Phylodynamics Simulation of Phylogenetic Trees

Hsiang-Yu Yuan

Date: July 22, 2013

Version: 1.2

Update: 19 May 2015

Section:

Background

System Architecture

System Design

Package Descriptions

Program Flow

Background:

This document describes the system design and program flow of the new project based on the previous project entitled *critical analysis of binding avidity hypothesis*. The previous project ends in early, 2014. The files are located in ‘BindingSim’ and ‘binding\_v3’.

Package Descriptions:

There are two types of packages. One is for developing. When a special function is developed, I will create a folder for that project. For example, the “BindingAvidity/mat/Mbindingavidity/” stores the files for ‘binding avidity simulation’, and the package “InferPhylo/make\_infection\_tree/” stores the code to reconstruct the virus transmission tress.

After the functions are developed, all the functions are integrated to make the system to easily run in server or PC. Most changes for integration are to create folders for main scripts (or main functions), to create folders for output, and make all processes can be executed from a single main file.

Packages for modules developing:

* BindingAvidity/mat/Mbindingavidity: Phylodynamic simulation of viral infection using Individual-based model.
* InferPhylo/make\_infection\_tree: Inferring phylogenetic trees based on finding MRCA in each lineage.
* tradeoff: Show the viral within host fitness

Packages for release version:

* binding\_v3
  + binding\_ode: ODE to quickly calculate stationary status
  + main
    - calbinding: calculate binding avidity on the transmission tree
    - infertree: infer the transmission tree
    - job: script for job submitting on clusters
    - plotfig: plot simulated lineages
    - main\_simulation.m
  + mainout
    - ibms: output of stochastic simulation
    - ode: output of deterministic simulation
    - tree: output of transmission tree
  + make\_infection\_tree: resample and generate the transmission tree. Packaged is distributed from “InferPhylo/make\_infection\_tree/”
  + Mbindingavidity: The packaged is distributed from “BindingAvidity/mat/Mbindingavidity/”
  + tradeoff

System Architecture (for modules developing)

3 modules in maintained in 3 separate folders. Once the packages are done, copy to the integrated release version folder.

BindingSim

tradeoff

InferPhylo

BindingAvidity

make\_infection\_tree

Mbindingavidity

trees

Viruses

System produces 3 objects:

1. dat\_x\_trans\_tmp.m: to reconstruct the transmission tree
2. DataTLSIR.mat: to store SIR dynamics
3. virus\_traits.mat: to store virus traits

System Design

Mbindingavidity generate virus array

make\_infection\_tree produce the virus transmission tree first.

Using the transmission tree and virus traits, we can produce virus annotated tree.

**main\_binding\_fromeq.m**

*make\_infection\_tree*

*Mbindingavidity*

Nexus, Newick, mat

*copy*

Tree

**main\_simulate\_genealogy\_indiv\_binding**

dat\_x\_trans\_tmp.mat

virus\_traits.mat

Traits

viruses

*tau\_leap*

virus\_traits.mat

Mbinding/cal\_meanbding.m

Mean\_binding.mat

plot\_simulation.m

Program Flow for generating virus annotated tree

1. Edit parameter file params\_std\_reinfect\_larg.dat in dat/ folder (N=3x10^6)
2. Run Main\_binding\_fromeq.m
   1. Generate SIR, Traits, and Viruses.

Viruses: dat\_x\_trans\_tmp.m

SIR: DataTLSIR.mat

Traits: virus\_traits.mat

1. Copy dat\_x\_trans.tmp.mat and virus\_traits.mat to make\_infection\_tree/ folder
2. Generate virus transmission tree (n = 300)
   1. Make\_Infection\_Tree (main\_simulate\_genealogy\_indiv\_binding.m)

Input: {'births', 'deaths', 'parent'}

Output: outfile\_treeData (indiv\_genealogy\_300.tree)

Output:

indiv\_treeData

indiv\_genealogy

1. BuildTree\_indiv\_nexus.m

Create a new file BuildTree\_indiv\_nexus.m

The tree will be produced in nexus file format, which can be read by figtree.