to delve deeper into the internal implementation of the scaffolder in order to better understand how it works and possibly extend its functionalities. The software is under continuous development with various incoming overhauls, and the documentation will improve accordingly.

CHAPTER 1

Reference API

1.1 instagraal

1.1.1 instagraal package

Submodules

instagraal.cuda_lib_gl_single module

class instagraal.cuda_lib_gl_single.**sampler**(*use_rippe*, *S_o_A_frags*, *collector_id_repeats*, *frag_dispatcher*, *id_frag_duplicated*, *id_frags_blacklisted*, *n_frags*, *n_new_frags*, *init_n_sub_frags*, *n_new_sub_frags*, *np_rep_sub_frags_id*, *sub_sampled_sparse_matrix*, *np_sub_frags_len_bp*, *np_sub_frags_id*, *np_sub_frags_accu*, *np_sub_frags_2_frags*, *mean_squared_frags_per_bin*, *norm_vect_accu*, *sub_candidates_dup*, *sub_candidates_output_data*, *S_o_A_sub_frags*, *sub_collector_id_repeats*, *sub_frag_dispatcher*, *sparse_matrix*, *mean_value_trans*, *n_iterations*, *is_simu*, *gl_window*, *pos_vbo*, *col_vbo*, *vel*, *pos*, *raw_im_init*, *pbo_im_buffer*, *gl_size_im*)

Bases: object

apply_replay_simu(*id_fA*, *id_fB*, *op_sampled*, *dt*)

approx_all_likelihood_on_zeros()

approx_single_likelihood_on_zeros()

approx_single_likelihood_on_zeros_mut()

approx_single_likelihood_on_zeros_nuisance()

bomb_the_genome()

create_gpu_struct(*data*)

display_current_matrix(*filename*)

dist_inter_genome(*tmp_gpu_vect_frags*)

estimate_parameters(*max_dist_kb*, *size_bin_kb*, *display_graph*)
estimation by least square optimization of Rippe parameters on the experimental data :param max_dist_kb: :param size_bin_kb:

estimate_parameters_rippe(*max_dist_kb*, *size_bin_kb*, *display_graph*)
estimation by least square optimization of Rippe parameters on the experimental data :param max_dist_kb: :param size_bin_kb:

eval_all_sub_likelihood()

eval_likelihood()

eval_likelihood_4_nuisance()

eval_likelihood_init()

eval_likelihood_on_mut(*id_mut*)

explode_genome(*dt*)

extract_current_sub_likelihood()

extract_uniq_mutations(*id_fi*, *id_fj*, *flip_eject*)

f_hic(*x*, *param*)

f_rippe(*x*, *param*)

fill_dist_all_mut()

fill_dist_single()

fill_dist_single_mut(*id_mut*)

free_gpu()

genome_content()

insert_blocks(*id_fA*, *id_fB*, *max_id*)

insert_repeats(*id_f_ins*)

loadProgram(*filename*)

load_gl_cuda_tex_buffer(*im_init*)

load_gl_cuda_vbo()

local_flip(*id_fA*, *mode*, *max_id*)

modify_genome(*n*)

modify_gl_cuda_buffer(*id_fi*, *dt*)

modify_image_thresh(*val*)

modify_param_simu(*param_simu*, *id_val*, *val*)

perform_mutations(*id_fA*, *id_fB*, *max_id*, *is_first*)

pop_out_pop_in(*id_f_pop*, *id_f_ins*, *mode*, *max_id*)

prepare_sparse_call()

return_neighbours(*id_fA*, *delta0*)

return_rippe_vals(*p0*)

set_jumping_distributions_parameters(*delta*)

setup_all_gpu_struct()

setup_distri_frags()

setup_model_parameters(*param*, *d_max*)

setup_rippe_parameters(*param*, *d_max*)

setup_rippe_parameters_4_simu(*kuhn*, *lm*, *slope*, *d*, *val_inter*, *d_max*)

setup_thrust_modules()

show_sub_slice(*id_ctg1*, *id_ctg2*, *id_frag_a*, *id_frag_b*)

slice_sparse_mat(*id_ctg1*, *id_ctg2*, *id_fragA*, *id_fragB*)

sparse_data_2_gpu()

sparse_data_4_gl(*precision*)

step_nuisance_parameters(*dt*, *t*, *n_step*)

step_sampler(*id_frag*, *n_neighbours*, *dt*)

step_sampler_debug(*id_frag*, *n_neighbours*)

temperature(*t*, *n_step*)

test_copy_struct(*id_fA*, *id_f_sampled*, *mode*, *max_id*)

test_thrust()

transloc(*id_fA*, *id_fB*, *max_id*)

update_neighbourhood()

instagraal.fragment module

class instagraal.fragment.**basic_fragment**

Bases: object

classmethod initiate(*np_id_abs*, *id_init*, *init_contig*, *curr_id*, *start_pos*, *end_pos*, *length_kb*, *gc_content*, *init_frag_start*, *init_frag_end*, *sub_frag_start*, *sub_frag_end*, *super_index*, *id_contig*, *n_accu_frags*)

class instagraal.fragment.**fragment**

Bases: object

classmethod copy(*frag*)

classmethod initiate(*np_id_abs*, *id_init*, *init_contig*, *curr_id*, *start_pos*, *end_pos*, *length_kb*, *gc_content*)

update_name(*contig_id*)

instagraal.glutil module

instagraal.glutil.**draw_axes**()

instagraal.glutil.**draw_line**(*v1*, *v2*)

instagraal.glutil.**init**(*width*, *height*)

instagraal.glutil.**lights**()

instagraal.gpustruct module

class instagraal.gpustruct.**GPUStruct**(*objs*)

Bases: object

copy_from_gpu(*skip=None*)

copy_to_gpu(*skip=None*)

get_packed()

get_ptr()

instagraal.init_nuisance module

instagraal.init_nuisance.**estimate_max_dist_intra**(*p*, *val_inter*)

instagraal.init_nuisance.**estimate_param_hic**(*y_meas*, *x_bins*)

instagraal.init_nuisance.**log_residuals**(*param*, *y*, *x*)

instagraal.init_nuisance.**log_residuals_4_min**(*param*, *y*, *x*)

instagraal.init_nuisance.**peval**(*x*, *param*)

instagraal.init_nuisance.**residual_4_max_dist**(*x*, *p*)

instagraal.init_nuisance.**residuals**(*param*, *y*, *x*)

instagraal.instagraal module

Large genome reassembly based on Hi-C data.

Usage:

instagraal <hic_folder> <reference.fa> [<output_folder>] [–level=4] [–cycles=100] [–coverage-std=1] [–neighborhood=5] [–device=0] [–circular] [–bomb] [–save-matrix] [–pyramid-only] [–save-pickle] [–simple] [–quiet] [–debug]

Options:

-h, --help	Display this help message.
--version	Display the program's current version.

-l 4, –level 4 Level (resolution) of the contact map. Increasing level by one means a threefold smaller resolution but also a threefold faster computation time. [default: 4]

-n 100, –cycles 100 Number of iterations to perform for each bin. (row/column of the contact map). A high number of cycles has diminishing returns but there is a necessary minimum for assembly convergence. [default: 100]

-c 1, –coverage-std 1 Number of standard deviations below the mean. coverage, below which fragments should be filtered out prior to binning. [default: 1]

-N 5, –neighborhood 5 Number of neighbors to sample for potential mutations for each bin. [default: 5]

–device 0 If multiple graphic cards are available, select a specific device (numbered from 0). [default: 0]

-C, --circular	Indicates genome is circular. [default: False]
-b, --bomb	Explode the genome prior to scaffolding. [default: False]
--pyramid-only	Only build multi-resolution contact maps (pyramids) and don't do any scaffolding. [default: False]
--save-pickle	Dump all info from the instaGRAAL run into a pickle. Primarily for development purposes, but also for advanced post hoc introspection. [default: False]
--save-matrix	Saves a preview of the contact map after each cycle. [default: False]
--simple	Only perform operations at the edge of the contigs. [default: False]
--quiet	Only display warnings and errors as outputs. [default: False]
--debug	Display debug information. For development purposes only. Mutually exclusive with –quiet, and will override it. [default: False]

instagraal.instagraal.**main**()

class instagraal.instagraal.**window**(*name*, *folder_path*, *fasta*, *device*, *level*, *n_iterations_em*, *n_iterations_mcmc*, *is_simu*, *scrambled*, *perform_em*, *use_rippe*, *gl_size_im*, *sample_param*, *thresh_factor*, *output_folder*)

Bases: object

A window displaying the live movie of the calculations performed by the scaffolder.

[description]

Parameters

- **name** (*str*) – The name of the project. Will determine the window title.
- **folder_path** (*str or pathlib.Path*) – The directory containing the Hi-C conact map.
- **fasta** (*str or pathlib.Path*) – The path to the reference genome in FASTA format.
- **device** (*int*) – The identifier of the graphic card to be used, numbered from 0. If only one is available, it should be 0.
- **level** (*int*) – The level (resolution) at which to perform scaffolding.
- **n_iterations_em** (*int*) – The number of EM (expectation maximization) iterations.
- **n_iterations_mcmc** (*int*) – The number of MCMC (Markov chain Monte-Carlo) iterations.
- **is_simu** (*bool*) – Whether the parameters should be simulated. Mutually exclusive with use_rippe and will override it.
- **scrambled** (*bool*) – Whether to scramble the genome.
- **perform_em** (*bool*) – Whether to perform EM (expectation maximization).
- **use_rippe** (*bool*) – Whether to explicitly use the model from Rippe et al., 2001.

- **gl_size_im** (*int*) – The size of the window to be displayed.
- **sample_param** (*bool*) – Whether to sample the parameters.
- **thresh_factor** (*float*) – The sparsity (coverage) threshold below which fragments are discarded, as a number of standard deviations below the mean.
- **output_folder** (*str or pathlib.Path*) – The path to the output folder where the scaffolded genome and other relevant information will be saved.

cuda_gl_init()

debug_test_EM(*delta*)

debug_test_model(*id_fi*, *delta*)

draw()
Render the particles

full_em(*n_cycles*, *n_neighbours*, *bomb*, *id_start_sample_param*, *save_matrix=False*)

glinit()

glut_print(*x*, *y*, *font*, *text*, *r*, *g*, *b*, *a*)

modify_image_thresh(*val*)
modify threshold of the matrix

on_click(*button*, *state*, *x*, *y*)

on_key(**args*)

on_mouse_motion(*x*, *y*)

remote_update()

render()

render_image()

replay_simu(*file_simu*, *file_likelihood*, *file_n_contigs*, *file_distances*)

save_behaviour_to_txt()

setup_simu(*id_f_ins*)

simple_start(*n_cycles*, *n_neighbours*, *bomb*)

start_EM()

start_EM_all()

start_EM_no_scrambled()

start_EM_nuisance()

start_MCMC()

start_MTM()

test_model(*id_fi*, *delta*)

timer(*t*)

instagraal.leastsqbound module

Constrained multivariate Levenberg-Marquardt optimization

An updated version of this file can be found at https://github.com/jjhelmus/leastsqbound-scipy

The version here has known bugs which have been fixed above, proceed at your own risk.

- Jonathan J. Helmus (jjhelmus@gmail.com)

instagraal.leastsqbound.**calc_cov_x**(*infodic*, *p*)
Calculate cov_x from fjac, ipvt and p as is done in leastsq

instagraal.leastsqbound.**err**(*p*, *bounds*, *efunc*, *args*)

instagraal.leastsqbound.**external2internal**(*xe*, *bounds*)
Convert a series of external variables to internal variables

instagraal.leastsqbound.**i2e_cov_x**(*xi*, *bounds*, *cov_x*)

instagraal.leastsqbound.**internal2external**(*xi*, *bounds*)
Convert a series of internal variables to external variables

instagraal.leastsqbound.**internal2external_grad**(*xi*, *bounds*)
Calculate the internal to external gradiant

Calculates the partial of external over internal

instagraal.leastsqbound.**leastsqbound**(*func*, *x0*, *bounds*, *args=()*, ***kw*)
Constrained multivariant Levenberg-Marquard optimization

Minimize the sum of squares of a given function using the Levenberg-Marquard algorithm. Contraints on parameters are inforced using variable transformations as described in the MINUIT User's Guide by Fred James and Matthias Winkler.

Parameters:

- func functions to call for optimization.
- x0 Starting estimate for the minimization.
- **bounds (min,max) pair for each element of x, defining the bounds on** that parameter. Use None for one of min or max when there is no bound in that direction.
- args Any extra arguments to func are places in this tuple.

Returns: (x,{cov_x,infodict,mesg},ier)

Return is described in the scipy.optimize.leastsq function. x and con_v are corrected to take into account the parameter transformation, infodic is not corrected.

Additional keyword arguments are passed directly to the scipy.optimize.leastsq algorithm.

instagraal.linkage module

instagraal.log module

Basic logging setup for instaGRAAL.

Logging level can be set by the user and determines the verbosity of the whole program.

instagraal.optim_rippe_curve_update module

instagraal.optim_rippe_curve_update.**estimate_max_dist_intra**(*p*, *val_inter*)

instagraal.optim_rippe_curve_update.**estimate_max_dist_intra_nuis**(*p*, *val_inter*, *old_s*)

instagraal.optim_rippe_curve_update.**estimate_param_rippe**(*y_meas*, *x_bins*)

instagraal.optim_rippe_curve_update.**log_peval**(*x*, *param*)

instagraal.optim_rippe_curve_update.**log_residuals**(*p*, *y*, *x*)

instagraal.optim_rippe_curve_update.**peval**(*x*, *param*)

instagraal.optim_rippe_curve_update.**residual_4_max_dist**(*x*, *p*)

instagraal.optim_rippe_curve_update.**residuals**(*p*, *y*, *x*)

instagraal.parse_info_frags module

instagraal.pyramid_sparse module

Pyramid library

Create and handle so-called 'pyramid' objects, i.e. a series of decreasing-resolution contact maps in hdf5 format.

instagraal.pyramid_sparse.**abs_contact_2_coo_file**(*abs_contact_file*, *coo_file*)

Convert contact maps between old-style and new-style formats.

A legacy function that converts contact maps from the older GRAAL format to the simpler instaGRAAL format. This is useful with datasets generated by Hi-C box.

Parameters

- **abs_contact_file** (*str, file or pathlib.Path*) – The input old-style contact map.
- **coo_file** (*str, file, or pathlib.Path*) – The output path to the generated contact map; must be writable.

instagraal.pyramid_sparse.**build**(*base_folder*, *size_pyramid*, *factor*, *min_bin_per_contig*)

Build a pyramid of contact maps

Build a fragment pyramid for multi-scale analysis

Parameters

- **base_folder** (*str or pathlib.Path*) – Where to create the hdf5 files containing the matrices.
- **size_pyramid** (*int*) – How many levels (contact maps of decreasing resolution) to generate.
- **factor** (*int*) – Subsampling factor (binning) from one level to the next.
- **min_bin_per_contig** (*int*) – The minimum number of bins per contig below which binning shall not be performed.

instagraal.pyramid_sparse.**build_and_filter**(*base_folder*, *size_pyramid*, *factor*, *thresh_factor=1*)

Build a filtered pyramid of contact maps

Build a fragment pyramid for multi-scale analysis and remove high sparsity (i.e. low-coverage) and short fragments.

Parameters

- **base_folder** (*str or pathlib.Path*) – Where to create the hdf5 files containing the matrices.
- **size_pyramid** (*int*) – How many levels (contact maps of decreasing resolution) to generate.
- **factor** (*int*) – Subsampling factor (binning) from one level to the next.
- **thresh_factor** (*float, optional*) – Number of standard deviations below the mean coverage beyond which lesser covered fragments will be discarded. Default is 1.

Returns **obj_pyramid** – The pyramid object containing all the levels.

Return type Pyramid

instagraal.pyramid_sparse.**file_len**(*fname*)

instagraal.pyramid_sparse.**fill_sparse_pyramid_level**(*pyramid_handle*, *level*, *contact_file*, *nfrags*)

Fill a level with sparse contact map data

Fill values from the simple text matrix file to the hdf5-based pyramid level with contact data.

Parameters

- **pyramid_handle** (*h5py.File*) – The hdf5 file handle containing the whole dataset.
- **level** (*int*) – The level (resolution) to be filled with contact data.
- **contact_file** (*str, file or pathlib.Path*) – The binned contact map file to be converted to hdf5 data.
- **nfrags** (*int*) – The number of fragments/bins in that specific level.

instagraal.pyramid_sparse.**get_contig_info_from_file**(*contig_info*)

instagraal.pyramid_sparse.**get_frag_info_from_fil**(*fragments_list*)

instagraal.pyramid_sparse.**init_frag_list**(*fragment_list*, *new_frag_list*)

Adapt the original fragment list to fit the build function requirements

Parameters

- **fragment_list** (*str, file or pathlib.Path*) – The input fragment list.
- **new_frag_list** (*str, file or pathlib.Path*) – The output fragment list to be written.

Returns **i** – The number of records processed this way.

Return type int

class instagraal.pyramid_sparse.**level**(*pyramid*, *level*)

Bases: object

build_seq_per_bin(*genome_fasta*)

define_inter_chrom_coord()

generate_new_fasta(*vect_frags*, *new_fasta*, *info_frags*)

init_data()

load_data(*pyramid*)

Parameters **pyramid** – hic pyramid

instagraal.pyramid_sparse.**main**()

instagraal.pyramid_sparse.**new_remove_problematic_fragments**(*contig_info*, *fragments_list*, *abs_fragments_contacts*, *new_contig_list_file*, *new_fragments_list_file*, *new_abs_fragments_contacts_file*, *pyramid*)

class instagraal.pyramid_sparse.**pyramid**(*pyramid_folder*, *n_levels*)

Bases: object

build_frag_dictionnary(*fragments_list*, *level*)

close()

full_zoom_in_frag(*curr_frag*)

Parameters **curr_frag** –

get_level(*level_id*)

load_reference_sequence(*genome_fasta*)

update_super_index(*dict_frag*, *super_index_file*)

update_super_index_in_dict_contig(*dict_frag*, *dict_contig*)

zoom_in_area(*area*)

zoom in area

zoom_in_frag(*curr_frag*)

Parameters **curr_frag** –

zoom_in_pixel(*curr_pixel*)

return the curr_frag at a higher resolution

zoom_out_frag(*curr_frag*)

Parameters **curr_frag** –

zoom_out_pixel(*curr_pixel*)

return the curr_frag at a lower resolution

instagraal.pyramid_sparse.**remove_problematic_fragments**(*contig_info*, *fragments_list*, *abs_fragments_contacts*, *new_contig_list_file*, *new_fragments_list_file*, *new_abs_fragments_contacts_file*, *pyramid*, *thresh_factor=1*)

instagraal.pyramid_sparse.**subsample_data_set**(*contig_info*, *fragments_list*, *fact_sub_sample*, *abs_fragments_contacts*, *new_abs_fragments_contacts_file*, *min_bin_per_contig*, *new_contig_list_file*, *new_fragments_list_file*, *old_2_new_file*)

instagraal.simu_single module

instagraal.simu_single.**kth_diag_indices**(*a*, *k*)

class instagraal.simu_single.**simulation**(*name*, *folder_path*, *fasta*, *level*, *n_iterations*, *is_simu*, *gl_window*, *use_rippe*, *gl_size_im*, *thresh_factor=1*, *output_folder=None*)

Bases: object

blacklist_contig()

create_new_sub_frags()

create_sub_frags()

discard_low_coverage_frags()

export_new_fasta()

init_gl_image()

load_gl_buffers()

modify_sub_vect_frags()
include repeated frags

modify_vect_frags()
include repeated frags

plot_info_simu(*collect_likelihood_input*, *collect_n_contigs_input*, *file_plot*, *title_ax*)

release()

select_data_set(*name*)

select_repeated_frags()

instagraal.vector module

class instagraal.vector.**Vec**

Bases: numpy.ndarray

props = ['x', 'y', 'z', 'w']

instagraal.vector.**normalize**(*u*)

instagraal.version module

Module contents

CHAPTER 2

Indices and tables

- genindex
- modindex
- search

Python Module Index

i

instagraal, 13
instagraal.cuda_lib_gl_single, 3
instagraal.fragment, 5
instagraal.glutil, 6
instagraal.gpustruct, 6
instagraal.init_nuisance, 6
instagraal.instagraal, 6
instagraal.leastsqbound, 9
instagraal.log, 9
instagraal.optim_rippe_curve_update, 10
instagraal.pyramid_sparse, 10
instagraal.simu_single, 12
instagraal.vector, 13
instagraal.version, 13

Index

A

abs_contact_2_coo_file() (*in module instagraal.pyramid_sparse*), 10
apply_replay_simu() (*instagraal.cuda_lib_gl_single.sampler method*), 3
approx_all_likelihood_on_zeros() (*instagraal.cuda_lib_gl_single.sampler method*), 3
approx_single_likelihood_on_zeros() (*instagraal.cuda_lib_gl_single.sampler method*), 3
approx_single_likelihood_on_zeros_mut() (*instagraal.cuda_lib_gl_single.sampler method*), 3
approx_single_likelihood_on_zeros_nuisance() (*instagraal.cuda_lib_gl_single.sampler method*), 3

B

basic_fragment (*class in instagraal.fragment*), 5
blacklist_contig() (*instagraal.simu_single.simulation method*), 13
bomb_the_genome() (*instagraal.cuda_lib_gl_single.sampler method*), 4
build() (*in module instagraal.pyramid_sparse*), 10
build_and_filter() (*in module instagraal.pyramid_sparse*), 10
build_frag_dictionnary() (*instagraal.pyramid_sparse.pyramid method*), 12
build_seq_per_bin() (*instagraal.pyramid_sparse.level method*), 11

C

calc_cov_x() (*in module instagraal.leastsqbound*), 9
close() (*instagraal.pyramid_sparse.pyramid method*), 12
copy() (*instagraal.fragment.fragment class method*), 5
copy_from_gpu() (*instagraal.gpustruct.GPUStruct method*), 6
copy_to_gpu() (*instagraal.gpustruct.GPUStruct method*), 6
create_gpu_struct() (*instagraal.cuda_lib_gl_single.sampler method*), 4
create_new_sub_frags() (*instagraal.simu_single.simulation method*), 13
create_sub_frags() (*instagraal.simu_single.simulation method*), 13
cuda_gl_init() (*instagraal.instagraal.window method*), 8

D

debug_test_EM() (*instagraal.instagraal.window method*), 8
debug_test_model() (*instagraal.instagraal.window method*), 8
define_inter_chrom_coord() (*instagraal.pyramid_sparse.level method*), 11
discard_low_coverage_frags() (*instagraal.simu_single.simulation method*), 13
display_current_matrix() (*instagraal.cuda_lib_gl_single.sampler method*), 4
dist_inter_genome() (*instagraal.cuda_lib_gl_single.sampler method*), 4
draw() (*instagraal.instagraal.window method*), 8
draw_axes() (*in module instagraal.glutil*), 6
draw_line() (*in module instagraal.glutil*), 6

E

err() (*in module instagraal.leastsqbound*), 9
estimate_max_dist_intra() (*in module instagraal.init_nuisance*), 6
estimate_max_dist_intra() (*in module instagraal.optim_rippe_curve_update*), 10

estimate_max_dist_intra_nuis() (*in module instagraal.optim_rippe_curve_update*), 10
estimate_param_hic() (*in module instagraal.init_nuisance*), 6
estimate_param_rippe() (*in module instagraal.optim_rippe_curve_update*), 10
estimate_parameters() (*instagraal.cuda_lib_gl_single.sampler method*), 4
estimate_parameters_rippe() (*instagraal.cuda_lib_gl_single.sampler method*), 4
eval_all_sub_likelihood() (*instagraal.cuda_lib_gl_single.sampler method*), 4
eval_likelihood() (*instagraal.cuda_lib_gl_single.sampler method*), 4
eval_likelihood_4_nuisance() (*instagraal.cuda_lib_gl_single.sampler method*), 4
eval_likelihood_init() (*instagraal.cuda_lib_gl_single.sampler method*), 4
eval_likelihood_on_mut() (*instagraal.cuda_lib_gl_single.sampler method*), 4
explode_genome() (*instagraal.cuda_lib_gl_single.sampler method*), 4
export_new_fasta() (*instagraal.simu_single.simulation method*), 13
external2internal() (*in module instagraal.leastsqbound*), 9
extract_current_sub_likelihood() (*instagraal.cuda_lib_gl_single.sampler method*), 4
extract_uniq_mutations() (*instagraal.cuda_lib_gl_single.sampler method*), 4

F

f_hic() (*instagraal.cuda_lib_gl_single.sampler method*), 4
f_rippe() (*instagraal.cuda_lib_gl_single.sampler method*), 4
file_len() (*in module instagraal.pyramid_sparse*), 11
fill_dist_all_mut() (*instagraal.cuda_lib_gl_single.sampler method*), 4
fill_dist_single() (*instagraal.cuda_lib_gl_single.sampler method*), 4
fill_dist_single_mut() (*instagraal.cuda_lib_gl_single.sampler method*), 4
fill_sparse_pyramid_level() (*in module instagraal.pyramid_sparse*), 11
fragment (*class in instagraal.fragment*), 5
free_gpu() (*instagraal.cuda_lib_gl_single.sampler method*), 4
full_em() (*instagraal.instagraal.window method*), 8
full_zoom_in_frag() (*instagraal.pyramid_sparse.pyramid method*), 12

G

generate_new_fasta() (*instagraal.pyramid_sparse.level method*), 11
genome_content() (*instagraal.cuda_lib_gl_single.sampler method*), 4
get_contig_info_from_file() (*in module instagraal.pyramid_sparse*), 11
get_frag_info_from_fil() (*in module instagraal.pyramid_sparse*), 11
get_level() (*instagraal.pyramid_sparse.pyramid method*), 12
get_packed() (*instagraal.gpustruct.GPUStruct method*), 6
get_ptr() (*instagraal.gpustruct.GPUStruct method*), 6
glinit() (*instagraal.instagraal.window method*), 8
glut_print() (*instagraal.instagraal.window method*), 8
GPUStruct (*class in instagraal.gpustruct*), 6

I

i2e_cov_x() (*in module instagraal.leastsqbound*), 9
init() (*in module instagraal.glutil*), 6
init_data() (*instagraal.pyramid_sparse.level method*), 11
init_frag_list() (*in module instagraal.pyramid_sparse*), 11
init_gl_image() (*instagraal.simu_single.simulation method*), 13
initiate() (*instagraal.fragment.basic_fragment class method*), 5
initiate() (*instagraal.fragment.fragment class method*), 5
insert_blocks() (*instagraal.cuda_lib_gl_single.sampler method*), 4
insert_repeats() (*instagraal.cuda_lib_gl_single.sampler method*), 4
instagraal (*module*), 13

instagraal.cuda_lib_gl_single (*module*), 3
instagraal.fragment (*module*), 5
instagraal.glutil (*module*), 6
instagraal.gpustruct (*module*), 6
instagraal.init_nuisance (*module*), 6
instagraal.instagraal (*module*), 6
instagraal.leastsqbound (*module*), 9
instagraal.log (*module*), 9
instagraal.optim_rippe_curve_update (*module*), 10
instagraal.pyramid_sparse (*module*), 10
instagraal.simu_single (*module*), 12
instagraal.vector (*module*), 13
instagraal.version (*module*), 13
internal2external() (*in module instagraal.leastsqbound*), 9
internal2external_grad() (*in module instagraal.leastsqbound*), 9

K

kth_diag_indices() (*in module instagraal.simu_single*), 12

L

leastsqbound() (*in module instagraal.leastsqbound*), 9
level (*class in instagraal.pyramid_sparse*), 11
lights() (*in module instagraal.glutil*), 6
load_data() (*instagraal.pyramid_sparse.level method*), 11
load_gl_buffers() (*instagraal.simu_single.simulation method*), 13
load_gl_cuda_tex_buffer() (*instagraal.cuda_lib_gl_single.sampler method*), 4
load_gl_cuda_vbo() (*instagraal.cuda_lib_gl_single.sampler method*), 4
load_reference_sequence() (*instagraal.pyramid_sparse.pyramid method*), 12
loadProgram() (*instagraal.cuda_lib_gl_single.sampler method*), 4
local_flip() (*instagraal.cuda_lib_gl_single.sampler method*), 4
log_peval() (*in module instagraal.optim_rippe_curve_update*), 10
log_residuals() (*in module instagraal.init_nuisance*), 6
log_residuals() (*in module instagraal.optim_rippe_curve_update*), 10
log_residuals_4_min() (*in module instagraal.init_nuisance*), 6

M

main() (*in module instagraal.instagraal*), 7
main() (*in module instagraal.pyramid_sparse*), 11
modify_genome() (*instagraal.cuda_lib_gl_single.sampler method*), 4
modify_gl_cuda_buffer() (*instagraal.cuda_lib_gl_single.sampler method*), 4
modify_image_thresh() (*instagraal.cuda_lib_gl_single.sampler method*), 4
modify_image_thresh() (*instagraal.instagraal.window method*), 8
modify_param_simu() (*instagraal.cuda_lib_gl_single.sampler method*), 4
modify_sub_vect_frags() (*instagraal.simu_single.simulation method*), 13
modify_vect_frags() (*instagraal.simu_single.simulation method*), 13

N

new_remove_problematic_fragments() (*in module instagraal.pyramid_sparse*), 12
normalize() (*in module instagraal.vector*), 13

O

on_click() (*instagraal.instagraal.window method*), 8
on_key() (*instagraal.instagraal.window method*), 8
on_mouse_motion() (*instagraal.instagraal.window method*), 8

P

perform_mutations() (*instagraal.cuda_lib_gl_single.sampler method*), 4
peval() (*in module instagraal.init_nuisance*), 6
peval() (*in module instagraal.optim_rippe_curve_update*), 10
plot_info_simu() (*instagraal.simu_single.simulation method*), 13
pop_out_pop_in() (*instagraal.cuda_lib_gl_single.sampler method*), 4
prepare_sparse_call() (*instagraal.cuda_lib_gl_single.sampler method*), 5
props (*instagraal.vector.Vec attribute*), 13
pyramid (*class in instagraal.pyramid_sparse*), 12

R

release() (*instagraal.simu_single.simulation method*), 13
remote_update() (*instagraal.instagraal.window method*), 8
remove_problematic_fragments() (*in module instagraal.pyramid_sparse*), 12
render() (*instagraal.instagraal.window method*), 8
render_image() (*instagraal.instagraal.window method*), 8
replay_simu() (*instagraal.instagraal.window method*), 8
residual_4_max_dist() (*in module instagraal.init_nuisance*), 6
residual_4_max_dist() (*in module instagraal.optim_rippe_curve_update*), 10
residuals() (*in module instagraal.init_nuisance*), 6
residuals() (*in module instagraal.optim_rippe_curve_update*), 10
return_neighbours() (*instagraal.cuda_lib_gl_single.sampler method*), 5
return_rippe_vals() (*instagraal.cuda_lib_gl_single.sampler method*), 5

S

sampler (*class in instagraal.cuda_lib_gl_single*), 3
save_behaviour_to_txt() (*instagraal.instagraal.window method*), 8
select_data_set() (*instagraal.simu_single.simulation method*), 13
select_repeated_frags() (*instagraal.simu_single.simulation method*), 13
set_jumping_distributions_parameters() (*instagraal.cuda_lib_gl_single.sampler method*), 5
setup_all_gpu_struct() (*instagraal.cuda_lib_gl_single.sampler method*), 5
setup_distri_frags() (*instagraal.cuda_lib_gl_single.sampler method*), 5
setup_model_parameters() (*instagraal.cuda_lib_gl_single.sampler method*), 5
setup_rippe_parameters() (*instagraal.cuda_lib_gl_single.sampler method*), 5
setup_rippe_parameters_4_simu() (*instagraal.cuda_lib_gl_single.sampler method*), 5
setup_simu() (*instagraal.instagraal.window method*), 8
setup_thrust_modules() (*instagraal.cuda_lib_gl_single.sampler method*), 5
show_sub_slice() (*instagraal.cuda_lib_gl_single.sampler method*), 5
simple_start() (*instagraal.instagraal.window method*), 8
simulation (*class in instagraal.simu_single*), 12
slice_sparse_mat() (*instagraal.cuda_lib_gl_single.sampler method*), 5
sparse_data_2_gpu() (*instagraal.cuda_lib_gl_single.sampler method*), 5
sparse_data_4_gl() (*instagraal.cuda_lib_gl_single.sampler method*), 5
start_EM() (*instagraal.instagraal.window method*), 8
start_EM_all() (*instagraal.instagraal.window method*), 8
start_EM_no_scrambled() (*instagraal.instagraal.window method*), 8
start_EM_nuisance() (*instagraal.instagraal.window method*), 8
start_MCMC() (*instagraal.instagraal.window method*), 8
start_MTM() (*instagraal.instagraal.window method*), 8
step_nuisance_parameters() (*instagraal.cuda_lib_gl_single.sampler method*), 5
step_sampler() (*instagraal.cuda_lib_gl_single.sampler method*), 5
step_sampler_debug() (*instagraal.cuda_lib_gl_single.sampler method*), 5
subsample_data_set() (*in module instagraal.pyramid_sparse*), 12

T

temperature() (*instagraal.cuda_lib_gl_single.sampler method*), 5
test_copy_struct() (*instagraal.cuda_lib_gl_single.sampler method*), 5
test_model() (*instagraal.instagraal.window method*), 8
test_thrust() (*instagraal.cuda_lib_gl_single.sampler method*), 5
timer() (*instagraal.instagraal.window method*), 8

transloc() (*instagraal.cuda_lib_gl_single.sampler method*), 5

U

update_name() (*instagraal.fragment.fragment method*), 5
update_neighbourhood() (*instagraal.cuda_lib_gl_single.sampler method*), 5
update_super_index() (*instagraal.pyramid_sparse.pyramid method*), 12
update_super_index_in_dict_contig() (*instagraal.pyramid_sparse.pyramid method*), 12

V

Vec (*class in instagraal.vector*), 13

W

window (*class in instagraal.instagraal*), 7

Z

zoom_in_area() (*instagraal.pyramid_sparse.pyramid method*), 12
zoom_in_frag() (*instagraal.pyramid_sparse.pyramid method*), 12
zoom_in_pixel() (*instagraal.pyramid_sparse.pyramid method*), 12
zoom_out_frag() (*instagraal.pyramid_sparse.pyramid method*), 12
zoom_out_pixel() (*instagraal.pyramid_sparse.pyramid method*), 12