User Guide

Title: IAGS: Inferring Ancestor Genome Structure in a wide range of evolutionary scenarios

Environmental requirements

Python 3.6

Packages	Version used in Research
numpy	1.16.4
pandas	0.20.3
matplotlib	3.0.3

Gurobi solver 9.0.2 (https://www.gurobi.com/) with Academic License.

Development environment: Windows 10

Development tool: Pycharm

IAGS file format

IAGS take the GRIMM format. For example,

```
s 1 2 3 5 7 8 -12 -9 -13 14 15 16 -17 18 -19
s 20 21 22 24 26 27 28 31 32 -33
s -34 35 -36 38 41 42 45 -46 -47 48 49 50 51 52 53 132 -78
s 79 80 81 83 -85 -86 87
s 70 71 72 -115 -104 -103 -102 106 109 -108
s -111 93 95 -145 -57 -56 -55 -54 -130 -128 77 133
s 121 122 123 124 125 126 127 76 -75 118 119 120
s 62 135 -136 139 141 59 60 -142 -58 -144 -143 -134
s -166 167 168 169 170 171 113 114 -100 -101 -97 -96 146 -147 148 152 153 154 -155 156 157 158
s -89 90 91
```

Fig. 1| Example block sequence file format for IAGS. Brassica rapa block sequence.

The first item represents the chromosome type. Since the result of IAGS may produce circular genome structure, we used "s" represents a string chromosome and "c" represents a circular chromosome. The next items are synteny block order and "-" represent reverse blocks. All number are synteny block index and split by space. For some output, the block may contain bar, like "_1", "_2". For example,

```
s 514_1 562_1 565_1 572_1 518_1 515_1 534_1 -566_1 -522_1 -521_1 -534_2 -515_2 -518_2 -572_2 -565_2 566_2
s -519_1 540_1 559_1 539_1 -523_1
s 521_2 522_2
s -523_2 -533_1 531_1 -552_1 -570_1 -537_1
s 524_1 -519_2 524_2 531_2 533_2
s 537_2 570_2 -543_1 -544_1 -545_1 -546_1 -547_1 -562_2 514_2 -555_1
s 547_2 546_2 545_2 544_2 543_2 -552_2 555_2
c -559_2 -540_2 -539_2
```

Fig. 2| Example block sequence file with bar.

which used to mark blocks with multi-copy.

We allow users to build synteny blocks in different ways and encourage user to use DRIMM-Synteny (https://doi.org/10.1093/bioinformatics/btq465) to build non-overlapping synteny blocks. But the copy number of input blocks should satisfy target copy number based on whole genome duplication (no WGD block copy number is 1, one WGD block copy number is 2 and two WGD block copy number is 4).

Core functions

1. GMP model: ./model/GMPmodel.py

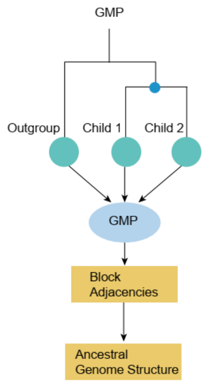

Fig. 3| GMP workflow.

GMP model takes into some species block sequence files and transforms block sequence into block adjacencies. IAGS uses GMP integer programming formulations based on these block adjacencies to get ancestral block adjacencies and then directly transforms to block sequence.

Parameters for GMP:

Parameters	**Meaning**
species_file_list	input species block sequence file list
outdir	output directory
ancestor_name	ancestor name

Example usage:

./scenarios/Brassica.py

Important output: ancestor_name.block, for example:

./outputdata/Brassica/Brassica.block

2. **GGHP model:** ./model/GGHPmodel.py

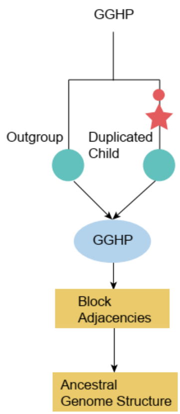

Fig. 4| GGHP workflow.

GGHP model takes into duplicated and outgroup species block sequences. Ancestor block copy number should be only one. IAGS transforms both block sequences into block adjacencies. IAGS uses GGHP integer programming formulations based on block adjacencies to get ancestral block adjacencies and then directly transforms to block sequence. For basic GGHP, target copy number of duplicated species is two and outgroup species is one. IAGS allow multiple species as input which duplicated species block sequences and outgroup species block sequences should be merged together, respectively and the input target block copy number should be summed, respectively.

Parameters for GGHP:

Parameters	**Meaning**
dup_child_file	block sequence file for duplicated species
outgroup_file	block sequence file for outgroup species
outdir	output directory
ancestor_name	ancestor name

dup_copy_number	target copy number of duplicated species
out_copy_number	target copy number of outgroup species

Example usage:

./scenarios/Yeast.py

Important output: ancestor_name.block, for example:

./outputdata/Yeast/preWGD_yeast.block

3. **Multi-copy GMP model:** ./model/MultiGMPmodel.py

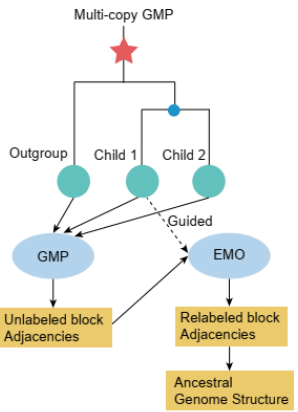

Fig. 5| Multi-copy GMP workflow.

Multi-copy GMP model takes into some species block sequence files and transforms block sequence into block adjacencies which is same with GMP. But GMP integer programming formulations can just obtain ancestral block adjacencies. Ancestral block adjacencies are multi-copy. IAGS followed child guide strategy to transform multi-copy ancestral block adjacencies to sequences using EMO integer programming formulations.

Parameters for Multi-copy GMP:

Parameters	Meaning
species_file_list	input species block sequence file list
outdir	output directory
guided_species_for_matching	a guided child species block sequence file
ancestor_name	ancestor name
ancestor_target_copy_number	target copy number of ancestor species

Example usage:

./scenarios/Gramineae.py (Ancestor 2)

Important output: ancestor_name.block, for example:

./outputdata/Gramineae/Ancestor2/Ancestor2.block

4. Multi-copy GGHP model: ./model/MultiGGHPmodel.py

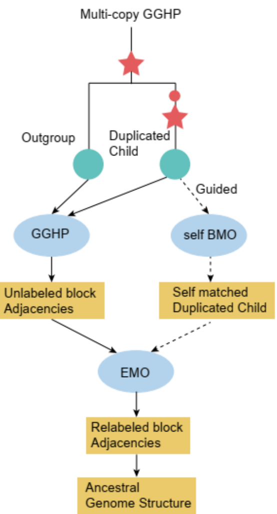

Fig. 6| Multi-copy GGHP workflow.

Multi-copy GGHP model takes into duplicated and outgroup species block sequences. Ancestor block copy number can be more than one. IAGS transforms both block sequences into block adjacencies which is same with GGHP. But GGHP integer programming formulations can just obtain ancestral block adjacencies. Ancestral block adjacencies are multi-copy. IAGS followed child guide strategy to transform multi-copy ancestral block adjacencies to sequences. IAGS first used self-BMO integer programming formulation to remove the influence of WGD in child species and then used EMO integer programming formulation.

Parameters for Multi-copy GGHP:

Parameters	**Meaning**
dup_child_file	block sequence file for duplicated species
outgroup_file	block sequence file for outgroup species
outdir	output directory
ancestor_name	ancestor name
dup_copy_number	target copy number of duplicated species
out_copy_number	target copy number of outgroup species
ancestor_target_copy_number	target copy number of ancestor species

Example usage:

./scenarios/Papaver.py (Ancestor 3)

Important output: ancestor_name.block, for example:

./outputdata/Papaver/Ancestor3/Ancestor3.block

Supporting functions

1. Evaluation of inferred ancestor: ./util/statisticsAdjacency.py

IAGS provides inferred ancestor evaluation function which contains three part. Firstly, calculating ancestral adjacencies support table. All species used for calculating this ancestor should first match with this ancestor by BMO integer programming formulations and then counting the number of block adjacencies appeared in all species, respectively. Secondly, ancestors may appear circular chromosomes. Here, IAGS allows user to cut one adjacency in circular chromosomes with minimum number of support to make circular to strings. Finally, IAGS calculates completely rearranged breakpoints ratio and obtains estimation accuracy by accuracy estimation function.

Parameters for statisticsAdjacency:

Parameters	Meaning
ancestor_file	block sequence file for inferred ancestor
ancestor_copy_number	target copy number of ancestor
ancetsor_name	ancestor name
speciesAndCopyList	all species block sequences file, target copy number and species name
outdir	output directory
model_type	model used for obtaining ancestor, including GMP, GGHP, MultiCopyGMP and MultiCopyGGHP

cutcycle	parameter for whether cut circular chromosomes
getCRBratio	parameter for whether calculated CRB ratio and estimation accuracy

Example usage:

./scenarios/Gramineae.py (Ancestor 4)

Important output:

Calculating ancestral adjacencies support table, for example:

./outputdata/Gramineae/Ancestor4/ev_Ancestor4.xls

Processing circular chromosomes, for example:

./outputdata/Gramineae/Ancestor4/Ancestor4.cutcycle.block

Calculates CRB ratio and estimation accuracy, for example:

./outputdata/Gramineae/Ancestor4/ev.txt

2. **Counting shuffling events:** ./util/calculateFissionAndFussions.py

IAGS provides downstream analysis for counting shuffling events, like fissions and fusions, which takes into two species block sequences and copy number of species 2 (ancestor) cannot larger than species 1 (descendant). If the copy number of species 1 is not equal to species 2 because of WGDs, block sequence of species 2 should be amplified to species 1. Then, IAGS used BMO matching both species and transformed to adjacencies. The adjacencies absent in species 2 are fusions and absent in species 1 are fissions.

Parameters for calculateFissionAndFussions:

Parameters	**Meaning**

species1_file	species 1 block sequence file
species2_file	species 2 block sequence file
sp1_copy_number	target copy number of species 1
sp2_copy_number	target copy number of species 2
outdir	output directory

Example usage:

./scenarios/PapaverShufflingEvents.py

Important output:

./outputdata/Papaver/shufflingEvents.txt

3. Rearrangement painting: ./util/chromosomeRearrangementPainting.py

IAGS allows output chromosomes rearrangement painting which takes into two species block sequences files. One is target species (ancestor) and the other is rearranged species (descendant). IAGS used BMO matching both species and then plots chromosomes painting.

Parameters for calculateFissionAndFussions:

Parameters	**Meaning**
block_length_file	a table recorded each block length
rearranged_species_block_file	rearranged species block sequence file
rearranged_species_name	name of rearranged species
rearranged_species_copy_number	target copy number of rearranged species

target_species_block_file	target species block sequence file
target_species_name	name of target species
target_species_copy_number	target copy number of target species
colorlist	colors for chromosomes in target species
outdir	output directory

Example usage:

./scenarios/PapaverChromosomePainting.py

Important output:

./outputdata/Papaver/plot/

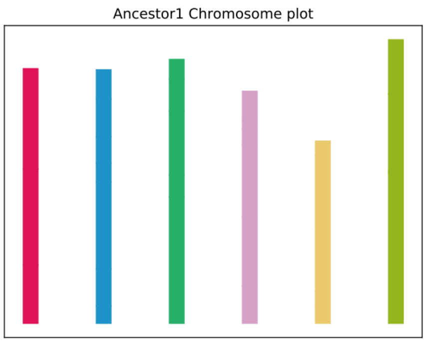

Fig. 7| Target species chromosome painting.

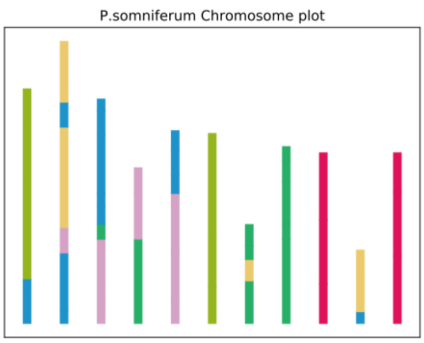

Fig. 8| Rearranged species chromosome painting.