RangeShifter

Modelling eco-evolutionary dispersal dynamics &

Investigating species' responses to environmental changes

Version 2.0 DRAFT

Systems Manual

Stephen C. F. Palmer

November 2019

Code structure

RangeShifter v2.0 (RS) comprises two executable files, each of which is compiled for both 32-bit and 64-bit processors: the batch version (*RSbatch_v2_0_NNbit.exe*) and the graphical user interface (GUI) version (*RangeShifter_v2_0_NNbit.exe*) (where *NN* is 32 or 64). Accordingly, the source code is split into three folders: *Batch* and *VCL* for code applicable respectively to the batch and GUI versions only, and *Common* for code used in both versions.

In general, the code for each defined class (Figure 1) is located within a pair of *.h* and *.cpp* files bearing the name of the class, but there are some exceptions: code for the *InitDist* class is located in the *Landscape.** files, code for *DistCell* in the *Cell.** files, code for *Chromosome* in the *Genome.** files and code for the four environment-related parameter classes (*paramGrad*, *paramStoch*, *paramInit* and *paramSim*) is in the *Parameter.** files. The *BatchMode.** files contain all code relating to the parsing and reading of input files for the batch mode (but no class definitions), and a group of five files holds the code for Agner Fog's random number generators (used for the 32-bit versions only, as they are incompatible with 64-bit processing).

Relationships between Objects

The Landscape

Although the user must specify whether a model runs on a cell-based or a patch-based landscape, internally all landscapes are patch-based; for a cell-based model, each suitable *Patch* comprises only one *Cell*.

The Community

At present, RS models only a single species, but the code has broadly speaking been designed to allow for future extension to two or more species. This is achieved by the *Community* comprising many *SubCommunity* objects, each of which occurs within a single *Patch* and in turn comprises one or more *Population* objects, one of each *Species* (and currently limited to only one).

Links between Landscape and Community structures

During processing, there are occasions when the logic of the program is driven through the landscape side of the structure (e.g. drawing a map) and occasions when it is driven through the community side of the structure (e.g. during reproduction, etc.). Therefore, each side needs to be able to reference the corresponding class(es) on the other side of the structure. However, the compiler will not allow *Class1.h* to include *Class2.h* and *vice versa*, as it creates a circularity, and therefore we cannot have, for example, a *Patch* holding a pointer to its corresponding *SubCommunity* and likewise the *SubCommunity* holding a pointer to its corresponding *Patch*. This issue is circumvented by converting some pointers to integers (e.g. the pointer to the *SubCommunity* held by a *Patch* pointer; Figure 1a). in such instances, the data type *intptr* is used, which is defined and *unsigned int* for 32-bit processing and *unsigned long long* for 64-bit processing.

General coding issues

Version control

The *Version.h* file, which is held in the *Batch* or *VCL* folder, and differs between them (and further for other specific variants, see below), contains preprocessor *#define* MACRO statements, which control how the program is compiled. Each should be defined as 1 or 0,

depending on whether or not the particular feature is required in the compiled program. The options comprise:

MACRO	Purpose
RSDEBUG	Switch on debug log files (see below)
CLUSTER	Activates code required to compile on the Maxwell cluster at University of Aberdeen
RSWIN64	Set to 1 for 64-bit processing
RSRANDOM	Set to 1 for the new RS-specific random number generation (see below)
RANDOMCHECK	Option to bypass normal processing and run a check on random number generation
BATCH	Set to 1 for batch version, otherwise 0
VCL	Set to 1 for VCL version, otherwise 0

Random number generation

For consistency with RS v1.1, the random number functions of Agner Fog is included within the code, but they are incompatible with 64-bit processing, and must not be used in a 64-bit compilation. Instead, a new class *RSrandom* is provided, which is compulsory for 64-bit versions and optional for 32-bit versions. It is based on the standard random distributions, but interface functions are provided so that random distributions can be sampled anywhere in the program in exactly the same format as in v1.1; only the *#define RSRANDOM* statement needs be altered.

The *#define RANDOMCHECK* option may be used in the batch version only to bypass normal processing and instead provide samples from five random distributions: random, Bernoulli, normal, Poisson and random integer. Parameters for the last four are provided in the input file *RandomCheck.txt*, and outputs are provided in five text files in the *Outputs* folder. See the code in *RSrandom.cpp* for the format of the input file.

Batch processing

At the start of a batch run, each of the input files is parsed and checked for invalid syntax, taking into account the general model settings applied in the *Control* file. Error messages written to the *BatchLog.txt* file are mostly constructed in the function *BatchError()* from a set of standard phrases. The batch will be processed only if there are no syntax errors found during parsing.

However, input files are not checked against each other during parsing. Further checks are therefore made when parameter settings for individual simulations are read during processing, and any invalid combinations encountered are returned as an error code reported in the *BatchNN_RS_log.csv* file and described in *Batch_error_codes.xlsx*. For example, an invalid patch number read from the patch file will result in error code 14. When any such error occurs, only the current simulation is aborted; subsequent simulations may run successfully. Note that many of the errors listed in *Batch_error_codes.xlsx* should never occur, as they were inherited from error checks made in the code of v1.0, and are now checked in the input file parsing functions.

Debugging logs

Setting the macro *RSDEBUG* to 1 will activate writing of the *DebugLog.txt* file. There are many debugging statements throughout the code which will write information to this file, but most are commented out unless required for a specific purpose. As the *DebugLog* file is not opened until the model is run, a second debugging file, *DebugGUI.txt*, is provided in the *VCL* version, to which debugging messages may be written during processing of the input forms. Note that care must be taken in the *VCL* version not to write any lines to the *DebugLog* file

during processing of the input forms, i.e. before it is opened; doing so will result in an empty file.

Running batches on the Maxwell cluster

Changes to the code required to run the batch version on the Maxwell high performance cluster are activated by setting the CLUSTER macro. When running on the cluster, the program should always be passed two arguments (see User Manual, section 3.3), but in a slightly modified form, such that the second argument is a positive integer ***n*** rather than the name of the control file. This causes the file *CONTROL**n**.txt* to be read as the control file for that particular invocation of the program, and it follows that each line in the list of jobs to be queued must be given a unique value of ***n***. Good practice directs that the batch number within the file *CONTROL**n**.txt* should also be ***n***.

Customised variants

There are currently several customised variants of RS v2 developed to address particular research questions. Each of these has its own *Batch* and *VCL* folders, and may also have a *Common* folder holding new units unique to the variant. Each variant also incorporates code from the *Common* folder of the standard project. Within the standard *Common* files, code for a particular customised variant is identified by a macro (below appearing as sub-headings) set in the *Version.h* file of the customised project.

Features provided by some of these customised variants may potentially be incorporated into the standard RS v2 once fully tested. Those considered suitable are marked ***. Most are not yet documented in the format required for the RS User Manual.

HEATMAP *

When the transfer model is SMS, an option is provided to produce a single map at the end of each replicate showing the total number of visits by dispersers to each cell during the course of the simulation. The map is produced in two formats: as a bitmap and as a raster text file suitable for input into ArcGIS.

At present, this variant has been compiled in its VCL form only, as only the GUI version of RS incorporates the Embarcadero-specific code to generate bit-maps.

[Current location of code - *C:\Ash_dieback\RS_heatmap\src*]

EVOLSMS *

The SMS parameters *DP*, *GB*, *AlphaDB* and *BetaDB* can evolve in the same way as kernel and CRW parameters. Optionally, habitat-independent per-step mortality risk can be varied along temporal linear gradients.

[Current location of code - *C:\RS_v2_EVOLSMS\src*]

SPATIALMORT

An option is provided for a stage-structured species to apply spatially variable density-independent mortality, which acts additionally to and following annual mortality as specified in the transition matrix. The additional mortality typically represents some form of hunting pressure or other environmental mortality risk.

The additional mortality is specified in two input files (which may be the same physical file) identical in dimensions to the habitats file, the first of which is applied before a specified mortality change year, and the second thereafter.

[Current location of code - *C:\Lynx\RangeShifter_v2_spatial_mort\src*]

GROUPDISP ***

This variant includes three features principally relevant to plant dispersal: an hermaphrodite breeding system, a pollen kernel and group dispersal.

Under hermaphroditic breeding, all mature individuals are treated as breeding females and as breeding males. For each 'mother', the 'father' of her offspring in the current may either be a single individual or randomly selected from all available 'fathers' for each offspring. Self-fertilisation may optionally be prohibited.

The pollen kernel option enables the 'father' of each juvenile to be selected at random from all mature individuals present within one of three strata: the local population, the immediate Moore neighbourhood and globally from anywhere on the landscape, ratios being set by two independent and constant parameters. For simplicity of coding, all local and neighbourhood 'fathers' are also included within the global set.

Individuals may disperse in groups of size $N \sim \text{Poisson}(m)$, where the mean group size m is supplied as a runtime parameter. All individuals within a group suffer the same fate, i.e. all die or all settle within the same population.

This variant has its own *Common* folder in which are located files defining the *Group* class, which inherits from the *Individual* class.

[Current location of code - *C:\CIBIO\Group_dispersal\src*]

VIRTUALECOLOGIST ***

A prototype virtual ecologist is provided, which samples patches on the landscape by one of three methods (static random, static random within specified limits, dynamic random within a specified number of rows of a northwards moving range front), samples individuals from within the sampled patches, and calculates certain landscape genetics statistics (mean no. of alleles, Wright's F-statistics, Jost's D, isolation by distance). The statistics may be based on all loci or on a sub-set of loci specified in an input file. The genomes of sampled individuals may optionally be written to an output file.

Files defining the *VirtualEcologist* class are located in the *Common* folder of the *GROUPDISP* variant.

[Current location of code - *C:\CIBIO\Group_dispersal\src*]

PEDIGREE

This provides additional optional functionality for the *GROUPDISP* variant by maintaining a pedigree of all breeding individuals in the local population for which it is established.

Files defining the *Pedigree* class are located in the *Common* folder of the *GROUPDISP* variant.

[Current location of code - *C:\CIBIO\Group_dispersal\src*]

BUTTERFLYDISP ***

Two additional features provide greater realism in modelling dispersal of butterflies and similar taxa having non-overlapping generations.

Dispersal may optionally be timed to occur during reproduction, so that a female may be mated on emergence in her natal patch, disperse to a new patch, and lay her eggs in the new patch regardless of whether there are any males present. The size of her clutch is determined by the density of individuals present in her new patch, not that in her old patch. A mated female carries with her the identity of her mate, so that her juveniles can inherit his alleles if

the model incorporates genetics. If there were no males in the female's natal patch, she can disperse unmated to a new patch (NB as yet there is no specific rule compelling her to do so; if she does not emigrate, she produces no juveniles) and mate there with any male present (whether born there or immigrant).

In a model incorporating global environmental stochasticity, the time-series of annual variations (ε) may be read from an input text file rather than generated internally, in which case the auto-correlation and amplitude parameters are redundant. The file must contain two columns, *Year* and *Epsilon*. Entries in the Year column must start at zero, increase incrementally, and continue to at least one more than the number of years in the simulation (excess rows are ignored). *Epsilon* can take any value, but should be centred on zero, and for all practical purposes should vary between limits of -3 and +3 (equivalent to the maximum amplitude which can be generated internally).

[Current location of code - *C:\RS_v2_Butterfly\src*]

SEASONAL ***

This variant enables the implementation of models incorporating variable seasonal demography. To date, it has not been compiled on its own, but only for the *PARTMIGRN* variant, for which it is a necessary precursor.

[Current location of code - *C:\Partial_migration\RangeShifter\src*]

PARTMIGRN

This variant enables the implementation of theoretical and applied models incorporating variable seasonal migration. The methods are described in a separate document *Partial_migration_in_RS_v**.docx*.

The code for this variant is mostly nested within code provided for the *SEASONAL* variant.

[Current location of code - *C:\Partial_migration\RangeShifter\src*]

RS_ABC

Certain RS parameters may be estimated by the method of approximate Bayesian computation by fitting the model to various levels of observed data.

The method is described in a separate manual *RangeShifter_v2.0_ABC_v**.docx*.

When running this variant on the Maxwell cluster (which is expected to be the normal approach, given the typically large number of samples to be run), the second run-time argument ***n*** defines the batch number directly, and a single control file *CONTROL.txt* is read by every invocation of the program (the batch number specified in that file being ignored).

[Current location of code - *C:\RS_v2_ABC\src*]

GOBYMODEL

In this variant, a population comprises individuals of two phenotypes: the default social phenotype, and an asocial phenotype which exhibits altered behaviour in terms of density-dependent fecundity, emigration and settlement. Typically, additional input parameters are set such that asocial individuals show higher fecundity at low density and lower fecundity at high density, increased emigration probability at low density and inverse density-dependent settlement probability.

[Current location of code - *C:\PROBIS\RS_v2_0_goby_model\src*]

SOCIALMODEL

This is a specific variant to re-implement as an individual-based model the social personality polymorphism model of Fogarty et al. (2011) Am. Nat., 177, 273-287. It is not envisaged that this particular model would ever be incorporated into a standard version of RS.

[Current location of code - *C:\PROBIS\RS_v2_0_social_model\src*]

ROBFITT

This is a bespoke variant written specifically for Rob Fitt's PhD model, in which genetic data are output for a set of defined patch numbers hard-coded in the file *PatchSelection.cpp*. Otherwise, additional code is similar to that for the *GROUPDISP* variant.

[Current location of code - *C:\RangeShifter_v2_0\Rob_Fitt\src*]

RS_CONTAIN ***

This is a bespoke variant currently under development for the *CONTAIN* project at Aberdeen, which aims to develop methods for the adaptive management of invasive non-native species (INNS) in Latin America. The new functionality currently comprises new kernel methods to model the dispersal of seeds by wind (the WALD kernel) and by animal vectors (the 2Dt kernel), habitat-dependent demography and a management control module to implement methods of control (culling) of the INNS. Ultimately, some or all of the new functionality may be included in a future release of the standard RS. The two kernels and habitat-dependent demography could in principle be incorporated at any time, but the methods of management control are likely to be refined during the course of the project, and should not therefore be considered for inclusion in RS until the end of the project (January 2022) at the earliest.

The methods are described in a separate manual *RS_CONTAIN_UserManual_v**.docx*.

[Current location of code - *C:\CONTAIN\RS_CONTAIN\src*]

CULLDEMO

This was a prototype for *RS_CONTAIN*, and may be discarded.

Figure 1a Structure diagram for RangeShifter v2 (part 1)

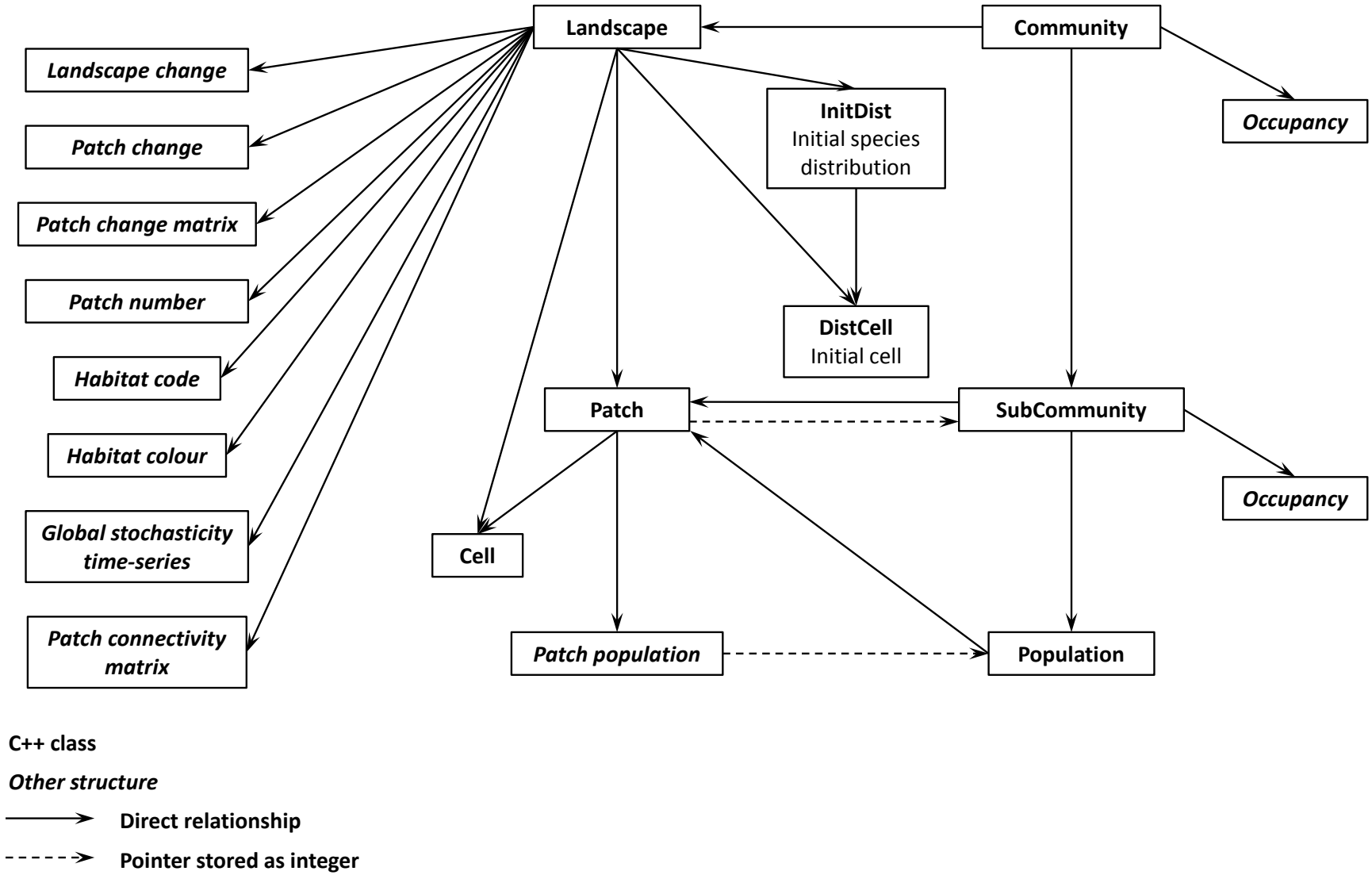

Figure 1b Structure diagram for RangeShifter v2 (part 2)