README.md

MAGNET

code quality A license GPL (>= 2)

Shell script pipeline for inferring ML gene trees for many loci (e.g. genomic RAD loci)

NOTICE

> MAGNET is currently only being actively maintained and developed within **PIrANHA**, where it is at **v1.2.0**. The present repository will be periodically updated to include a standalone version of the MAGNET software. This helps users interested *only* in MAGNET; however, it is strongly recommended that you install the full PIrANHA distribution and request MAGNET updates or bug fixes through the **PIrANHA repo** (e.g. **contact the author**, or raise an issue **here**).

LICENSE

CITATION

If you use scripts from this repository as part of your published research, then I require you to cite the PIrANHA repository and/or MAGNET package as follows (also see DOI information below):

- Bagley, J.C. 2020. PIrANHA v0.4a4. GitHub repository, Available at: http://github.com/justincbagley/PIrANHA/.
- Bagley, J.C. 2020. MAGNET v1.2.0. GitHub package, Available at: http://github.com/justincbagley/MAGNET.

Alternatively, please provide the following link to this software program in your manuscript:

http://github.com/justincbagley/MAGNET

Example citations using the above URL: "We estimated a gene tree for each RAD locus in RAxML v8 (Stamatakis 2014) using the MAGNET v1.2.0 pipeline (http://github.com/justincbagley/MAGNET). Each RAxML run specified the GTRGAMMA model and coestimated the maximum-likelihood phylogeny and bootstrap proportions from 500 bootstrap pseudoreplicates."

DOI

The DOI for MAGNET, via Zenodo, is as follows: DOI 10.5281/zenodo.399054. Here is an example of citing MAGNET using the DOI:

- Bagley, J.C. 2020. MAGNET v1.2.0. GitHub package, Available at: https://doi.org/10.5281/zenodo.596774.

INTRODUCTION

The estimation of species-level phylogenies, or "species trees" is a fundamental goal in evolutionary biology. However, while "gene trees" estimated from different loci provide insight into the varying evolutionary histories of different parts of the genome, gene trees are random realizations of a stochastic evolutionary process. Thus, gene trees often exhibit conflicting topologies, being incongruent with each other and incongruent with the underlying species tree due to a variety of genetic and biological processes (e.g. gene flow, incomplete lineage sorting, introgression, selection).

With the advent of recent advances in DNA sequencing technologies, biologists now commonly sequence data from multiple loci, and even hundreds to thousands of nuclear loci can quickly be sequenced using massively parallel sequencing on NGS sequencing platforms. Full-likelihood or Bayesian algorithms for inferring species trees and population-level parameters from multiple loci, such as *BEAST and SNAPP, are computationally burdensome and may be difficult to apply to large amounts of data or distantly related taxa (or other cases that complicate obtaining MCMC convergence). By contrast, a number of recently developed and widely used "summary-statistics" approaches rely on sets of gene trees to infer a species tree for a set of taxa (reviewed by Chifman and Kubatko, 2014; Mirarab and Warnow, 2015). These methods are specifically designed to estimate gene trees or use gene trees input by the user, which are treated as observed data points analyzed in a distance-based or coalescent algorithm. Moreover, summary-statistics approaches to species tree inference tend to be accurate and typically much faster than full-data approaches (e.g. Mirarab et al., 2014; Chifman and Kubatko, 2014). Examples of species tree software in this category include programs such as BUCKy (Larget et al., 2010), STEM (Liu et al., 2010), spedeSTEM, NJst (Liu and Yu, 2011), ASTRAL and ASTRAL-II (Mirarab and Warnow, 2015), and ASTRID (Vachaspati and Warnow, 2015). Phylogenetic network models implemented in recent software like SplitsTree and SNaQ also improve network and inference by analyzing sets of gene trees.

Despite the importance of gene trees in species tree and network inference, few resources have been specifically designed to aid rapid estimation of gene trees for different loci. MAGNET (MAny GeNE Trees) is a shell script pipeline within the PIrANHA (PhylogenetIcs ANd PHylogeogrAphy) repository (https://github.com/justincbagley/PIrANHA) that helps fill this gap by automating inferring a maximum-likelihood (ML) gene tree for each locus in a multilocus dataset. Here, the term "locus" is used loosely to refer to a DNA alignment of homologous nucleotide characters including both variable and invariant DNA sites. The MAGNET package was originally coded up to aid analyses of RAD loci generated by massively parallel sequencing of ddRAD-seq genomic libraries (Peterson et al. 2012). However, MAGNET can be used to estimate gene trees from any multilocus dataset in the appropriate format, and three starting file types are supported: single NEXUS, single G-PhoCS, or multiple PHYLIP files.

HARDWARE AND SETUP

💻 MAGNET focuses on allowing users to automate the workflow necessary for quickly estimating many gene trees for many loci on their local machine.

👍 No special hardware or setup is necessary, unless the user is interested in estimating gene trees on a remote supercomputing cluster (see below).

SOFTWARE DEPENDENCIES

MAGNET v1.2.0 is a software package composed of `shell`, `R`, and Perl scripts and also calls several software programs that it relies on as dependencies. These dependencies are described in some detail in README files for different scripts in the package. However, here I provide a list of them, with asterisks preceding those already included with the MAGNET distribution:

- bioscripts.convert v0.4 Python package (available at: https://pypi.python.org/pypi/bioscripts.convert/0.4; also see README for `NEXUS2gphocs.sh`).
- RAxML, installed and running on local machine (available at: http://sco.h-its.org/exelixis/web/software/raxml/index.html).

Users must install all software not included in MAGNET, and ensure that it is available via the command line on their local machine. On the user's local machine, Perl should be available by simply typing "Perl" at the command line; Python should be available by typing "python" at the command line; and bioscripts.convert package should be available by typing "convbioseq" at the command line. Also, RAxML should be compiled using SSE3 install commands, so that RAxML can be called by simply typing "raxmlHPC-SSE3" on the command line. For detailed instructions for setting up RAxML this way, refer to the newest RAxML user manual (available at: http://sco.h-its.org/exelixis/resource/download/NewManual.pdf).

INPUT FILE FORMAT

MAGNET assumes that you are starting from multilocus DNA sequence data in one of three formats. The *First format* that is supported is that of a single datafile in G-Phocs (Gronau et al. 2011) format, with the extension ".gphocs". The *Second format* that is supported is NEXUS format, with data in a single file having the extension ".nex". For genomic data in aligned sequence format, such as aligned RAD tags (e.g. ddRAD-seq contigs) or other SNP data derived from genotyping-by-sequencing (GBS) methods, the user should assemble the data, call SNPs, and output SNP sequence data files in .gphocs or .nex format prior to running MAGNET. This can easily be done by running pyRAD or ipyrad (Eaton 2014) while calling for output in all formats (*; you'll get .gphocs and .nex files).

However, this may not always be possible, and .gphocs format is not yet among the most popular file formats in phylogenomics/population genomics. Thus, I have added a `NEXUS2gphocs.sh` shell script utility within MAGNET (in the "shell" folder) that will convert a sequential NEXUS file into .gphocs format for you. An example NEXUS file "example.nex" is included in the distribution. Feel free to use the `NEXUS2gphocs.sh` utility script independently of MAGNET to convert from NEXUS to .gphocs format. However, when doing this, *make sure to follow the usage guidelines below*.

The *Third format* that is supported in MAGNET is that of DNA sequence alignments for multiple loci contained in separate PHYLIP files for each locus.

Users must specify the input fileType with the `-f` flag. Options are 1 for a single G-PhoCS- or NEXUS-formattted , or 2 for the multiple PHYLIP option. If `-f 1`, then the program will expect as standard input (stdin) the name of the . However, if `-f 2`, then you should supply MAGNET with the path to the desired working directory; often, this will simply be the current working directory, in which case the user can simply type "." for , but the relative or absolute path to the will also work fine.

PIPELINE

Apart from input file conversion steps, the MAGNET pipeline works by calling five different scripts, in series, each designed to conduct a task whose output is processed in the next step of the pipeline. First, the `gphocs2multiPhylip.sh` shell script is used to extract loci from the input file and place each locus in a PHYLIP-formatted file with extension ".phy". Second, a shell script named `MultiRAxMLPrepper.sh` is used to place the .phy files into separate "run folders" and prepare them to be run in RAxML. Third, a script named `RAxMLRunner.sh` is called to run RAxML on the contents of each run folder. In a "clean-up" step, MAGNET moves all .phy files files remaining in the working directory to a new folder, "phylip_files", which is created within the working directory.

After running the MAGNET pipeline, the shell script `getGeneTrees.sh` automates post-processing of the gene trees output by RAxML, including organizing all inferred gene trees into a single "gene_trees" folder in the working directory, and combining the individual 'best' gene trees resulting from each run into a single file named "besttrees.tre". Also, if bootstrap pseudoreplicates were performed and the bootstrap tree files are detected, then the `getBootTrees.sh` script conducts similar processing on the bootstrap trees for each loucus, which are collated, renamed, and given a list file containing the name of each file. Given the directory of bootstrap trees resulting from a MAGNET run ("bootstrap_trees") can take up substantial disk space (>200 MB), users may wish to compress this directory to a zip file, for example using `$ zip -r bootstrap_trees.zip bootstrap_trees/` at the conclusion of a run.

A new feature of MAGNET (as of December 2018) is the `-r|--resume` flag, a long option allowing the user to resume a previous MAGNET run in a working directory where MAGNET was previously run (specified to stdin as). For example, do a "resume MAGNET" run by calling the program with `./MAGNET.sh [options] -r 1` or `./MAGNET.sh [options] --resume 1`.

USAGE

Additional input file and usage information is available in the usage or help texts. To get regular usage info for MAGNET, type `$ ./MAGNET.sh`, `$ ./MAGNET.sh -h .`, or `./MAGNET.sh -help` while in the MAGNET directory. However, it is more useful (particularly when running for the first time) to get *verbose usage info* for MAGNET, including detailed descriptions of each option; do this by typing `$ ./MAGNET.sh -H .` or `./MAGNET.sh -Help` (capital "H" flag) at the command line while in the MAGNET directory. The verbose usage text is as follows:

```
$ ./MAGNET.sh -H

Usage: $(basename "$0") [OPTION]...

 ${bold}Options:${reset}
  -f, --filetype     fileType (def: 1; also: 2) starting file type; if 1, script expects as
                     stdin a single input NEXUS file in the current directory; if 2, then
                     script expects multiple input PHYLIP files in current directory
  -i, --input        inputNEXUS (def: NULL) input NEXUS file (mandatory for -f 1)
  -e, --exec         executable (def: $MY_RAXML_EXECUTABLE) name of RAxML executable available
                     from user's command line interface
  -b, --boot         numBootstraps (def: $MY_NUM_BOOTREPS) RAxML bootstrap pseudoreplicates
  -r, --raxmlmodel   raxmlModel (def: $MY_RAXML_MODEL; other: GTRGAMMAI, GTRCAT, GTRCATI)
  -s, --simplemodel  simpleModel (def: $MY_SIMPLE_MODEL; other: JC69, K80, HKY85) specifies
                     simple DNA substitution model that will override any other model (even
                     across partitions)
  -g, --gapthresh    gapThreshold (def: $MY_GAP_THRESHOLD=essentially zero gaps allowed unless
                     >1000 individuals; takes float proportion value) gap threshold value
  -m, --missing      indivMissingData (def: $MY_INDIV_MISSING_DATA=allowed; 0=removed) missing
                     data setting
  -o, --outgroup     outgroup (def: NULL) outgroup given as single taxon name (tip label) or
                     comma-separted list
  -h, --help         echo this help text and exit
  -H, --Help         echo verbose help text and exit
  -V, --version      echo version and exit
  -R, --resume       resume (def: 0, off; 1, on) option allowing user to resume a previous
                     MAGNET run in the current working directory
  -d, --debug        debug (def: 0, off; 1, on) run function in Bash debug mode

 ${bold}OVERVIEW${reset}
 The goal of MAGNET is to infer a maximum-likelihood (ML) gene tree in RAxML for each of
 multiple loci, starting from one or multiple DNA sequence alignment input files. If supplied
 with a single G-PhoCS ('*.gphocs') or NEXUS ('*.nex') data file (using -f1 or -i <inputNEXUS>
 -f1 options), then each locus is split into a separate PHYLIP alignment file, and RAxML
 (Stamatakis 2014) is run to infer gene trees for each locus. If a NEXUS datafile is supplied,
 it is converted into G-PhoCS format (Gronau et al. 2011) while splitting loci into separate
 interleaved sequence blocks based on information provided in a sets block at the end of the
 NEXUS file (e.g. defined using 'charset' commands), which is mandatory. However, if -f2, then
 the program will run in current directory, assuming it contains multiple PHYLIP-formatted
 alignment files. Under this scenario, MAGNET will skip directly to running the PHYLIP files
 in RAxML using user-specified options.
        Sequence names may not include hyphen characters, or there could be issues. For detailed
 information on MAGNET and its various dependencies, see 'README.md' file in the distribution
 folder; however, it is key that dependencies are available from the command line interface.
 Among the most important options is <resume> (-r|--resume, off by default), which tells the
 program to resume a previous MAGNET run in current directory, including detecting incomplete
 RAxML run folders, and running RAxML without overwriting results from the previous run(s).

 ${bold}DETAILS${reset}
 The -f flag (also --filetype) specifies the starting fileType. If -f 1, then the mandatory
        input is the name or path to the corresponding <inputNEXUS> starting file, which is
        passed using the -i|--input flag. If -f 2, then mandatory input is the name or path to
        the working directory (type '.' for current directory, or supply a relative or absolute
        path).

 The -i flag (also --input) passess the name of the input NEXUS file, <inputNEXUS> parameter,
    to the program.

 The -e flag (also --exec) sets the name of the RAxML executable that will be called. The
    default executable name is 'raxml', but the user may wish to change this to something
    specific to their install or parallelization needs (e.g. 'raxmlHPC-PTHREADS-SSE3'). The
    default setting should work on local machine or supercomputing cluster installs. However,
    this should be tested beforehand by entering 'raxml' at the command prompt. On some
    version fo Linux this yields the following error message:

           'raxml: error while loading shared libraries: libmpi.so.12: cannot open shared object
            file: No such file or directory'.

        If this occurs, then Open MPI related libraries are installed in a non-standard location
        and you will need to add this location to your LD_LIBRARY_PATH, e.g.:

        'export LD_LIBRARY_PATH=/usr/local/openmpi-1.8.1/lib:$LD_LIBRARY_PATH'

        See the following URL: for more insight into this problem: https://stackoverflow.com/
    questions/14769599/mpi-error-loading-shared-libraries. However, simply using a different
    raxml executable that does not rely on these libararies will also immediately solve the
    problem. In my experience, just setting MAGNET to call the 'raxmlHPC' executable immed-
    iately solves this issue on Mac and Linux (so also try simply running MAGNET with '-e
    raxmlHPC' or '--exec raxmlHPC').

 The -b flag sets the number of boostrap pseudoreplicates for RAxML to perform while
    estimating the gene tree for each locus. The default is 100; remove bootstrapping by
    setting to 0.

 The -r flag sets the RAxML model for each locus. This uses the full default GTRGAMMA model,
        and at present it is not possible to vary the model across loci. If you want to use HKY
        or K80, you will need to use the -s flag (below).

 The -s flag sets a simple RAxML model for each locus/partition, which will override any
        model set using the -r flag above and apply to all partitions. In the current version of
        RAxML, it is possible to specify the JC69, K80, and HKY85 models as overrides. By default,
        this option is turned off and the model set under the -r flag is used instead.

 The following two options are available **ONLY** if you are starting from a NEXUS input file:

        The -g flag supplies a 'gap threshold' to an R script, which deletes all column sites in
        the DNA alignment with a proportion of gap characters '-' at or above the threshold value.
        If no gap threshold is specified, all sites with gaps are removed by default. If end goal
        is to produce a file for G-PhoCS, you  will want to leave <gapThreshold> at the default.
        However, if the next step in your pipeline involves converting from .gphocs to other data
        formats, you will likely want to set <gapThreshold> = 1 (e.g. before converting to PHYLIP
        format for RAxML).

        The -m flag allows users to choose their level of tolerance for individuals with missing
        data. The default is <indivMissingData> = 1, allowing individuals with runs of 10 or more
        missing nucleotide characters ('N') to be kept in the alignment. Alternatively, setting
        <indivMissingData> = 0 removes all such individuals from each locus; thus, while the input
        file would have had the same number of individuals across loci, the resulting file could
        have varying numbers of individuals for different loci.

 The -o flag sets the outgroup exactly the same way as that described in the RAxML v8 user's
        manual, as a single name or as a comma-separated list with no spaces between taxon names.
        The first name in the list is prioritized, e.g. when members of the list are not mono-
    phyletic.

 -R | --resume is among the most important options available in MAGNET because it tells the
        program to resume a previous run in current directory, including to detect incomplete run
        subfolders and run RAxML there without overwriting results from run folders with finished
        runs. The default setting is to run without this option.

 The -d flag runs this function in Bash debug mode (set -xv), which is intended for debugging
        for development purposes. If you find a bug, please contact the author at jbagley@jsu.edu.

 ${bold}Usage examples:${reset}

    ./MAGNET.sh -f 2 -b 100 -g 1 -m 1                        Run MAGNET with 100 bootstrap pseudo-
                                                             replicates, gaps allowed, missing
                                                             data allowed, and the GTRGAMMA model
    ./MAGNET.sh -f 2 -b 100 -s HKY85 -g 1 -m 1               Same as above, but using the simpler
                                                             HKY85 substitution model for all loci
    ./MAGNET.sh -f 2 -e raxmlHPC -b 100 -s HKY85 -g 1 -m 1   Same as above, but using raxmlHPC
                                                             executable
    ./MAGNET.sh -H                                           Show this help text and exit

 ${bold}CITATION${reset}
 Bagley, J.C. 2020. PIrANHA v0.4a4. GitHub repository, Available at:
        <https://github.com/justincbagley/piranha>.

 ${bold}REFERENCES${reset}
 Gronau, I., Hubisz, M.J., Gulko, B., Danko, C.G., Siepel, A. 2011. Bayesian inference of
        ancient human demography from individual genome sequences. Nature Genetics, 43, 1031-1034.
 Stamatakis, A. 2014. RAxML version 8: a tool for phylogenetic analysis and post-analysis of
        large phylogenies. Bioinformatics, 30, 1312-1313.

 Created by Justin Bagley on/before Aug 29 13:12:45 2016 -0700.
 Copyright (c) 2016-2020 Justin C. Bagley. All rights reserved.
```

NOTES ON NEXUS2gphocs USAGE

- You may use `NEXUS2gphocs.sh` as a standalone script for converting prior to running G-PhoCS on your data.
- However, in its current form, you must move `NEXUS2gphocs.sh` (out of the shell folder) *and* rmGapSites.r (out of the R folder) into the MAGNET directory in order to run NEXUS2gphocs as a standalone script (this assumes the target inputFile is also located in the MAGNET dir). You could also move both scripts into another working directory containing your target inputFile.
- You can get the usage info for `NEXUS2gphocs.sh`, in similar fashion to that above, by typing `./NEXUS2gphocs.sh`, `./NEXUS2gphocs.sh -h .`, or `./NEXUS2gphocs.sh -help` into the command line, and then pressing enter. The `NEXUS2gphocs` usage text is sufficiently similar to the latter part of the MAGNET usage printed above that it doesn't bear repeating here.

USAGE EXAMPLES

Below I give some examples of how to use the software under the two most common scenarios:

SCENARIO 1. If your data contain very little missing data and, in particular, they contain no individuals with all missing data for a locus, then it should be fine to run MAGNET using the default options on either a single input file (`-f 1`) or multiple PHYLIP input files (`-f 2`), as follows:

```
##--Scenario 1, generic usage:
./MAGNET.sh -f1 <inputFile>
./MAGNET.sh -f2 <workingDir>     ## multiple PHYLIP input files case.

##--Examples:
cd ~/Downloads/MAGNET-master/
./MAGNET.sh -f1 example.nex
./MAGNET.sh -f2 .     ## multiple PHYLIP input files case.
```

SCENARIO 2. If your data are relatively lower quality data (e.g. from NGS runs) and you have lots of missing data, including individuals with all missing data for a locus (as is common for RAD tag/SNP data), then RAxML will not run properly under the default MAGNET options. You will likely get up to ~10 messages like "ERROR: Sequence XXXXX consists entirely of undetermined values which will be treated as missing data", follwed by a summary like this: "ERROR: Found 10 sequences that consist entirely of undetermined values, exiting...", and RAxML will quit. The rest of the pipeline will be affected, for example the final summary gene tree file will make no sense because it will simply include a concatenation of all files in the working directory.

To avoid the above issues caused by large amounts of missing data, you should run MAGNET while **setting the -m flag to 0** (indivMissingData=0) to specify that individuals with missing data are NOT allowed:

```
##--Scenario 2, all params except indivMissingData set to default options:
./MAGNET.sh -f1 -m0 <inputFile>
./MAGNET.sh -f1 -m0 <workingDir>     ## multiple PHYLIP input files case.

##--Example:
cd ~/Downloads/MAGNET-master/
./MAGNET.sh -f1 -m0 example.nex
./MAGNET.sh -f2 -m0 .     ## multiple PHYLIP input files case.
```

In addition to the above, here are illustrations of varying the **RAxML options**:

```
##--Scenario 1, GTRCAT model, instead of the default GTRGAMMA model:
./MAGNET.sh -f1 -rGTRCAT <inputFile>
./MAGNET.sh -f2 -rGTRCAT .     ## multiple PHYLIP input files case.

##--Scenario 1, adding name of an outgroup taxon:
./MAGNET.sh -f1 -rGTRCAT -o outgroup <inputFile>
./MAGNET.sh -f2 -rGTRCAT -o outgroup .     ## multiple PHYLIP input files case.

##--Scenario 1, overriding -r model with HKY85 and adding an outgroup:
./MAGNET.sh -f1 -rGTRCAT -sHKY85 -o outgroup <inputFile>
./MAGNET.sh -f2 -rGTRCAT -sHKY85 -o outgroup .     ## multiple PHYLIP input files case.

##--Scenario 2, 500 bootstrap reps per locus, instead of the default 100:
./MAGNET.sh -f1 -b500 -m0 <inputFile>
./MAGNET.sh -f2 -b500 -m0 .     ## multiple PHYLIP input files case.

##--Scenario 2, *zero* bootstrap reps per locus:
./MAGNET.sh -f1 -b0 -m0 <inputFile>
./MAGNET.sh -f2 -b0 -m0 .     ## multiple PHYLIP input files case.
```

ACKNOWLEDGEMENTS

I gratefully acknowledge Nayoki Takebayashi, who wrote and freely provided some Perl scripts I have used in PIrANHA and MAGNET. I also thank the Brigham Young University Fulton Supercomputing Lab (FSL) for providing computational resources used during the development of this software. J.C.B. received stipend support from a Ciência Sem Fronteiras (Science Without Borders) postdoctoral fellowship from the Brazilian Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq; Processo 314724/2014-1). Lab and computer space was also supplied by The University of Alabama, during an internship in the Lozier Lab in the UA Department of Biological Sciences.

REFERENCES

- Chifman J, Kubatko L (2014) Quartet inference from SNP data under the coalescent model. Bioinformatics, 30, pages 3317–3324.
- Eaton DAR (2014) PyRAD: assembly of de novo RADseq loci for phyloge-netic analyses. Bioinformatics, 30, 1844–1849.
- Gronau I, Hubisz MJ, Gulko B, Danko CG, Siepel A (2011) Bayesian inference of ancient human demography from individual genome sequences. Nature Genetics, 43, 1031-1034.
- Larget BR, Kotha SK, Dewey CN, Ané C (2010) BUCKy: gene tree/species tree reconciliation with Bayesian concordance analysis. Bioinformatics, 26(22):2910-2911.
- Liu L, Yu L, Edwards SV (2010) A maximum pseudo-likelihood approach for estimating species trees under the coalescent model. BMC Evol Biol, 10(1):302.
- Liu L, Yu L (2011) Estimating species trees from unrooted gene trees. Syst Biol, 60(5):661-667.
- Mirarab S, Warnow T (2015) ASTRAL-II: coalescent-based species tree estimation with many hundreds of taxa and thousands of genes. Bioinformatics, 30:44-52.
- Peterson BK, Weber JN, Kay EH, Fisher HS, Hoekstra HE (2012) Double digest RADseq: an inexpensive method for de novo SNP discovery and genotyping in model and non-model species. PLoS One, 7, e37135.
- Stamatakis A (2014) RAxML version 8: a tool for phylogenetic analysis and post-analysis of large phylogenies. Bioinformatics, 30.9, 1312-1313.
- Vachaspati P, Warnow T (2015) ASTRID: Accurate Species TRees from Internode Distances. BMC Genomics, 16(Suppl 10):S3.

December 21, 2020 Justin C. Bagley, Tuscaloosa, AL, USA