Code directory:

Dependencies:

methods1lib(custom); mclib(custom); mplib(custom);
pandas; matplotlib; numba; scipy; numpy; tqdm; scikit-image

Workflow of main (patientParamInf) script

Load cell data (Keren et al. cellData.csv)
Set parameters for: patient, depiction, MPL param inference; MCMC simulation
Load cell_center_data: pathology()
Infere parameters from image with MPL: paramInference()
Simulate with infered connectivity matrix: mcGenerator()

Parameters:

patient: int (1-41): One out of 41 original tissue samples from Karen et al. group

depiction: str; 'Detailed', 'Grained', 'General', 'CellGroup', 'ImmuneGroup': Selects how the tissue sample is depicted.
cellTypes_orig: i4[1]; Integer array of size one. Number of original cell Types

limiter = int-False: If the limiter is an integer then the computation (MPL Analysis & MCMC generation) is only done for the number of cells selected by the limiter. If the limiter is set to False, the every cell from the tissue sample is considered int the MPL optimisation/ MCMC generation.
runs: int: number MPL and MCMC runs (random independent restarts) executed.
N: int; number of epochs in the MCMC generative model
boxlength: float: length of the computation domain, i.e: the size of the lattice
This is useful for minimal image convention/ periodic boundary conditions, which are so far commented out.
TT_enc: f8[1]; float array of size 1; Monte carlo temperature for encoding (generating) a configuration.
TT_dec: f8[1]; float array of size 1; Monte carlo temperature for decoding (parameter inference) the image.

Libraries & key functions

methods1lib: Data preprocessing (.tiff ->.np); Graining depiction; rdf analysis;

Functions:

cell_centers, immuneGroup, cellTypes = pathology(cell_data,patient,depiction=depiction,
plotArr=True,plotImg=True)

Converts the pathological image data: p_%d_labeldcellData.tiff (patient) - to:
cell_centers: f8[n_cells,n_cells]; Cell positions
immuneGroup: i4[n_cells]; Cell types (indices match with cell_centers)
cellTypes: str[n_cellTypes]: Names of different cell types according to depiction

mclib: Generative Monte Carlo model & helper functions

Functions:

latticeplt(positions,types,cellTypes,T,mcStep)
Plotts the lattice of cells.
positions: f8[n_cells,n_cells]; Positions of n_cells
types: i4[n_cells]; integer 1D array of cell types
cellTypes: str[n_cellTypes]; Array of strings with different cell names
T: f8; MC temperature
mcStep: f8; Monte carlo step at which the config was made (0 = init config)

rlimit = minDistance(positions)
Computes a distance rlimit (f8) s.t. every cell has as less neighbors as possible but at least one from the positions (f8[n_cells,n_cells]).

types_inf,Energy_vec,T,mcStep =mcGenerator(Jmat_optArr,cell_center_data,rlimit,runs,N,
limiter,TT_enc,boxlength,cellTypes_orig)

Performs a MCMC optimisation for the given Hamiltonian and connectivities
Jma_optArrt[: f8n_cellTypes,n_cellTypes,runs,1]; symmetric square matrix of inter cellular connectivities.
cell_center_data:f8[x-positions,y-positions,cellType]; Concatenated array of positions and immuneGroup
rlimit: f8[1]; cellular interaction range
runs: i4; number independent optimisations for same connectivities
N: i4; monte carlo sampling epochs
limiter: i4; Limits the evaluation to a subset of cell_center_data
TT_enc: f8[1] MC encoding (driving) temperature
boxlength: f8; size of computation domain (lattice)
cellTypes_orig: i4[1]; Number of original cell types (if the limiter is small this differs from len(cellTypes)).

mpllib: Parameter Inference with maximum pseudo likelihood & helpber functions

Functions:

neighbor_register = neighbors(positions,immune_tag,rlim,boxlength)
Creates an object with info about the neighbor identity and type of a cell.

cell_centers: f8[n_cells,n_cells]; Cell positions
immune_tag: i4[n_cells]; Cell types (indices match with cell_centers)
rlim: f8[1]; cellular interaction range
boxlength: f8; size of computation domain (lattice) (obsolet since periodic boundary conditions are commented out)
neighbor_register: object[n_cells]; Properties are:
i: i4; cell identity// t: i4; cell type// ni: i4[:]; neighborhood identities// nt: i4[:]; neighborhood cell types.

mpl = mplikelihood(Jparam,n_cellTypes,neighbor_register,beta)
Evaluates the maximum pseudo likelihood for given configuration and connectivities.
Jparam: f8[n_param]; Inter cellular connectivities// n_param: number of free parameters
n_cellTypes i4; Number of cell types under consideration
neighbor register: custom object// see: neighbors()
beta: inverse mc temperature

Jmat_optArr,evalvec,rlimit,avg_neighb,n_cells= paramInference(cell_center_data,runs,limiter,
TT_dec,boxlength)
Jmat_optArr: f8[n_cellTypes,n_cellTypes,runs,1]; Infered cellular connectivities for independent number of runs.
evalvec: f8[1]; Evaluation of mplikelihood() at optimal parameters.
rlimit: f8[1]; range of cellular interactions
avg_neighb: f8[1]; average number of neighbors for given configuration neighbor_register
n_cells: f8[1]; total number of cells for given configuration

Roadmap to existing data (naming convention)

General/Graind: depiction of considered sample
patho/config: original pathological data/ simulated configuration (images)
P1/4/12: patient sample
labeledcellData.tiff: Karen et al. staining images.
cellData.csv: Karen et al. cell type analysis.
Jmat_optArr: Inferred MPL connectivities (affinities). The preallocation with 999 has to be dropped!
Energy: MCMC energy. The preallocation with 0 has to be dropped!
Pconfig/patho: probability of cell types (columns) being in the neighborhood of a reference cell type (row) of a simulated configuration or the original pathological data.