nature protocols

https://doi.org/10.1038/s41596-024-00968-2

Protocol

Check for updates

Detection of colinear blocks and synteny and evolutionary analyses based on utilization of MCScanX

Yupeng Wang[1,2], Haibao Tang[2,3], Xiyin Wang[2,4], Ying Sun[1], Paule V. Joseph[5,6] & Andrew H. Paterson[2]

Abstract

As different taxa evolve, gene order often changes slowly enough that chromosomal 'blocks' with conserved gene orders (synteny) are discernible. The MCScanX toolkit (https://github.com/wyp1125/MCScanX) was published in 2012 as freely available software for the detection of such 'colinear blocks' and subsequent synteny and evolutionary analyses based on genome-wide gene location and protein sequence information. Owing to its simplicity and high efficiency for colinear block detection, MCScanX provides a powerful tool for conducting diverse synteny and evolutionary analyses. Moreover, the detection of colinear blocks has been embraced as an integral step for pangenome graph construction. Here, new application trends of MCScanX are explored, striving to better connect this increasingly used tool to other tools and accelerate insight generation from exponentially growing sequence data. We provide a detailed protocol that covers how to install MCScanX on diverse platforms, tune parameters, prepare input files from data from the National Center for Biotechnology Information, run MCScanX and its visualization and evolutionary analysis tools, and connect MCScanX with external tools, including MCScanX-transposed, Circos and SynVisio. This protocol is easily implemented by users with minimal computational background and is adaptable to new data of interest to them. The data and utility programs for this protocol can be obtained from http://bdx-consulting.com/mcscanx-protocol.

Key points

- During evolution, chromosomes are dynamically reorganized by duplication, inversion and translocation. Comparative analysis of genomes is empowered by identifying homologous genes that remain in their ancestral positions (colinearity). The MCScanX software package aids evolutionary studies by facilitating the detection of colinearity blocks, in particular, enabling alignments of multiple chromosomes (or segments).
- MCScanX generates easy-to-read output files and has built-in visualization tools for easy representation of evolutionary insights.

Key references

Tang, H. B. et al. *Genome Res.* **18**, 1944–1954 (2008): https://doi.org/10.1101/gr.080978.108

Wang, Y. et al. *Nucleic Acids Res.* **40**, e49 (2012): https://doi.org/10.1093/nar/gkr1293

Wang, Y. et al. *PLoS One* **11**, e0155637 (2016): https://doi.org/10.1371/journal.pone.0155637

[1]BDX Research & Consulting LLC, Herndon, VA, USA. [2]Plant Genome Mapping Laboratory, The University of Georgia, Athens, GA, USA. [3]Center for Genomics and Biotechnology, Fujian Provincial Key Laboratory of Haixia Applied Plant Systems Biology, Key Laboratory of Genetics, Breeding and Multiple Utilization of Crops, Ministry of Education, Fujian Agriculture and Forestry University, Fuzhou, China. [4]Center for Genomics, College of Science, North China University of Science and Technology, Tangshan, China. [5]Section of Sensory Science and Metabolism, National Institute on Alcohol Abuse and Alcoholism, Bethesda, MD, USA. [6]National Institute of Nursing Research, Bethesda, MD, USA. e-mail: paule.joseph@nih.gov; paterson@uga.edu

Introduction

A frequent need in comparative genomic studies is the accurate identification of homologous genes on different chromosomes (or segments) that are arranged in similar linear order (colinearity)[1–4]. The identification of these so-called 'colinear blocks'[5] on a genome scale can shed light on ancient polyploidy events[1,6,7], genomic rearrangements[8,9] and gene duplications, as well as provide inferences of gene orthology relationships[10–12]. Colinearity is a special case of the more widely used term 'synteny', which originates from Latin for 'same thread' (*syn tene*) and broadly refers to parallels in gene arrangement in divergent genomes or subgenomes.

During evolution, many genomes have been extensively shaped and restructured by whole-genome duplications (WGDs), small-scale duplications and chromosomal rearrangements[13–16]. The identification of WGDs frequently relies on discerning colinear gene pairs between large chromosomal regions and investigating the timing of these large-scale duplication events[1,2,6,7]. Synteny and colinearity analyses also facilitate the inference of localized events based on incongruity with their surroundings, such as 'illegitimate recombination', often resulting in gene conversion in which one gene copy 'overwrites' the other[17,18] or genes that have transposed from their ancestral location to a new location in a genome[8,19,20]. Indeed, small-scale gene duplications may have substantially contributed to genetic novelty during evolution[21,22].

With more and more genomes sequenced, the need for synteny and colinearity analyses has become widespread. In the past two decades, tens of software tools have been developed, many of which were limited to gene colinearity detection between only two preselected chromosomes. To accommodate the abundance of genome duplications in plants that require alignments of multiple chromosomes (or segments) from two or more genomes while also addressing increasing needs across vertebrate and invertebrate taxa, MCScanX was developed[23].

The MCScanX (https://github.com/wyp1125/MCScanX) software package included a core program named MCScanX for detection of colinear blocks and 12 utility programs for synteny visualization and evolutionary analyses. The core function of MCScanX (i.e., to efficiently identify colinear blocks in intra- and inter-species BLASTP outputs) has been most frequently implemented by the scientific community, consistent with the exponential growth of sequencing data. Synteny is often visualized by parallel plots or circle plots, in which the rectangular bars/arcs represent genomic segments, and lines/ribbons connecting a pair of genomic loci show syntenic pairs of genes. While MCScanX provides a few static visualization approaches, which are easy to use and can provide quick insights, it is also a frequent practice that MCScanX-identified colinear blocks are further processed and visualized by using external programs, which may provide interactive and/or more enriched plots, such as Circos[24], SynVisio[25] and TBTools[26]. These implementations will be illustrated below. The data and utility programs of this protocol can be downloaded from http://bdx-consulting.com/mcscanx-protocol.

Development of the protocol

MCScanX was built on a previous algorithm named 'MCScan'[2], which used a dynamic algorithm for chaining colinear gene pairs from BLASTP[27] m8 outputs, which are further processed by users (i.e., executing a utility program by 'python filter_blast.py xyz.blast.unfiltered xyz.blast'). We simplified input file preparation so that users can run the software on the direct BLASTP m8 outputs and a .gff file, which can be easily derived from the .gff files available from major databases. In addition, MCScanX removed the step of applying the Markov Cluster algorithm to restrict BLASTP-generated homologous gene pairs (i.e., "more xyz.blast | mcl - --abc –abc-neg-log -abc-tf 'mul(0.4343), ceil(200)' -o xyz.mcl"), which led to longer colinear blocks being obtained relative to MCScan.

Upon identification of colinear blocks by the MCScanX main program, further synteny visualization and evolutionary analyses can be conducted by using MCScanX utility tools or external software tools. Based on the execution of MCScanX within and between related genomes, we also developed a separate software package named 'MCScanX-transposed', which was designed for detecting genes transposed within different epochs and for

integrative analysis of gene duplication modes for a genome[20]. For input file preparation for MCScanX-transposed, researchers can refer to that with regard to MCScanX.

Comparison with other methods

Detection of colinear blocks is an essential step in inferences about gene and genome evolution, for which many tools and algorithms have been developed. MCScanX built on MCScan[2] and ColinearScan[3], and other preexisting synteny detection tools included DAGchainer[28] and SyMAP[20]. A comparison among these early colinear block detection tools can be found in Table 3 of the original MCScanX paper[23], which showed that MCScanX provided the most functionality in five aspects, including graphic visualization, multiple genomes, multi-alignments, evolutionary analyses of synteny and colinearity and analyses of gene families.

After the release of MCScanX, a few additional colinear block detection tools were published. halSynteny performs pairwise alignment blocks for any pair of genomes and identifies synteny in multiple full-genome alignments[29]. SynChro falls into the same field that reconstructs synteny blocks between pairwise comparisons of multiple genomes on the basis of a simple algorithm that computes reciprocal best hits to reconstruct the backbones of the synteny blocks and then automatically completes these blocks with non–reciprocal best hit syntenic homologs[30]. VGSC is an online service used to visualize synteny and colinearity in various graphical formats (e.g., JPEG, Bitmap, PNG, SVG, EPS and PDF) pending the upload of sequence alignment files from BLAST and colinearity relationships from other synteny analysis tools[31]. The JAX synteny browser is also a web-based synteny viewer application with preloaded human and laboratory mouse data to allow researchers to highlight or selectively display genome features according to the biological attributes of the features[32]. A detailed comparison of these post-MCScanX tools and a few early tools is shown in Supplementary Table 1. Although these later tools provide different input requirements and visualization avenues, no study has shown that the underlying algorithm for gene colinearity detection has significantly outperformed the algorithm adopted by MCScanX.

Applications of the protocol

A core functionality of MCScanX is to efficiently identify colinear blocks from BLASTP outputs, either in inter-species (comparing two or more different taxa) or intra-species (comparing 'subgenomes' within the same nucleus) scenarios, and this still represents its most efficient and effective application. The colinear block data, which are generated in a newly created and straightforward plain text file format, are often used as input files to generate further research insights such as chromosomal rearrangements, WGD events, gene duplications and genome and gene family evolution by using up to 12 downstream synteny visualization or evolutionary analysis tools already included in the software package. MCScanX has default settings (e.g., at least five genes are required to infer a colinear block, no more than 25 gaps are allowed between pairs of colinear genes and a gene with multiple BLATP hits within a five-gene window is collapsed as a single BLATP hit) that are efficient and useful for general-purpose users. For more advanced users, it offers broad parameters (described in 'Experimental design') with enough controls to explore more deeply.

For genome sequencing projects, the colinear block data are often visualized by circle or bar synteny plots to shed light on genome duplication and evolution, as well as chromosomal rearrangements. The synteny visualization and evolutionary analysis tools provided by MCScanX are used by the scientific community, though less widely than the main MCScanX program. By reviewing citation data, we found that frequently implemented MCScanX-downstream analysis tools included those for synteny visualization and classifying gene duplication modes. Moreover, the *add_ka_and_ks_to_collinearity.pl* program in MCScanX can compute evolutionary rates (including non-synonymous (Ka) and synonymous (Ks) substitution rates and their ratio, Ka/Ks) and add them to the colinear gene pairs in the colinear blocks. A separate tool named 'MCScanX-transposed' is also used to identify transposed gene duplications that occurred within different epochs.

Because MCScanX is widely accepted as a standard tool for colinear block detection, third-party downstream tools have incorporated MCScanX into their computation procedure or layered synteny and evolutionary analyses on top of MCScanX results. The TBTools software

for interactive analyses of big biological data (https://github.com/CJ-Chen/TBtools-II) has incorporated MCScanX as a tool for synteny analysis. SynVisio (https://synvisio.github.io) is an interactive multiscale synteny visualization tool for MCScanX. SynVisio is fully compatible with the data formats for MCScanX and enables researchers to explore MCScanX-identified colinear blocks through coordinated multiple views, including parallel plots at several scales, dot plots and a dynamic filter panel. Furthermore, SynVisio provides stacked parallel plots and hive plots that can visualize conservation across multiple genomes.

Today, pan-genome sequences have been deciphered in many taxa, revealing that a reference genome sequence is inadequate to represent a species or a genus. MCScanX was used as an integral step of synteny-based pan-genome analysis pipelines such as GENESPACE[33] and has already assisted many pan-genome studies such as those in *Panicum hallii*[33], *Ceratopteris richardii*[34], *Carya illinoinensis*[35], *Brassica napus*[36], *Pisum sativum*[37], and *Sorghum bicolor*[38]. This highlights the usefulness of MCScanX for pan-genome studies, which are expected to be carried out more frequently in the coming years.

Limitations of the protocol

Gene annotations and genome sequences are required to use MCScanX to detect colinear blocks and visualize synteny. This protocol will not be feasible solely on the basis of genome sequences or next-generation sequencing data. For users who want to identify colinear blocks among multiple species, although MCScanX can execute in minutes, to prepare the BLASTP input file, users need to apply BLASTP between any two genomes or within single genomes. For example, to analyze four genomes, BLASTP needs to be run 16 times (i.e., n^2). This step could significantly prolong or even abort the computation. Relative to the visualization tools (circle_plotter, dot_plotter, dual_synteny_plotter and bar_plotter) of the MCScanX package, which can configure only chromosome IDs and dimensions for the plots and can output plots only in PNG format, third-party tools may provide better-quality synteny visualization because they may provide more plotting configurations, interactive visualization, more image formats or vector images. In this case, users may take time to learn the usage of third-party tools or make manual format conversions for external tools.

Overview of the procedure

The procedure described here comprises five main parts (Fig. 1) that can be summarized as follows:

1. *Part 1: MCScanX installation (Steps 1–6)*. We describe how to install MCScanX on a Linux operation system (OS).
2. *Part 2: input file preparation (Steps 7–20)*. We describe how to download genome files and generate the .gff and .blast input files for MCscanX executions.
3. *Part 3: detection of colinear blocks (Steps 21–23)*. We describe how to correctly run MCScanX for the detection of colinear blocks.
4. *Part 4: synteny visualization (Steps 24 and 25)*. We describe how to visualize synteny based on the colinear block data generated by MCScanX, by using either MCScanX downstream tools or external tools (e.g., Circos and SynVisio).
5. *Part 5: downstream evolutionary analyses (Step 26)*. We describe how to perform various evolutionary analyses based on the colinear block data generated by MCScanX.

Data and files are often interconnected among different parts and steps. Moreover, external comparative genomic tools may contain the source code of MCScanX, require MCScanX to be pre-installed or process the output files of MCScanX.

Experimental design

Application scenarios

The procedure outlined in this protocol is designed to help users install and run MCScanX efficiently, to quickly generate research insights. MCScanX can detect colinear blocks in chromosome scaffolds or chromosomes from a single genome or multiple genomes. MCscanX can process an unlimited number of genomes of any size, depending on the capacity of the underlying computing hardware. Users may consider adjusting parameter settings as described in MCScanX parameters. In this protocol, we applied MCScanX to six plant genomes (Table 1).

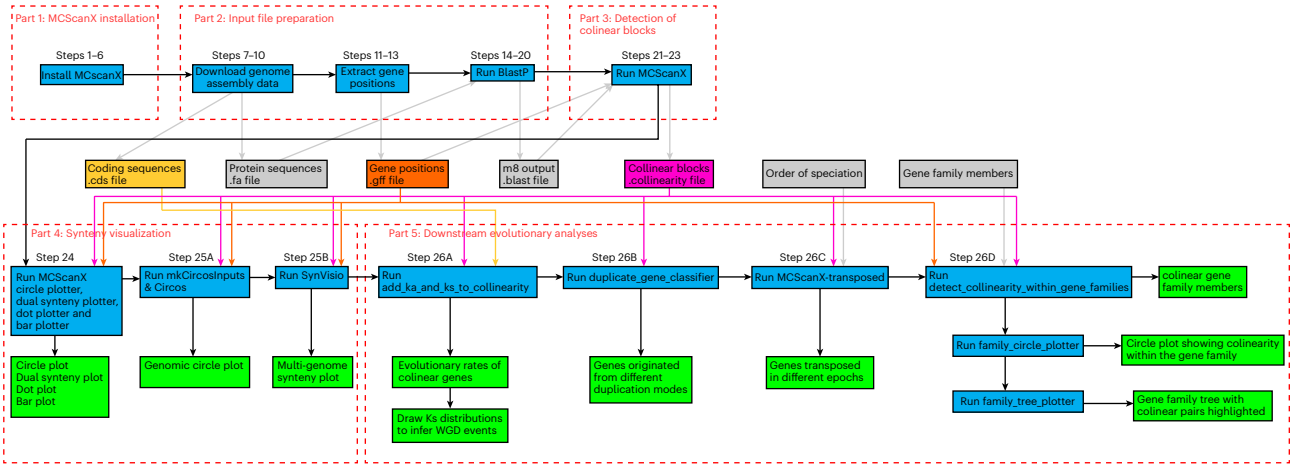

Fig. 1 | Schematic illustration of the use of MCScanX and external tools for colinear block detection, synteny visualization and evolutionary analyses. Steps are denoted in blue boxes and connected by black arrows. Green boxes denote results/insights. Boxes of other colors denote data/files, and data flow is denoted by non-black arrows. The '.collinearity' file is the key output file resulting from MCScanX, and it is further processed by various downstream actions.

Software and hardware requirements

MCScanX should be installed and run on Linux or Mac OS. Although MCScanX can be installed and run on a personal computer with a Linux virtual machine or Mac OS, because of limited CPU speed and memory, it is more suitable for users to learn the usage of MCScanX on small-scale testing data (e.g., a single *Arabidopsis thaliana* genome), included in the 'data' folder of the MCScanX package. Running MCScanX on high-performance computing (HPC) Linux clusters is more common for dealing with multiple genomes. HPC resources are available in most research institutions. Users may work with the HPC administrators to allocate system resources for running MCScanX. Cloud computing services through Amazon Web Services, Google Cloud Platform or Microsoft Azure are alternative options to run MCScanX. However, users need to have a good understanding of their billing approaches to avoid surprisingly high cloud usage charges.

Input files

The input files required for detecting colinear blocks, which are also a minimum set of input files for MCScanX, include the complete protein sequences and gene positions for each analyzed genome. We suggest that all users who plan to use MCScanX for their research first follow our steps with example data to get familiar with usage of this tool. To run the downstream analysis tools of MCScanX, more input files may be needed, depending on the specific tools. The detailed input files for running MCScanX and frequently used downstream tools are described in Table 2. Because we applied MCScanX to six genomes in this protocol, these example data have a size of ~4 GB, rendering it unreasonable to be included within the original MCScanX package.

Table 1 | Genome information used as examples in this protocol

Taxon	No. of chromosomes	GenBank assembly	NCBI Genome	Taxon abbreviation
Arabidopsis thaliana	5	GCA_000001735.2	https://www.ncbi.nlm.nih.gov/datasets/genome/GCF_000001735.4/	at
Arabidopsis suecica	13	GCA_019202805.1	https://www.ncbi.nlm.nih.gov/datasets/genome/GCA_019202805.1/	as
Arabidopsis arenosa	8	GCA_905216605.1	https://www.ncbi.nlm.nih.gov/datasets/genome/GCA_905216605.1/	aa
Thlaspi arvense	7	GCA_911865555.2	https://www.ncbi.nlm.nih.gov/datasets/genome/GCA_911865555.2/	ta
Brassica oleracea var. oleracea	9	GCA_000695525.1	https://www.ncbi.nlm.nih.gov/datasets/genome/GCF_000695525.1/	bo
Brassica carinata	17	GCA_016771965.1	https://www.ncbi.nlm.nih.gov/datasets/genome/GCA_016771965.1/	bc

Table 2 | Input files needed for running MCScanX and frequently used downstream tools

Program name(s)	Function	Files needed	Description
MCScanX	Identify colinear blocks	.blast	Direct BLASTP outputs in the m8 format; if multiple files, they should be combined
		.gff	Gene positions, following a tab-delimited format: 'sp&chr_NO gene starting_position ending_position'
duplicate_gene_classifier	Classify origins of the duplicate genes	.blast	BLASTP output from a single genome
		.gff	Gene positions, following a tab-delimited format: 'sp&chr_NO gene starting_position ending_position'
add_ka_and_ks_to_collinearity.pl	Add Ka and Ks values to gene pairs in colinear blocks	.collinearity	Output of MCScanX
		.cds	Fasta file containing the CDS of each gene
dot_plotter.java, *dual_synteny_plotter.java*, *circle_plotter.java* and *bar_plotter.java*	Visualize colinear blocks	.collinearity	Output of MCScanX
		.gff	Gene positions
		.ctl	Plot configurations

Thus, the example data and utility programs of this protocol (referred to as the 'MCScanX_protocol package') are compressed into a single ZIP file named 'MCScanX_protocol.zip', which is provided separately at http://bdx-consulting.com/mcscanx-protocol.

Preparation of BLASTP input

BLASTP should be run before running MCScanX to generate the .blast input file. To generate a proper .blast input file for MCScanX, we suggest that BLASTP is run with an E-value cutoff of 1×10^{-10} or 1×10^{-5} and the best five non-self-hits. For detection of inter-species colinear blocks, BLASTP should be run for all pairwise genomes, each time treating one genome as the reference genome and another as the query genome and keeping the top five hits. Because BLASTP may generate different results based on different reference genomes for the same pair of genomes, BLASTP should be run twice by switching the reference genome, and then both the outputs of the two runs should be kept. For detection of intra-species colinear blocks, BLASTP should be run within the analyzed genome (i.e., the analyzed genome is set as a reference as well as a query genome, with the top six hits kept). If the goal is to detect both inter- and intra-species colinear blocks, both inter- and intra-species BLASTP should be run, and the outputs should be combined as a single file.

MCScanX parameters

We use default settings that are efficient and valuable for general-purpose users throughout the procedure. However, colinear blocks generated by MCScanX can be affected by several parameters:

- The '-k' option represents MATCH_SCORE, assigned to a colinear gene pair within a colinear block. However, two consecutive pairs of colinear genes may encounter gaps (i.e., noncolinear genes) between them, which needs to be penalized.
- The '-g' option specifies a penalty score for opening a gap.
- For a colinear gene pair, its final_score is defined by MATCH_SCORE+NUM_GAPS·GAP_PENALTY. For a reported colinear block, a score is generated by summing all the final scores of its colinear gene pairs. Thus, to render a more relaxed colinear gene alignment, GAP_PENALTY can be tuned to a smaller absolute value (e.g., −0.5) than the default value (i.e., −1).
- The '-s' option, MATCH_SIZE, represents the number of genes required to 'call' (infer) a colinear block. Increasing MATCH_SIZE will reduce the number of colinear blocks detected. For small genomes, this option may be tuned to a smaller value (e.g., 4) than the default value (i.e., 5).
- The '-e' option represents the cutoff statistical threshold (E-value) for gene alignment of a colinear block. Increasing it will lead to more colinear blocks detected, whereas decreasing it will lead to fewer colinear blocks detected. A CUTOFF_score is defined by MATCH_SIZE*MATCH_SCORE. A colinear block will not be reported if its score does not exceed the CUTOFF_score.
- The '-m' option specifies the MAX_GAPS allowed for chaining the next colinear gene pair. A colinear block will stop expanding if more gaps than MAX_GAPS are encountered.

Protocol

Materials

Software

- MCScanX (https://github.com/wyp1125/MCScanX)
- MCScanX_protocol (http://bdx-consulting.com/mcscanx-protocol or https://zenodo.org/records/10023350)
- MCScanX-transposed (https://github.com/wyp1125/MCScanX-transposed)
- Circos (http://circos.ca/)
- SynVisio (https://synvisio.github.io)
- Rstudio (https://posit.co/download/rstudio-desktop/)

Requirements for installation of the MCScanX package

- Operating system: MacOS or Linux systems
  ▲ **CRITICAL** For Linux, g++ or build-essential libraries and the Java SE development kit are needed. Alternatively, a docker container image for MCScanX is available at https://hub.docker.com/r/wyp1125/mcscanx, which can avoid installation of g++ and Java compilers.
  ▲ **CRITICAL** For MacOS, Xcode (https://developer.apple.com/download/all/?q=Xcode) is needed.

Requirements for preparation of MCScanX inputs and running downstream tools

- libpng for Linux (http://www.libpng.org/pub/png/libpng.html)
- BLAST (https://ftp.ncbi.nlm.nih.gov/blast/executables/LATEST/)
- Clustalw (http://www.clustal.org/clustal2/)
- Bio-perl (https://bioperl.org/)

Hardware

- *Computer*. MCScanX can run on personal computers with ≥8 GB RAM. We have successfully tested the protocol on a Windows personal computer (AMD Ryzen 5 2500U and 8 GB RAM) with an Oracle VM VirtualBox (4 GB allocated to the virtual machine, VM) and on an iMac (3.6 GHz Quad-Core Intel Core i3 processor and 8 GB RAM). Workstations, supercomputers or cloud platforms such as Amazon Web Services, Google Cloud Platform and Azure can also be used.
  ▲ **CRITICAL** The required memory for computers is largely determined by the amount of input data.

Procedure

▲ **CRITICAL** Users need to modify the paths and folders in this protocol to refer to their working folders.
▲ **CRITICAL** This protocol details the steps for detecting colinear blocks among multiple genomes.

Part 1: MCScanX installation

● **TIMING ~10 min**
▲ **CRITICAL** This part of the procedure describes how to install MCScanX on a Linux OS, which is preferred in most usage cases. The OS used in this protocol is Linux 3.10.0-1160.71.1.el7.x86_64 and CentOS Linux release 7.8.2003 (Core). On Linux OS, g++ and java should be available before installation of MCScanX.
▲ **CRITICAL** If the user wants to install MCScanX on a Mac OS, Xcode (https://developer.apple.com/download/all/?q=Xcode) should be installed first. Then, MCScanX can be installed by following the installation procedure described below (Steps 1–6).

1. Install MCScanX by using option A if 'git' is available or option B if 'git' is not available.
   (A) **Installation of MCScanX if 'git' is available**
      (i) Clone the MCScanX repository and compile it by using these command lines:

```
$ git clone https://github.com/wyp1125/MCScanX.git
$ cd MCScanX
$ make
```

(B) **Installation of MCScanX If 'git' is not available**

(i) Run the following command lines:

```
$ wget https://github.com/wyp1125/MCScanX/archive/refs/heads/
master.zip
$ unzip master.zip
$ cd MCScanX-master
$ make
```

2. (Optional) For users who want to run the synteny visualization tools of MCScanX, install the 'libpng' library on the OS by using the following command lines (assuming an Ubuntu OS):

```
$ sudo apt-get install zlib1g-dev
$ sudo apt-get install libpng-dev
```

3. (Optional) For users who want to avoid potential compilation problems, use the MCScanX container by typing `docker pull wyp1125/mcscanx`.
4. After successful installation, check the usage of MCScanX by typing the following command:

```
$ ./MCScanX
```

Below is the screen output:

```
[Usage] ./MCScanX prefix_fn [options]-k MATCH_SCORE, final Score=MATCH_
SCORE+NUM_GAPS*GAP_PENALTY (default: 50)-g GAP_PENALTY, gap penalty
(default: -1)-s MATCH_SIZE, number of genes required to call a
collinear block (default: 5)-e E_VALUE, alignment significance
(default: 1e-05)-m MAX_GAPS, maximum gaps allowed (default: 25)-w
OVERLAP_WINDOW, maximum distance (# of genes) to collapse BLAST matches
(default: 5)-a only builds the pairwise blocks (.collinearity file)-b
patterns of collinear blocks. 0:intra- and inter-species (default);
1:intra-species; 2:inter-species(Optional)-h print this help page
```

5. (Optional) To make the MCScanX command work without specifying its absolution path, add the path of the MCScanX folder to the environment's PATH variable:

```
export PATH=/path/to/MCScanX:$PATH
```

6. (Optional) If using the docker container (Step 4), check the usage of MCScanX by typing `docker run mcscanx McscanX`.

Part 2: input file preparation

● **TIMING ~1–24 h**

▲ **CRITICAL** MCScanX reads two types of data file: .gff and .blast files. The .gff file for MCScanX is different from the GFF files downloaded from major databases such as the National Center for Biotechnology Information (NCBI) and ENSEMBL. Specifically, the .gff file for MCScanX holds gene positions in a tab-delimited format:

```
species_and_chromosome gene_id starting_position ending_position.
```

The user may view the 'at.gff' file under the 'data' folder of the MCScanX package to get a better sense of a typical .gff format by using the following command line:

```
less data/at.gff
```

Download genome assembly files

● **TIMING ~10 min**

7. Download genome assembly files, including genome, protein, coding sequence files and .gff files by following one of the three options (rsync, HTTPS or FTP) from NCBI. We suggest using the 'curl' command to download all compressed files to an '*ncbi_download*' folder. For example,

```
$curl -OJX GET
'https://api.ncbi.nlm.nih.gov/datasets/v2alpha/genome/accession/
GCF_000001735.4/download?include_annotation_type=GENOME_FASTA,GENOME_
GFF,CDS_FASTA,PROT_FASTA&filename=GCF_000001735.4.zip
' -H 'Accept: application/zip'
```

8. Open the '*ncbi_download*' folder and decompress the files one by one. An example is given by:

```
$ unzip GCF_000001735.4.zip -d at
```

9. For simplicity, rename the genome files by adding their taxon abbreviations as prefixes (i.e., '*sp_*').
10. Move the genome files of all species from the '*ncbi_download*' folder to an '*ncbi*' folder.

Generate the .gff' input file

● **TIMING ~10 min**

11. Create an '*intermediateData*' folder. Use the '*mkGFF3.pl*' program in the MCScanX_protocol package to read the '*sp_genomic.gff*' and '*sp_cds_from_genomic.fna*' files for each species (from Step 9) under the '*ncbi*' folder and generate a corresponding '*sp.gff*' file under the '*intermediateData*' folder by using the following command:

```
$perl mkGFF3.pl
```

12. If the downloaded .gff files have different formats, convert them to the .gff file compatible for MCScanX by using this command:

```
$cut -f
chromosome_column_id,gene_name_column_id,start_position_column_
id,end_position_column_id custom_gff_file | awk 'NF{print "two_letter_
species_abbreviation" $0}' > mcscanx_gff_file
```

▲ **CRITICAL STEP** GFF formats may be different among databases. This script is applicable to the NCBI database.

13. In the '*intermediateData*' folder (from Step 11), concatenate the .gff file of all species into a '*master.gff*' file by using the command line below:

```
$ cat *.gff > master.gff
```

Generate the .blast input file

● **TIMING ~1–24 h**

▲ **CRITICAL** BLAST is open source and freely available from the NCBI. If necessary, BLAST installation can be completed in 10 min by following the steps provided.

▲ **CRITICAL** We suggest directing all BLASTP outputs to a single folder, for example, the '*intermediateData*' folder in this protocol. A single run of BLASTP for two genomes may take several hours to complete. Note that for n genomes, n^2 runs of BLASTP are needed. Users may consider using the `nohup` command to make each BLASTP run in the background, in case the terminal is disconnected from the server because of inactivity. For those users that have access to HPC clusters, it might be possible to submit the BLASTP jobs to a queue system.

▲ **CRITICAL** The .blast file is the direct BLASTP output in its m8 format, with columns representing:

```
query_id subject_id per_identity aln_length mismatches gap_openings
q_start q_end s_start s_end e-value bit_score
```

The user may view the '*at.blast*' file under the '*data*' folder of the MCScanX package to get a better sense of the .blast format by using `less data/at.blast`.

14. (Optional) If not already installed, install a recent BLAST version by running the commands below:

```
$ lftp -e "cd blast/executables/LATEST; dir; quit" ftp.ncbi.nlm.nih.gov
| awk '{print $NF}' #check all available files
$ wget ftp://ftp.ncbi.nlm.nih.gov/blast/executables/LATEST/
(the_appropriate_file) #download it
$ tar -xzvf the_appropriate_file #decompress it
$ rm the_appropriate_file #remove the downloaded file
$ cd ncbi-blast-(version_for_ the_appropriate_file)
```

▲ **CRITICAL STEP** Users can add the folder of BLAST to their environment's PATH so that the BLAST commands are directly available.

15. Build the BLASTP database and place it in the '*ncbiDB*' folder. To build the BLASTP database, use this command:

```
$ ./makeblastdb -in ncbi/species1.fa -out ncbiDB/species1 -dbtype prot
```

▲ **CRITICAL STEP** The user can use `$ ./makeblastdb -help` to get a detailed description of command line arguments.

16. Execute all-against-all BLASTP running all the desired pairwise genomes with an E-value cutoff of 1×10^{-10} and the best five non-self-hits reported in each target genome. Use the following command to generate each pair of .blast files:

```
$ blastp -db ncbiDB/species1 -query ncbi/species2.fa -evalue
1e-10 -num_alignments 5 -outfmt 6 -out intermediateData/species1-2.
blast
```

▲ **CRITICAL STEP** The user may need to adjust the BLASTP parameters on the basis of genome sizes and research goals.

17. For each pair of genomes, switch the query and target genomes for a second execution.
18. Because colinear genes may exist in the same genomes, perform within a genome all-against-all BLASTP for each genome, with the best six hits being kept.
19. After BLASTP jobs are executed, check the job status every 2 h to monitor the progress. 36 runs may take several days to complete if using a single computer, whereas the job can be completed in several hours if using a queue system. Make sure that all BLASTP jobs are fully completed.
20. Concatenate the .blast files to generate a single 'master.blast' file by using this command:

```
$ cat *.blast > master.blast
```

Part 3: detecting colinear blocks by using MCScanX

● **TIMING ~10 min**

▲ **CRITICAL** We ran MCScanX with all default parameters among the six genomes (a match score of 50, gap penalty of 0.1, E-value of $1 \times 10^{0.5}$, maximum gap size between any two consecutive protein pairs of 25 and at least five consecutive proteins to define a colinear block). The user may adjust parameters as described in 'Experimental design'.

21. Before running MCScanX, place the 'master.blast' and 'master.gff' files into the same folder (such as the 'data' folder).
22. Run MCScanX by using the following command:

```
$ ./MCScanX data/master
```

▲ **CRITICAL STEP** Upon successful execution, MCScanX should generate two text files, a '.collinearity' file containing pairwise colinear blocks and a '.tandem' file listing all consecutive repeats, and an '.html' directory containing HTML files that display multiple alignments of colinear blocks against each reference chromosome. The '.collinearity' file is essential for further synteny visualization and evolutionary analyses. Users need to check whether these files have been successfully generated.

◆ **TROUBLESHOOTING**

23. (Optional) If running the MCScanX container, map the directory containing 'master.gff' and 'master.blast' to the '*/scratch*' directory inside the container, for example, '*docker run -v /directory_of_master.gff:/scratch mcscanx MCscanX /scratch/master*'.

Part 4: synteny visualization

● **TIMING ~2 h**

Visualizing synteny by using MCScanX downstream tools

● **TIMING ~1 h (~15 min to generate each plot)**

24. Visualize the colinear blocks identified by MCScanX by using the tools available under the 'downstream_analyses' folder of the MCScanX package. Perform this step by using option A to generate dual synteny plots, option B for dot plots, option C for circle plots and option D for bar plots.

(A) **Generation of dual synteny plots**

▲ **CRITICAL** The Java program *dual_synteny_plotter* generates the dual synteny plot that links all the colinear blocks between two sets of chromosomes by using straight lines.

(i) Prepare an input control file ('.ctl') containing plot size and chromosome IDs by modifying the example below:

```
550 //dimension (in pixels) of the x-axis 450 //dimension
(in pixels) of the y-axis at1,at2,at3,at4,at5 //chromosomes in
x-axis as1,as2,as3,as4,as5,as6,as7,as8,as9,as10,as11,as12,as13
//chromosomes in y-axis
```

(ii) Generate the dual synteny plot by using the above .ctl file, the .gff input file for MCScanX (from Step 13) and the '.collinearity' file generated by MCScanX by running the code line below:

```
$ java dual_synteny_plotter -g .gff -s .collinearity -c .ctl -o
_synteny.png
```

(B) **Generation of dot plots**

(i) Using the same set of input files from Step 24A(i and ii), run the Java program dot_plotter to generate a dot plot for all the colinear blocks on two sets of chromosomes by using the code line:

```
$ java dot_plotter -g.gff -s .collinearity -c .ctl -o _dot.png
```

(C) **Generation of circle plots**

▲ **CRITICAL** The Java program *circle_plotter* can be used to generate a circular plot that links all the syntenic blocks with curved lines between and within the chromosomes.

▲ **CRITICAL** Compared to the dual synteny and dot plot, the generation of a circle plot requires a different control.

(i) Generate a .ctl input file by using the details below as a reference:

```
800 //plot width and height (in pixels)
at1,at2,at3,at4,at5,aa1,aa2,aa3,aa4,aa5 //chromosomes in the circle
```

(ii) Run the following command to create the circle plot:

```
$ java circle_plotter -g .gff -s .collinearity -c .ctl -o _circle.png
```

(D) **Generation of bar plots**

(i) Using the *bar_plotter* program, generate a bar plot displaying chromosome rearrangement between reference and target chromosome sets provided by the user. Retrieve the input control file generated in Step 24A(i) and use the following command:

```
$ java bar_plotter -g .gff -s .collinearity -c .ctl -o _bar.png
```

◆ **TROUBLESHOOTING**

Visualizing synteny by using external tools

● **TIMING ~1 h (~30 min for each application)**

▲ **CRITICAL** Often, users of MCScanX use tools outside MCScanX to visualize genomic relationships or synteny among multiple genomes. Here, we describe the use of Circos (http://circos.ca) and SynVisio (https://synvisio.github.io), respectively.

25. Use the colinear blocks identified by the main MCScanX program (Step 22) to generate genomic circles by using Circos (option A) or to visualize synteny among multiple genomes by using SynVisio (option B).

(A) **Generating genomic circles by using Circos**

(i) Install Circos.

▲ **CRITICAL STEP** Installing and troubleshooting Circos is beyond this protocol. However, useful guidelines can be found at http://circos.ca/documentation/tutorials/.

(ii) Create Circos input files, by converting the '.collinearity' file generated by MCScanX (from Step 22) and the input .gff file for MCScanX (from Step 13) into '.karyotype' and '.links' files, respectively, by using the *mkCircosInputs.pl* program in the MCScanX_protocol package. Use the command line below:

```
$ perl mkCircosInputs.pl .gff .collinearity
```

(iii) To configure a Circos execution, generate a '.conf' file according to http://circos.ca/documentation/tutorials/configuration/configuration_files/. The aforementioned '.karyotype' file and '.links' file should be declared in the '.conf' file.

▲ **CRITICAL STEP** We offer two examples of config files that we used for this protocol, and these are available under the 'circosInput' folder. However, the user needs to modify the paths in these files.

(iv) Generate genomic circles by:

```
$circos -conf .conf
```

(B) **Visualizing synteny among multiple genomes by using SynVisio**

▲ **CRITICAL** SynVisio is an online interactive multiscale synteny visualization tool for MCScanX[25]. This tool developed for MCScanX can help users quickly generate

research insights. SynVisio explores the results of MCScanX and generates publication-ready images, including linear parallel plots, hive plots, stacked parallel plots and dot plots. SynVisio offers similar visualization options as those available with the MCScanX package, but more interactive. It requires the same two input files as MCScanX, plus additional, optional files to annotate the generated charts based on the users' needs.

▲ **CRITICAL** The steps that follow allow users to generate a stacked parallel plot by using SynVisio.

(i) Go to the 'upload' page of SynVisio (https://synvisio.github.io/#/upload) and upload the .collinearity file and .gff file from MCScanX (Step 22).

(ii) Go to the 'dashboard' page of SynVisio (https://synvisio.github.io/#/dashboard/bn), choose 'Multi Level Analysis' and 'Tree View', add chromosome levels and select chromosome IDs for each level.

(iii) Click on the 'GO' button to generate the plot.

Part 5: downstream evolutionary analyses

● **TIMING ~5 h**

26. Use the data generated by MCScanX for downstream evolutionary analysis. Use option A for evolutionary rate analysis for colinear genes, option B for classifying gene duplication modes, option C to detect transposed duplications and option D for colinearity analyses for gene families.

(A) **Evolutionary rate analysis for colinear genes**

● **TIMING ~1 h**

▲ **CRITICAL** Originally, the calculation of nonsynonymous (Ka) and synonymous (Ks) substitution rates was based on 'Bio::Align::DNAStatistics' module of the BioPerl package, which is outdated. Here, we provide an upgraded program named '*add_ka_and_ks_to_collinearity_Yn00.pl*', which used the 'Bio::Tools::Run::Phylo::PAML ::Yn00' module, available in the MCScanX_protocol package.

▲ **CRITICAL** Theoretically, Ka and Ks can be calculated for any files containing gene pairs in the same format as a '.collinearity' file. *add_ka_and_ks_to_collinearity_Yn00.pl* takes several minutes to several hours to finish depending on the computational system and the number of gene pairs. For example, 9,000 pairs take up to 1 h to finish. We recommend that the user makes the program run in a 'nohup' mode or submits a job to a queue system.

▲ **CRITICAL** Clustalw (http://www.clustal.org/clustal2/) and Bio-perl (https://bioperl.org/) are dependencies for (i.e., called from) the *add_ka_and_ks_to_collinearity_Yn00.pl* program.

(i) Run self-genome BLASTP and MCScanX to detect colinear blocks within each genome.

(ii) Generate a coding sequence file of the corresponding gene set in FASTA format.

(iii) Run *add_ka_and_ks_to_collinearity_Yn00.pl* on the .collinearity file for each genome. This step generates an output file that is a modified version of the .collinearity file, with each line containing a colinear gene pair and its Ka, Ks and Ka/Ks values. Perform this step by using the following code line:

```
$ perl add_Ka_and_Ks_to_collinearity_Yn00.pl -i sp.collinearity -d
sp.cds -o sp.KaKs
```

◆ **TROUBLESHOOTING**

(iv) Use Rstudio (https://posit.co/download/rstudio-desktop/) to visualize the Ks distributions for all colinear gene pairs for each genome by using the R script named 'visualize_Ks_distributions.R'.

(B) **Classifying gene duplication modes**

● **TIMING ~1 h**

▲ **CRITICAL** The *duplicate_gene_classifier* program within the MCScanX package, which incorporates the MCScanX algorithm, can be used to classify

the origins of the duplicate genes of a single genome into whole genome/segmental, tandem, proximal or dispersed duplications. As an example, we run this analysis for *A. thaliana* (at).

(i) Using self-genome BLASTP results (the '*at.blast*' file under the '*data*' folder of the MCScanX package), classify the origins of the duplicate genes and generate an output '*at.gene_type*' text file in the 'data' folder. To perform this step, use the command line below:

```
$ ./duplicate_gene_classifier data/at
```

▲ **CRITICAL STEP** The '*at.gene_type*' text file contains origin information for all the genes in the .gff file with gene type (0, 1, 2, 3 and 4 stand for a singleton, dispersed, proximal, tandem and WGD/segmental, respectively).
Moreover, a summary of the gene types for that species is printed to the screen, for example:

Type of gene	Code	Number
Singleton	0	3086
Dispersed duplication	1	7280
Proximal duplication	2	1033
Tandem duplication	3	30069
WGD or segmental duplication	4	6679

(ii) Using the output files of *duplicate_gene_classifier* of multiple genomes, investigate different patterns of gene duplication modes among different genomes via a bar plot.

(C) **Detecting transposed duplications by using MCScanX-transposed**

● **TIMING ~2 h**

▲ **CRITICAL** To illustrate the usage of *MCScanX-transposed.pl*, we apply it to the cacao (i.e., target) genome by using two other plant genomes as possible outgroups: yam and American chestnut. Whole-genome protein sequences and gene positions for *Theobroma cacao* v2.1 (cacao), *Dioscorea alata* v2.1 (yam) and *Castanea dentata* v1.1 (American chestnut) can be retrieved from Phytozome 13 (https://phytozomenext.jgi.doe.gov). MCScanX-transposed inputs can be prepared on the basis of the aforementioned MCScanX example (Steps 11–20).

(i) Prepare .gff files for the target genome (*target.blast/gff* file) and each of the outgroup genomes (*target_outgroup.blast/gff* file).

(ii) Run BLASTP within the target genome.

(iii) Run BLASTP between the target genome and each of the outgroup genomes.

(iv) Place all the .blast files and .gff files under a single folder, such as 'dir'. Generate '.pairs files' by executing *MCScanX-transposed.pl* by using the following command line:

```
$ perl MCScanX-transposed.pl -i dir -t target_species -c
outgroup1,outgroup2,…,outgroupN -x 2 -o outputDir
```

▲ **CRITICAL STEP** The target and outgroup genomes should be specified via the '-t' and '-c' (a comma should delimit outgroups) options of the command line. Multiple outgroups must be entered in the order of divergence from the target genome (most recent first).

▲ **CRITICAL STEP** Upon the successful execution of *MCScanX-transposed.pl*, different modes of gene duplication, including segmental, tandem, proximal and transposed duplications, are output as separate '.pairs' files, with each line containing one gene duplication. In transposed duplications, the first duplicated

gene is the transposed locus. If transposed duplications are classified into different epochs, the transposed duplications belonging to each epoch are also output as a separate file. The '.pairs' files have a format: 'Duplicate 1LocationDuplicate 2LocationE-value'.

(v) Run the *prep4KaKsFromTransposed.pl* program in the MCScanX_protocol package to convert the '.transposed.pairs' files generated by *MCScanX-transposed.pl* and a '.cds' file containing the coding sequence (CDS) of all genes into 'transposedDup.collinearity' and 'transposedDup.cds' files.

▲ **CRITICAL STEP** Users must modify the *prep4KaKsFromTransposed.pl* script for the path to *MCScanX-transposed.pl* outputs.

(vi) Run the *add_ka_and_ks_to_collinearity_Yn00.pl* program in the MCScanX_protocol package to calculate the evolutionary rates for each transposed duplication.

(D) **Colinearity analyses for gene families**

● **TIMING ~1 h**

▲ **CRITICAL** The MCScanX package provides several tools for analyzing gene family evolution on the basis of the colinear block results generated by MCScanX (Step 22). The *detect_collinearity_within_gene_families.pl* program outputs all colinear gene pairs among gene family members. The *family_circle_plotter* program generates a circle plot with these colinear gene pairs highlighted and places them on a gray background of whole-genome collinearity. The *family_tree_plotter* program can generate a gene family tree on which colinear gene pairs and tandem gene groups are linked with red and blue curves, respectively.

(i) Prepare a gene family file in tab-delimited format with the gene family name in the first column and gene family members in the subsequent columns.

(ii) Run the *detect_collinearity_within_gene_families.pl* tool with the '.collinearity' file (from Step 22) and the gene family file (from Step 26D(i)) as input files by using the code line below:

```
$perl detect_colinearity_within_gene_families.pl -i gene.family
-d .collinearity -o collinearity_within_gene_families
```

▲ **CRITICAL STEP** The output file contains the colinear pairs of the given gene families in tab-delimited format.

(iii) Generate a control file ('.ctl') containing the plot size and chromosome IDs. The control file can be easily made by modifying the 'family.ctl' example file shown here:
800 //plot width and height (in pixels)
at1,at2,at3,at4,at5,atC,atM //chromosomes in the circle

(iv) Run the *family_circle_plotter* program, providing the following input files: a .gff file containing all gene positions (from Step 13), a '.collinearity' file (from Step 22), a control file (from Step 26D(iii)) and a 'gene.family' file (refer to Step 26D(i)). The step can be executed by using the following code line:

```
$java family_circle_plotter -g .gff -s .collinearity -c family.
ctl -f gene.family -o gene_family_circle_plot.png
```

▲ **CRITICAL STEP** Users also have the choice to include just the chromosomes of interest in the 'family.ctl' file. This will generate a circle plot within the given chromosome set.

(v) Build a phylogenetic tree file for the gene family and save the tree into a 'family.tree' file in Newick format. The Newick trees are direct results from most online phylogenetic tools, such as Clustal Omega, and many phylogenetic software programs, such as PhyML.

(vi) To visualize colinear and tandem gene pairs on the phylogenetic tree of a gene family, run the *family_tree_plotter* program by using the 'family.tree' file

(Step 26D(v)), the '.collinearity' file (Step 22) and the '.tandem' file (Step 22) as input files by using the following code line:

```
$ java family_tree_plotter -t family.tree -s .collinearity
-d .tandem -x 400 -y 1100 -f 10 -o family_treePlot.png
```

Troubleshooting

Troubleshooting advice can be found in Table 3. In addition, the GitHub Issues page (https://github.com/wyp1125/MCScanX/issues) can be used for unforeseen troubleshooting and discussion.

Table 3 | Troubleshooting table

Step	Problem	Possible reason	Solution
22	MCScanX does not run or runs but without any output	The required inputs, .blast and .gff, are not under the same folder or are with different prefixes	Ensure that .blast and .gff are prefixed by the same name
		The BLASTP output is not in m8 format	Check the parameter setting for BLASTP with '-outfmt 6' as output format
		The .gff file is not in the correct format	Modify the *mkGFF3.pl* script as needed or use command lines to generate the correct .gff file for MCScanX
		Gene IDs are not the same between .blast and .gff files	Check gene IDs between the two input files
	MCScanX runs with only two outputs but not '.tandem'	No intra-species m8 blast result in .blast	Make sure to perform all-against-all BLASTP within each genome
24	Synteny plots (dual synteny plot, dot plot, bar plot or circle plot) cannot be generated correctly	Libpng may be missing	Install libpng by the command line: 'sudo apt install libpng-dev'
	MCScanX runs with only two outputs but not '.tandem'	No intra-species m8 blast result in .blast	Make sure to perform all-against-all BLASTP within each genome
26A	Calculation of Ka and Ks encounters running issues	The input '.cds', a coding sequence file of the corresponding gene set in fasta format, was not created correctly	Check *mkCDS.pl* or *prep4KaKsFromTransposed.pl* to ensure that the '.cds' is correctly generated
		Clustalw cannot be called	Install Clustalw and add its path to the environment's PATH variable
		Some Bioperl libraries may not be available	Install the required Bioperl libraries

Timing

Part 1, Steps 1–6, MCScanX installation: ~10 min
Part 2, Steps 7–10, download genome files: ~10 min
Part 2, Steps 11–13, generation of the GFF input file: ~10 min
Part 2, Steps 14–20, generation of the BLAST input file: ~1–24 h
Part 3, Steps 21–23, detection of colinear blocks by using MCScanX: ~10 min
Part 4, Step 24, visualization of synteny by using MCScanX downstream tools: ~1 h
Part 4, Step 25, visualization of synteny by using external tools: ~1 h
Part 5, Step 26, downstream evolutionary analyses: ~5 h

Anticipated results

We provide the genomic data of six Brassicaceae (mustard family) species in the MCScanX_protocol' package, which were obtained from the NCBI. Detailed information and links are available in Table 1. By following the steps for input file preparation and detailed commands in the '*README*' file in the MCScanX_protocol package, users can generate '*6species.gff*' and '*6species.blast*' input files for MCScanX. Using these files, MCScanX will create two new output files in the newly created folder '*6species.html*': '*6species.collinearity*' and '*6species.tandem*'. Of these two, the '*6species.collinearity*' file is the key result generated by MCScanX, because it contains data of colinear blocks with colinear gene pairs listed for each block and is used as the input file to visualize synteny by using either MCScanX downstream tools or external tools. To quickly visualize genome or chromosome-level synteny without file format conversion or file uploading, MCScanX visualization tools can be used. Here, we provide examples of dual synteny plots (Fig. 2a), dot plots (Fig. 2b), circle plots (Fig. 2c) and bar plots (Fig. 2d) between *A. thaliana* and *Arabidopsis suecica* (as) and between *A. suecica* and *Arabidopsis arenosa* (aa).

To make more enriched circle plots by Circos, users can conveniently apply the conversion program *mkCircosInputs.pl* to convert the '*6species.collinearity*' file to Circos input files. Figure 3 shows circle plots comparing *A. suecica*, *A. thaliana* and *A. arenosa*, generated by using MCScanX or Circos. The results (i.e., the connectivity patterns) match very well between the two tools, showing that the downstream tools integrated in MCScanX are as reliable and robust as the external ones.

SynVisio can interactively visualize MCScanX-generated synteny by uploading the '*6species.collinearity*' and '*6species.gff*' files to its website. As a quick visualization approach for generating comparative genomic insights, we made a phylogenetic tree of the six Brassicaceae (mustard family) species in Table 1 by using MEGA[39] and provided the colinear blocks identified among the six species (i.e., the '*6species.collinearity*' file) to SynVisio, to create a stacked parallel plot, with the arrangement of genomes corresponding to the phylogenetic tree (Fig. 4a,b). This type of synteny plot has been adopted in many publications using MCScanX.

There are various evolutionary analysis approaches based on the MCScanX algorithm or the '.collinearity' file generated by an MCScanX run. Investigating the Ks distributions of colinear genes within a genome may help to identify different large-scale gene duplication events if different peaks are observed. To illustrate this analysis, whole-genome protein sequences and gene positions for *A. thaliana* Araport11, *Gossypium raimondii* v2.1 (cotton), *Carya illinoinensis* Pawnee v1.1 (pecan), *Medicago truncatula* Mt4.0v1, *Zostera marina* v3.1 and *Musa acuminata* v1 were retrieved from Phytozome 13 (https://phytozome-next.jgi.doe.gov). By following the guidelines provided in Step 25A, plots of Ks distributions for these genomes can be made (Fig. 5).

A classification of gene duplication modes can help to understand different evolutionary forces shaping current genomes. By following the description in Step 25B, modes of gene duplication in *A. thaliana*, *A. suecica* and *A. arenosa* can be compared (Fig. 6). When transposed gene duplication is of particular interest, we can look at the cacao, yam and American chestnut genomes. The phylogenetic relationships of the three genomes are shown in Fig. 7a. By following the description in Step 25C, it is anticipated that *MCScanX-transposed.pl* identifies 2,856 WGD/segmental, 2,437 tandem, 1,622 proximal and 4,894 transposed duplications in the cacao genome, 2,148 *T. cacao* transposed duplications that occurred after *T. cacao* – *C. dentata* divergence, 2,746 that occurred between *T. cacao* – *C. dentata* and *T. cacao* – *D. alata* divergence and 4,028 after *T. cacao* – *D. alata* divergence. A comparison of the Ks (a proxy of time since duplication) distributions of the transposed duplications that occurred within these respective epochs with the phylogeny of the three genomes is shown in Fig. 7b.

To demonstrate the results of colinearity analyses for gene families, we obtained a list of published *A. thaliana* gene families from The Arabidopsis Information Resource

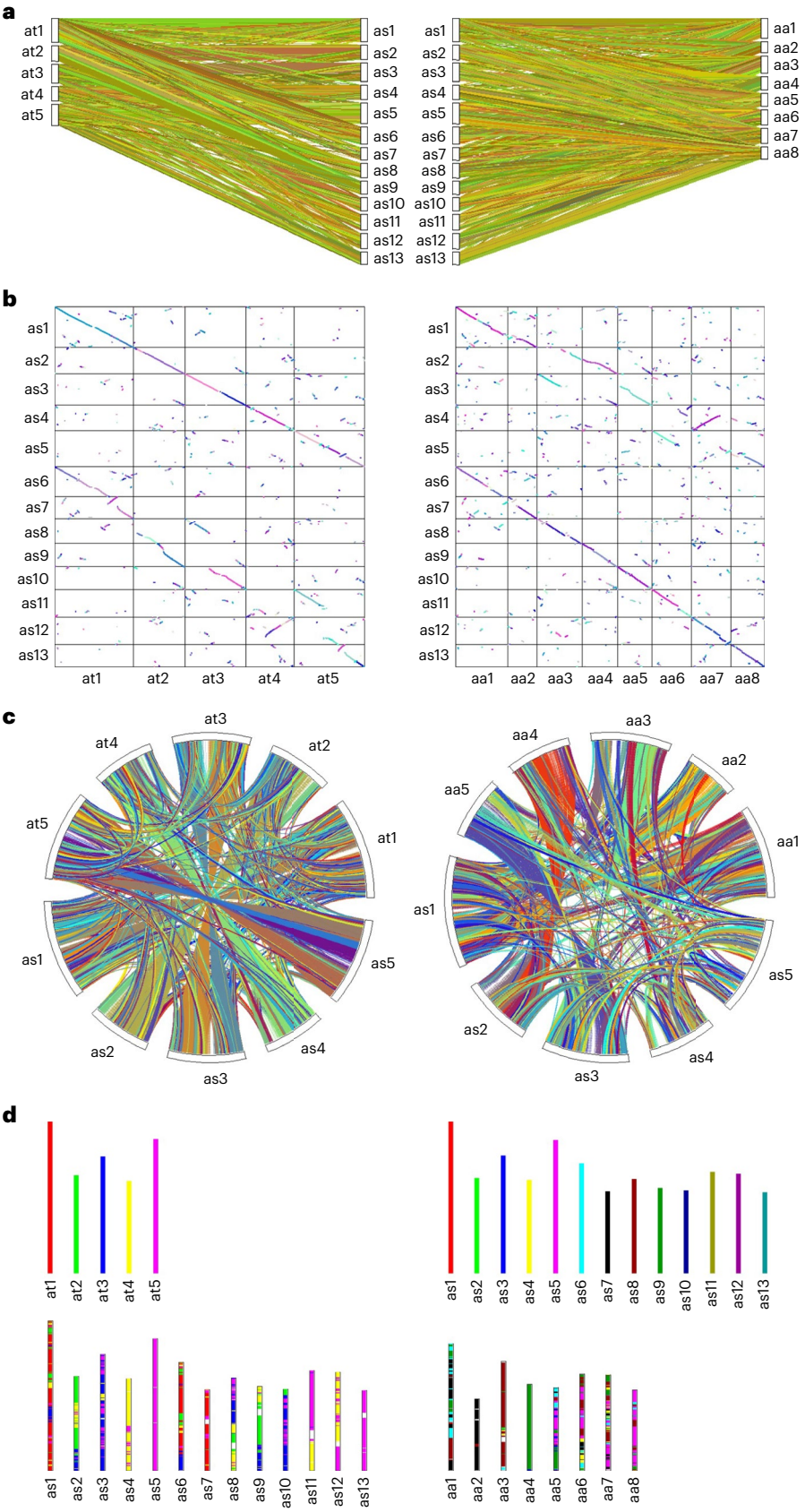
a
at1
at2
at3
at4
at5
as1
as2
as3
as4
as5
as6
as7
as8
as9
as10
as11
as12
as13
aa1
aa2
aa3
aa4
aa5
aa6
aa7
aa8
b
c
d

Fig. 2 | Synteny visualization by using downstream tools from the MCScanX package. a, Dual synteny plots generated by 'dual synteny plotter'. Users need to select two sets of chromosomes for synteny analysis, which will be displayed in two sets of vertical bars. Lines connect pairs of colinear genes between the two sets of chromosomes. **b**, Dot plots generated by 'dot plotter'. Users need to select two sets of chromosomes for synteny analysis, which will be displayed as the x-axis and y-axis of a dot plot. Dots represent pairs of colinear genes between the two sets of chromosomes. **c**, Circle plots generated by 'circle plotter'. Arcs represent chromosomes and curves connecting pairs of colinear genes. **d**, Bar plots generated by 'dot plotter'. Users need to select two sets of chromosomes. The first set of chromosomes is shown in the upper panel, where each chromosome is represented by a unique color. The second set of chromosomes is shown in the lower panel, where each chromosome contains colinear blocks denoted by the colors of the chromosomes where their paired colinear blocks are located.

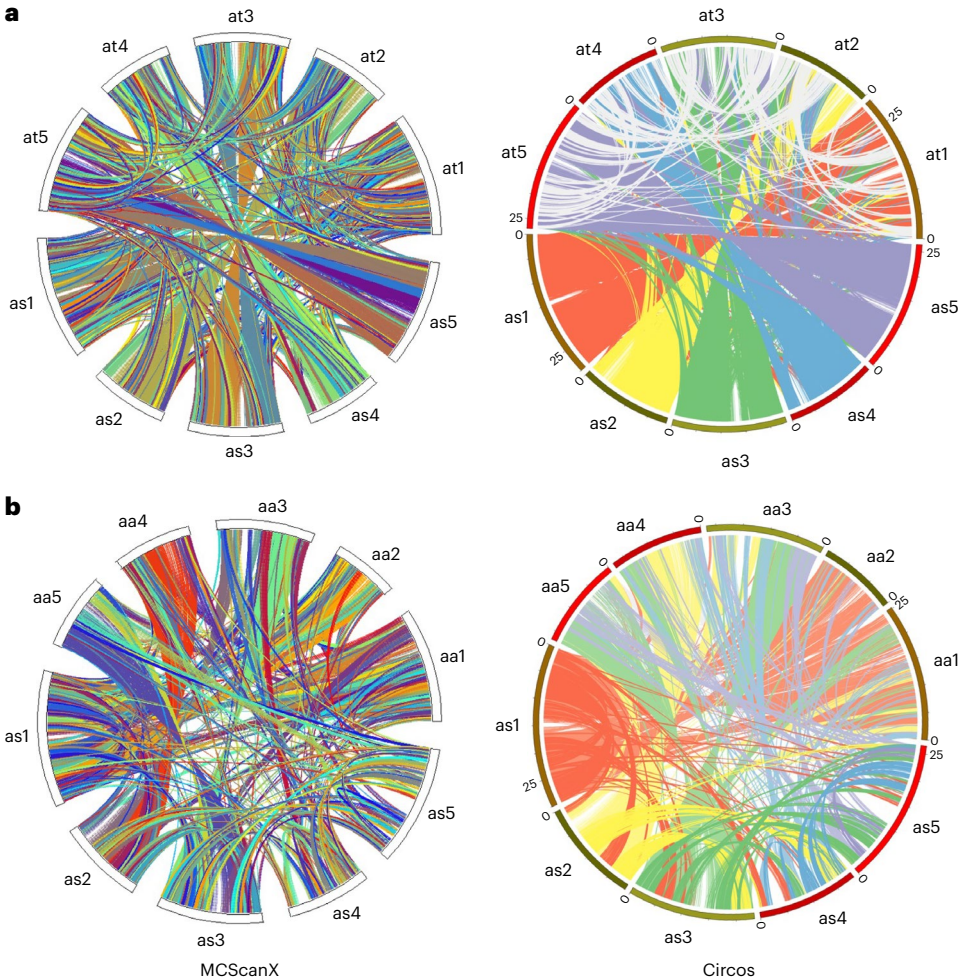

Fig. 3 | Comparison of genomic circle plots generated by circle_plotter of MCScanX and Circos. a, Genomic circle plots showing pairs of colinear genes between *A. thaliana* and *A. suecica*. **b**, Genomic circle plots showing pairs of colinear genes between *A. arenosa* and *A. suecica*. Plots in the left column were generated by using circle_plotter of MCScanX, whereas plots in the right column were generated by using Circos. Arcs represent chromosomes and curves connecting pairs of colinear genes. Chromosomes are placed in corresponding positions between the left and right columns. Although the curves are painted in different color schema, the left and right columns display very similar connectivity patterns, indicating that the circle_plotter of MCScanX and Circos can generate consistent synteny visualization. The circle_plotter of MCScanX can be used for quick synteny visualization, while interesting synteny patterns can be chosen for enriched visualization by Circos, which has the capacity to add more tracks of genomic features.

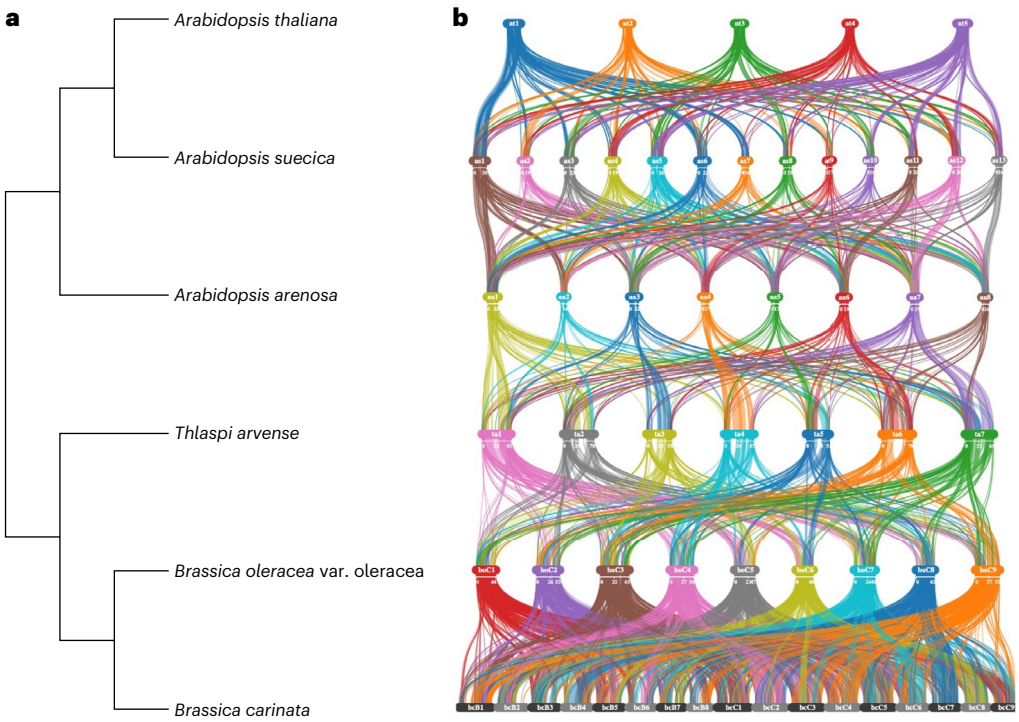

Fig. 4 | Visualizing synteny among multiple genomes by using SynVisio based on the MCScanX result. Representative example of a multi-genome synteny plot (also known as a stacked parallel plot) obtained by using SynVisio. **a**, Phylogenetic tree of the six analyzed genomes. **b**, Stacked parallel plot among the six analyzed genomes generated by SynVisio. The horizontal bars denote genomes, with individual chromosomes separated. The lines connect pairs of colinear genes. The multi-genome synteny plot, with genomes arranged according to their phylogeny, can reveal synteny conservation and chromosome evolution among related genomes, which can be further related to speciation events.

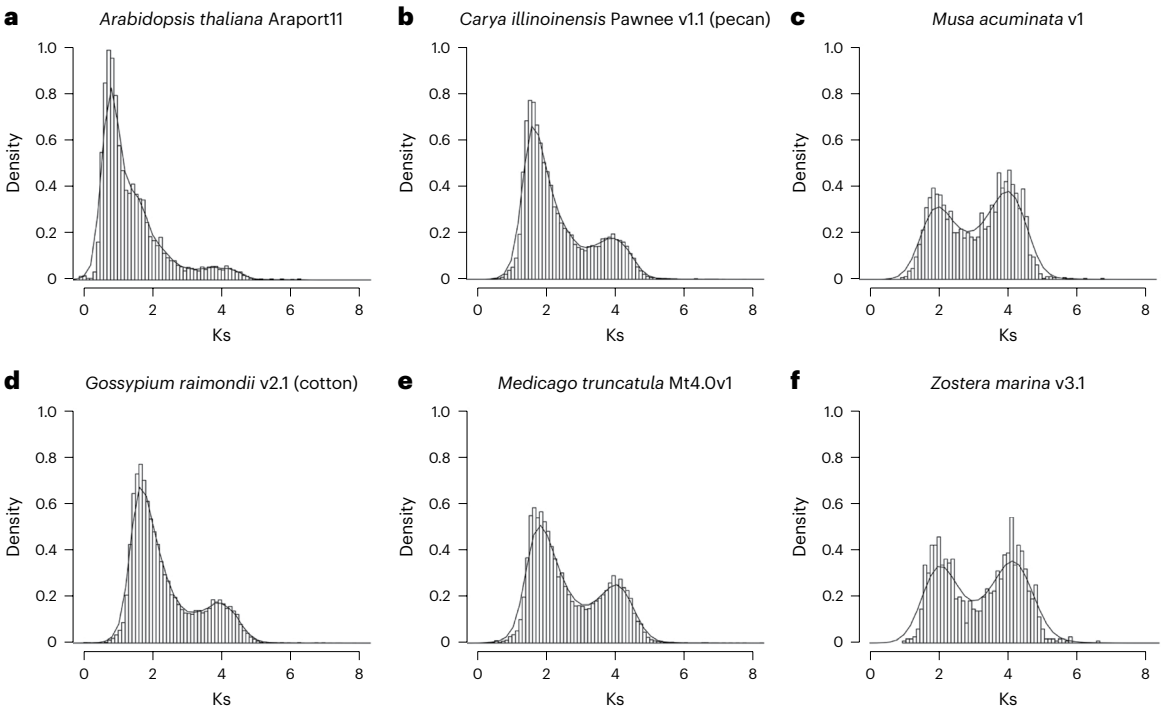

Fig. 5 | Visualization of the synonymous (Ks) substitution rate distributions of colinear genes in different genomes. Within-a-genome BLATP and MCScanX were run for each analyzed genome. The *add_Ka_and_Ks_to_collinearity_yn00.pl* program was applied to add evolutionary rates to each '.collinearity' file, and its Ks distribution was visualized by using Rstudio. Because Ks can be treated as a proxy for duplication age, peaks in Ks distributions are indicators of potential large-scale gene duplication events. In each of the six genomes, at least two peaks can be discerned, suggesting that each analyzed genome has experienced at least two large-scale gene duplication events.

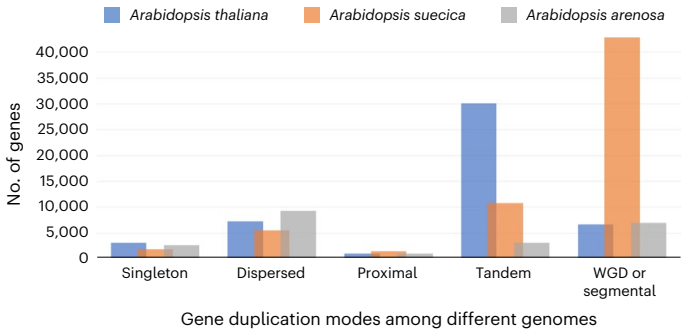

Fig. 6 | Comparison of gene duplication modes among closely related *Arabidopsis* taxa. Such analyses can reveal different evolutionary forces/events in shaping the current genomes. WGD/segmental duplications compose an overwhelming proportion of gene duplications in *A. suecica*, consistent with the fact that *A. seucica* is an allopolyploid species formed via hybridization of *A. thaliana* and *A. arenosa*. Tandem duplications are more prevalent in *A. thaliana* than in *A. arenosa*.

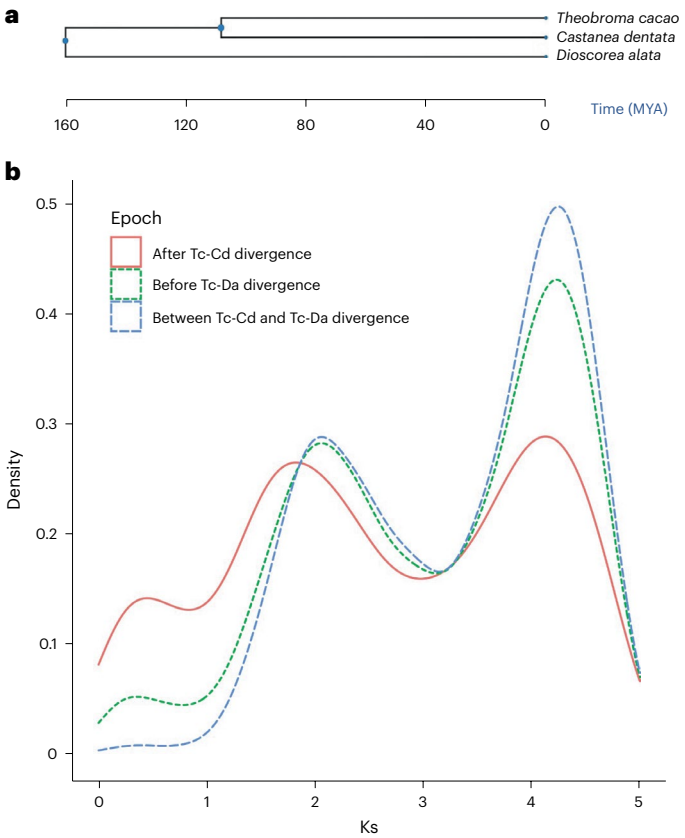

Fig. 7 | Identification of transposed duplications in the *T. cacao* genome by using MCScanX-transposed. The *D. alata* and *C. dentata* genomes were chosen as outgroups. **a**, Phylogenetic tree of the three analyzed genomes. Three epochs were derived: after *T. cacao* – *C. dentata* divergence, between *T. cacao* – *C. dentata* and *T. cacao* – *D. alata* divergence and before *T. cacao* – *D. alata* divergence. **b**, Comparison of Ks (a proxy for duplication age) distributions among transposed duplications belonging to different epochs. The plot was drawn by the 'geom_density' function of the 'ggplot' R library, where 'density' represents a kernel density estimate that is a smoothed version of the histogram. The transposed duplications after *T. cacao* – *C. dentata* divergence clearly show the lowest Ks, whereas the transposed duplications belonging to the other two epochs show higher Ks values but indistinguishable Ks distributions between each other. Comparison of Ks distributions can reveal whether Ks levels are consistent with epochs, indicating the reliability level of the identified transposed duplications. In this case, the transposed duplications after *T. cacao* – *C. dentata* divergence are more reliable. Cd, *Castanea dentata*; Da, *Dioscorea alata*; MYA, million years ago; Tc, *Theobroma cacao*.

(http://www.arabidopsis.org/browse/genefamily/index.jsp). We picked up two gene families, ABC Superfamily and class III peroxidase, as examples to illustrate the usefulness of MCScanX for analyzing the history of gene family expansion. We detected 13 and 16 colinear gene pairs from the members of the ABC Superfamily and class III peroxidase, respectively. Figure 8a shows the family circle plots for the ABC Superfamily and class III peroxidase. Phylogenetic Newick trees for the ABC Superfamily and class III peroxidases were built by using Clustal Omega, by applying all default parameters and the output format of PHYLIP (https://www.ebi.ac.uk/Tools/msa/clustalo/). Family tree plots summarize the phylogenetic tree, colinear and tandem relationships for the ABC Superfamily and class III peroxidases (Fig. 8b).

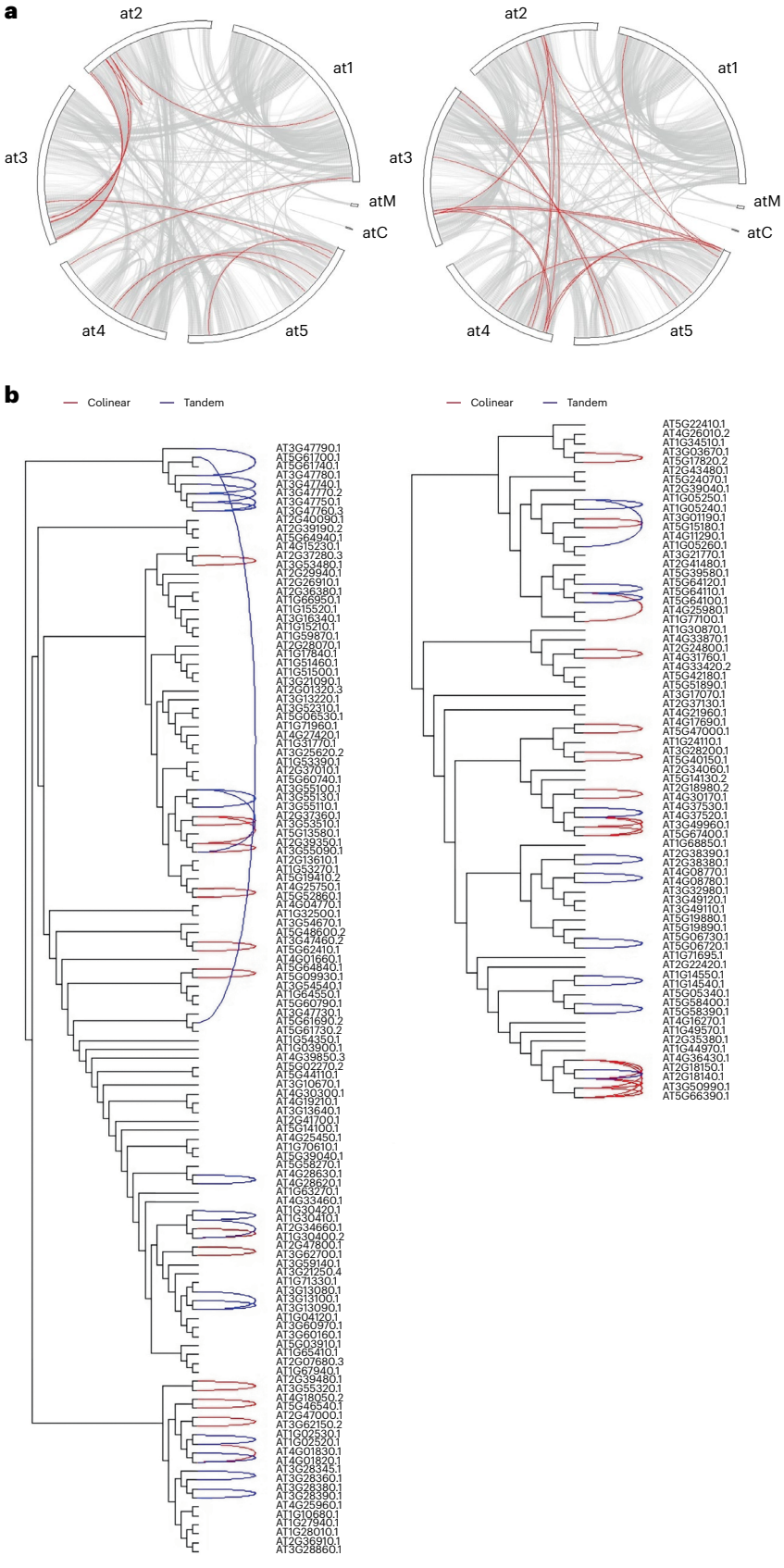

Fig. 8 | Analysis of gene family evolution by using MCScanX. Colinear and tandem gene pairs among the members of the ABC Superfamily (the left column) and class III peroxidase gene family in *A. thaliana* (the right column) were identified and highlighted by using MCScanX. **a**, Colinear relationships (red lines) among gene family members can be highlighted in a circle plot. **b**, Colinear and tandem relationships (red and blue lines, respectively) among gene family members can be highlighted in a phylogenetic tree. The plots show that there are multiple pairs of colinear and tandem genes in these two families, suggesting that both large-scale gene duplication and tandem duplication have played a role in the expansion of these two gene families.

Data availability

Users can download the reference genome sequences and gene annotations of this study from Phytozome 13 (https://phytozome-next.jgi.doe.gov) and NCBI genome (https://www.ncbi.nlm.nih.gov/datasets/genome). Small-scale testing data for MCScanX are available under the 'data' folder of the MCScanX package. The data and programs for this protocol are wrapped as a ZIP file named 'MCScanX_protocol.zip', which is available at http://bdx-consulting.com/mcscanx-protocol.

Code availability

MCScanX is available at https://github.com/wyp1125/MCScanX. MCScanX-transposed is available at https://github.com/wyp1125/MCScanX-transposed. Programs especially for this protocol are available within the 'MCScanX_protocol.zip' file, which is available at http://bdx-consulting.com/mcscanx-protocol.

Received: 21 February 2023; Accepted: 20 December 2023;
Published online: 15 March 2024

References

1. Tang, H. et al. Synteny and collinearity in plant genomes. *Science* **320**, 486–488 (2008).
2. Tang, H. B. et al. Unraveling ancient hexaploidy through multiply-aligned angiosperm gene maps. *Genome Res.* **18**, 1944–1954 (2008).
3. Wang, X. Y. et al. Statistical inference of chromosomal homology based on gene colinearity and applications to *Arabidopsis* and rice. *BMC Bioinforma.* **7**, 447 (2006).
4. Myers, P. Z. Synteny: inferring ancestral genomes. *Nat. Educ.* **1**, 47 (2008).
5. Darling, A. C., Mau, B., Blattner, F. R. & Perna, N. T. Mauve: multiple alignment of conserved genomic sequence with rearrangements. *Genome Res.* **14**, 1394–1403 (2004).
6. Bowers, J. E., Chapman, B. A., Rong, J. K. & Paterson, A. H. Unravelling angiosperm genome evolution by phylogenetic analysis of chromosomal duplication events. *Nature* **422**, 433–438 (2003).
7. Tang, H. B., Bowers, J. E., Wang, X. Y. & Paterson, A. H. Angiosperm genome comparisons reveal early polyploidy in the monocot lineage. *Proc. Natl Acad. Sci. USA* **107**, 472–477 (2010).
8. Freeling, M. et al. Many or most genes in *Arabidopsis* transposed after the origin of the order Brassicales. *Genome Res.* **18**, 1924–1937 (2008).
9. Pevzner, P. & Tesler, G. Genome rearrangements in mammalian evolution: lessons from human and mouse genomes. *Genome Res.* **13**, 37–45 (2003).
10. Jun, J., Mandoiu, I. I. & Nelson, C. E. Identification of mammalian orthologs using local synteny. *BMC Genomics* **10**, 630 (2009).
11. Tekaia, F. Inferring orthologs: open questions and perspectives. *Genomics Insights* **9**, 17–28 (2016).
12. Zheng, X. H., Lu, F., Wang, Z. Y., Hoover, J. & Mural, R. Using shared genomic synteny and shared protein functions to enhance the identification of orthologous gene pairs. *Bioinformatics* **21**, 703–710 (2005).
13. Freeling, M. Bias in plant gene content following different sorts of duplication: tandem, whole-genome, segmental, or by transposition. *Annu. Rev. f. Plant Biol.* **60**, 433–453 (2009).
14. Guo, H. et al. Gene duplication and genetic innovation in cereal genomes. *Genome Res.* **29**, 261–269 (2019).
15. Hakes, L., Pinney, J. W., Lovell, S. C., Oliver, S. G. & Robertson, D. L. All duplicates are not equal: the difference between small-scale and genome duplication. *Genome Biol.* **8**, R209 (2007).
16. Li, Z. et al. Multiple large-scale gene and genome duplications during the evolution of hexapods. *Proc. Natl Acad. Sci. USA* **115**, 4713–4718 (2018).
17. Liu, C. et al. Illegitimate recombination between homeologous genes in wheat genome. *Front. Plant Sci.* **11**, 1076 (2020).
18. Wang, X. Y., Tang, H. B., Bowers, J. E. & Paterson, A. H. Comparative inference of illegitimate recombination between rice and sorghum duplicated genes produced by polyploidization. *Genome Res.* **19**, 1026–1032 (2009).
19. Wang, Y., Ficklin, S. P., Wang, X., Feltus, F. A. & Paterson, A. H. Large-scale gene relocations following an ancient genome triplication associated with the diversification of core eudicots. *PLoS One* **11**, e0155637 (2016).
20. Wang, Y., Li, J. & Paterson, A. H. MCScanX-transposed: detecting transposed gene duplications based on multiple colinearity scans. *Bioinformatics* **29**, 1458–1460 (2013).
21. Qiao, X. et al. Gene duplication and evolution in recurring polyploidization-diploidization cycles in plants. *Genome Biol.* **20**, 38 (2019).
22. Wang, Y. P. et al. Modes of gene duplication contribute differently to genetic novelty and redundancy, but show parallels across divergent angiosperms. *Plos One* **6**, e28150 (2011).
23. Wang, Y. et al. MCScanX: a toolkit for detection and evolutionary analysis of gene synteny and collinearity. *Nucleic Acids Res.* **40**, e49 (2012).
24. Krzywinski, M. et al. Circos: an information aesthetic for comparative genomics. *Genome Res.* **19**, 1639–1645 (2009).
25. Bandi, V., Gutwin, C. Interactive exploration of genomic conservation. In *Proceedings of the 46th Graphics Interface Conference 2020* (Waterloo, 2020).
26. Chen, C. et al. TBtools: an integrative toolkit developed for interactive analyses of big biological data. *Mol. Plant* **13**, 1194–1202 (2020).
27. Altschul, S. F. et al. Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Res.* **25**, 3389–3402 (1997).
28. Haas, B. J., Delcher, A. L., Wortman, J. R. & Salzberg, S. L. DAGchainer: a tool for mining segmental genome duplications and synteny. *Bioinformatics* **20**, 3643–3646 (2004).
29. Lallemand, T., Leduc, M., Landes, C., Rizzon, C. & Lerat, E. An overview of duplicated gene detection methods: why the duplication mechanism has to be accounted for in their choice. *Genes (Basel)* **11**, 1046 (2020).
30. Drillon, G., Carbone, A. & Fischer, G. SynChro: a fast and easy tool to reconstruct and visualize synteny blocks along eukaryotic chromosomes. *PLoS One* **9**, e92621 (2014).
31. Xu, Y. et al. VGSC: a web-based vector graph toolkit of genome synteny and collinearity. *Biomed. Res. Int.* **2016**, 7823429 (2016).
32. Kolishovski, G. et al. The JAX Synteny Browser for mouse-human comparative genomics. *Mamm. Genome* **30**, 353–361 (2019).
33. Lovell, J. T. et al. The genomic landscape of molecular responses to natural drought stress in *Panicum hallii*. *Nat. Commun.* **9**, 5213 (2018).
34. Marchant, D. B. et al. Dynamic genome evolution in a model fern. *Nat. Plants* **8**, 1038–1051 (2021).
35. Lovell, J. T. et al. Four chromosome scale genomes and a pan-genome annotation to accelerate pecan tree breeding. *Nat. Commun.* **12**, 4125 (2021).
36. Song, J. M. et al. Eight high-quality genomes reveal pan-genome architecture and ecotype differentiation of *Brassica napus*. *Nat. Plants* **6**, 34–45 (2020).
37. Yang, T. et al. Improved pea reference genome and pan-genome highlight genomic features and evolutionary characteristics. *Nat. Genet.* **54**, 1553–1563 (2022).
38. Tao, Y. et al. Extensive variation within the pan-genome of cultivated and wild sorghum. *Nat. Plants* **7**, 766–773 (2021).
39. Tamura, K., Stecher, G. & Kumar, S. MEGA11: molecular evolutionary genetics analysis version 11. *Mol. Biol. Evol.* **38**, 3022–3027 (2021).

Acknowledgements

A.H.P. appreciates funding from the National Science Foundation (NSF: DBI 0849896, MCB 0821096 and MCB 1021718) and from a Regents Professorship from the University System of Georgia. H.T. is supported by funding from the National Key Research and Development Program (2021YFF1000104). X.W. is supported by funding from the China Natural Science Foundation program (32070669). P.V.J. is supported by the National Institute on Alcohol

Abuse and Alcoholism under award number Z01AA000135, the National Institute of Nursing Research and the Rockefeller University Heilbrunn Nurse Scholar Award. P.V.J. is supported by the Office of Workforce Diversity and the Office of Workforce Diversity, National Institutes of Health Distinguished Scholar Program.

Author contributions

A.H.P. conceived the project. A.H.P. and P.V.J. guided the project. X.W. and H.T. developed early versions of the genome alignment algorithms. Y.W. improved on algorithms and developed the MCScanX protocol. Y.S. performed the experiments. Y.W., P.V.J. and A.H.P. drafted the manuscript. All authors contributed to the final writing and editing of the manuscript.

Competing interests

The authors declare no competing interests.

Additional information

Supplementary information The online version contains supplementary material available at https://doi.org/10.1038/s41596-024-00968-2.

Correspondence and requests for materials should be addressed to Paule V. Joseph or Andrew H. Paterson.

Peer review information *Nature Protocols* thanks Ingo Ebersberger and the other, anonymous, reviewer(s) for their contribution to the peer review of this work.

Related links

Key references using this protocol

Tang, H. B. et al. *Genome Res.* **18**, 1944–1954 (2008): https://doi.org/10.1101/gr.080978.108
Wang, Y. et al. *Nucleic Acids Res.* **40**, e49 (2012): https://doi.org/10.1093/nar/gkr1293
Wang, Y. et al. *PLoS One* **11**, e0155637 (2016): https://doi.org/10.1371/journal.pone.0155637