Environmental Modelling & Software 62 (2014) 509–516

Contents lists available at ScienceDirect

Environmental Modelling & Software

journal homepage: www.elsevier.com/locate/envsoft

The parallel system for integrating impact models and sectors (pSIMS)☆

Joshua Elliott [a,b,*], David Kelly [a,b], James Chryssanthacopoulos [c], Michael Glotter [d], Kanika Jhunjhnuwala [e], Neil Best [a], Michael Wilde [a,b], Ian Foster [a,b,f]

[a] University of Chicago Computation Institute, 5735 S. Ellis Avenue, Chicago, IL 60637, USA
[b] Argonne National Laboratory Math & Comp. Science Division, Argonne, IL 60439, USA
[c] Columbia University Center for Climate Systems Research, 2880 Broadway, NY, NY 10025, USA
[d] University of Chicago Department of Geophysical Sciences, 5734 S. Ellis Avenue, Chicago, IL 60637, USA
[e] New Zealand Landcare Research, 231 Morrin Road, St Johns, Auckland 1072, New Zealand
[f] Department of Computer Science, University of Chicago, Chicago, IL 60637, USA

ARTICLE INFO

Article history:
Received 1 November 2013
Received in revised form
20 March 2014
Accepted 13 April 2014
Available online 22 May 2014

Keywords:
Climate change impacts, adaptation, and vulnerabilities (VIA)
Parallel computing
Data processing and standardization
Crop modeling
Forestry modeling
Multi-model
Ensemble simulation

ABSTRACT

We present a framework for massively parallel climate impact simulations: the parallel System for Integrating Impact Models and Sectors (pSIMS). This framework comprises a) tools for ingesting and converting large amounts of data to a versatile datatype based on a common geospatial grid; b) tools for translating this datatype into custom formats for site-based models; c) a scalable parallel framework for performing large ensemble simulations, using any one of a number of different impacts models, on clusters, supercomputers, distributed grids, or clouds; d) tools and data standards for reformatting outputs to common datatypes for analysis and visualization; and e) methodologies for aggregating these datatypes to arbitrary spatial scales such as administrative and environmental demarcations. By automating many time-consuming and error-prone aspects of large-scale climate impacts studies, pSIMS accelerates computational research, encourages model intercomparison, and enhances reproducibility of simulation results. We present the pSIMS design and use example assessments to demonstrate its multi-model, multi-scale, and multi-sector versatility.

Software and data availability

The first open release of the source code is planned for 2014.

1. Introduction

Understanding the vulnerability of human society to climate change is necessary for sound decision-making in climate policy. However, the research needed to build this understanding is hindered by the fact that science and information products must be integrated across vastly different spatial and temporal scales (Rosenzweig et al., 2013). Biophysical responses to global change generally depend on environmental (e.g., soil type), socioeconomic (e.g., farm and forest management), and climatic factors that vary substantially over regions. Global Gridded Biophysical Models (GGBMs) are designed to capture spatial heterogeneity and simulate biophysical responses (e.g., of crops and forests) to climate over large areas (Rosenzweig et al., 2014; Elliott et al., 2014a). GGBMs derived from site-based ("field-scale" or "stand-scale") models (e.g., APSIM, DSSAT, and CenW) estimate some measures of productivity (e.g., crop yield) from local management, climate, and soil profiles that represent a single field or stand. By running site-based models many times to simulate behavior at different sites—in some studies considered here, at 10s or even 100s of thousands of sites—GGBMs can provide information at an unprecedented detail and scale. However, such studies require high-resolution data concerning soil types, environmental conditions, and management practices at many individual sites.

The primary obstacle to obtaining the data inputs necessary for a comprehensive high-resolution assessment of climate impacts is often not that data does not exist, but that the task of converting this data into a usable form involves much effort and expertise. Researchers must typically catalog, assimilate, test, and process data from multiple sources with vastly different spatial and temporal scales and extents. Each dataset may come in a different

☆ Thematic Issue on Agricultural Systems Modeling & Software.

* Corresponding author. University of Chicago Computation Institute, 5735 S. Ellis Avenue, Chicago, IL 60637, USA. Tel.: +1 212 678 5630; fax: +1 773 834 3700.

E-mail address: jelliott@ci.uchicago.edu (J. Elliott).

http://dx.doi.org/10.1016/j.envsoft.2014.04.008
1364-8152/© 2014 Elsevier Ltd. All rights reserved.

format and have a unique set of issues and quirks. The labor-intensive and error-prone process of accessing, understanding, scaling, and integrating such diverse data involves the creation of what is, in effect, a custom data processing pipeline for every study. A comparably complex set of transformations must often be performed on simulation outputs to produce information products for stakeholders—including farmers, policy-makers, markets, and agro-business interests—that operate at very different scales.

To address these challenges and facilitate access to high-resolution climate impact modeling we have developed a suite of tools, data, and models called the parallel System for Integrating Impacts Models and Sectors (pSIMS). This system largely automates the labor-intensive processes of creating and running data ingest and transformation pipelines and allows researchers to use high-performance computing to run simulations that extend over large spatial extents, run for many growing seasons, or evaluate many alternative management practices or other input configurations. In so doing, pSIMS dramatically reduces the time and technical skills required to investigate global change vulnerability, impacts and potential adaptations. pSIMS is designed to support integration and high-resolution application of any site-based climate impact model that can be compiled in a Unix environment (with a focus on primary production: agriculture, livestock, and forestry). A variety of existing and ongoing efforts have developed software frameworks for parallel spatial simulations of a specific impact model on a specific compute cluster (e.g., Bryan, 2013; Nichols et al., 2011; Vital et al., 2013; Zhao et al., 2013, 2012). The open-source pSIMS framework attempts to improve on these by adding features such as multi-model versatility, system portability, and robust fault tolerance.

In this paper we detail the pSIMS structure and methodology (Sections 2–4); describe two example applications (Sections 5 and 6); and summarize features of the high-performance software and computational architecture (Section 7). The pSIMS methodology partitions the simulation process into four major stages: data ingest and standardization (Section 2), campaign specification (Section 3), campaign implementation (Section 4), and aggregation to arbitrary decision-relevant scales (Fig. 1). pSIMS currently supports GGBMs constructed from several site-based models: versions 4.0 and 4.5 of the Decision Support System for Agrotechnology Transfer (DSSAT; Jones et al., 2003; Hoogenboom et al., 2010), versions 7.4 and 7.5 of the Agricultural Production Systems Simulator (APSIM; Keating et al., 2003), and version 4.0 of the CenW forest growth simulation model (Kirschbaum, 1999). We denote the pSIMS implementations of these models as parallel DSSAT (pDSSAT), parallel APSIM (pAPSIM), and parallel CenW (pCenW), respectively.

To date, pSIMS has been used to conduct continental to global scale simulation experiments ranging in resolution from 3 to 30 arcminutes on six crops (maize, wheat, soy, rice, sorghum, and millet) and one tree species (*pinus radiata*) (e.g., Elliott et al. 2014a,b, Elliott et al., 2013, Rosenzweig et al., 2014). We have used pSIMS to conduct simulations with daily weather inputs from over 40 distinct data products including historical station observations, model reanalysis-based data, global and regional climate model outputs, and seasonal forecast model outputs, and for global and continental scale simulations over both historical and future periods under dozens of socio-economic scenarios including fixed present day management and various potential climate adaptation pathways.

2. The pSIMS climate data input pipeline

The minimum weather data requirements for site-based crop and climate impact models such as DSSAT, APSIM, and CenW are typically:

- Daily maximum temperature (degrees C at 2 m above the ground surface)
- Daily minimum temperature (degrees C at 2 m above the ground surface)
- Daily average downward shortwave radiation flux (W/m^2 measured at the ground surface)
- Total daily Precipitation (mm/day at the ground surface)

Some applications also require daily average wind speeds and a measure of humidity (typically expressed as daily average dew-point temperature or vapor pressure or the relative humidity measured at the time of daily max temperature).

Thousands of observational datasets, model-based reanalyses, and multi-decadal climate model simulation outputs at regional or global scales are available to drive impact simulations (see for example catalog and archive services such as Williams et al., 2009; Rutledge et al., 2006). These datasets are typically stored in standard formats, with substantial standard metadata (Eaton et al., 2010), and are frequently identified by a unique Digital Object Identifier (DOI; Paskin, 2005), making provenance and tracking feasible. Nevertheless, data must often be remapped before use due to varying spatial scales (from a few kilometers to several degrees), temporal resolutions (from an hour to a month), and map projections. For some use cases, one or another data product may be demonstrably superior to all others, but often simulations are re-run with multiple different inputs to obtain a clear understanding of the range of outcomes and uncertainty. For these reasons, input data sizes for high-resolution climate impact experiments can be large and data processing and management can be challenging.

Daily time series of high-resolution climate data from observation, reanalysis, or model simulations provide natural inputs to crop

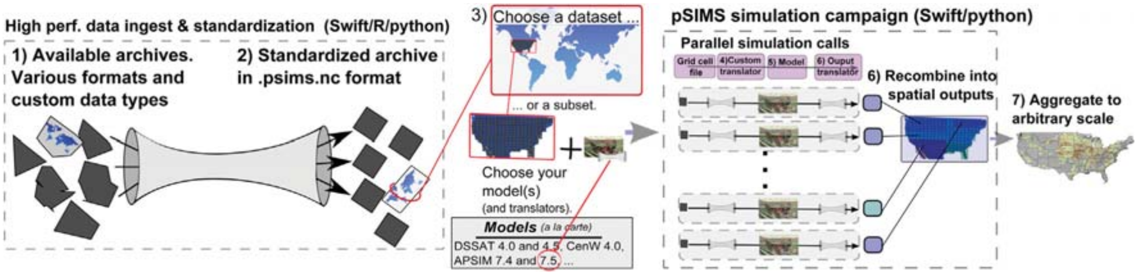
High perf. data ingest & standardization (Swift/R/python)
1) Available archives. Various formats and custom data types
2) Standardized archive in .psims.nc format
3) Choose a dataset ...
... or a subset.
Choose your model(s) (and translators).
Models (a la carte)
DSSAT 4.0 and 4.5, CenW 4.0, APSIM 7.4 and 7.5, ...
pSIMS simulation campaign (Swift/python)
Parallel simulation calls
Grid cell file
4)Custom translator
5) Model
6) Ouput translator
6) Recombine into spatial outputs
7) Aggregate to arbitrary scale

Fig. 1. Schematic of pSIMS workflow. 1) Data ingest from arbitrary file formats and datatypes. 2) Standardization reconstitutes each such dataset into one or more files in the portable site-based .psims.nc format. 3) Specification of a set of weather, soil, and management files and one or more climate impact models from the code library (Section 3). 4) Translation converts the selected .psims.nc file(s) into the custom file format(s) required by models (Section 4). 5) Simulation runs a separate simulation process per site, with Swift used to manage execution on selected computer(s). 6) Output reformatting extracts model outputs (dozens or even hundreds of time-series variables from each run) from model-specific custom output formats and translates them into a standard output format and then into compressed spatial NetCDF4 files (Section 4). 7) Aggregation masks and aggregates output variables to specified decision-relevant regions and spatial or temporal scales (Section 4).

and climate impact models. One common obstacle to the convenient use of such data relates to file organization and structure. Climate data is typically stored one or more variables at a time in large spatial raster files that are segmented into annual, monthly, or sometimes even daily or sub-daily time-slices, so that reading the data for all grid points and a single time slice is straightforward. Site-based impact models, on the other hand, typically require custom datatypes that encode long time series of daily data for all required variables at a single point. To create such a data file from gridded weather data can require accessing hundreds or thousands of spatial raster files, each of which must be read many times to extract the time series for each point (Malik et al., 2011). To ameliorate this challenge, we have defined a NetCDF-based datatype for the pSIMS framework—identified by a psims.nc extension—and tools for translating existing archives to this format (Fig. 2).

Each .psims.nc file represents a single grid cell or simulation site within the study area. Its contents are one or more $1 \times 1 \times T$ arrays, one per variable, where T is the number of time steps—an organization comparable to that often used for site-based weather station data. The time coordinates of each array are explicit in the definition of its dimensions, a strategy that facilitates near-real-time updates from upstream sources (since new data can be appended as it becomes available and the record of what has already been ingested is self-contained within the file metadata).

The spatial coordinates of each array are also explicit in its dimensions, a strategy that facilitates downstream spatial merging. Because a given file contains information about a single point location, the longitude and latitude vectors used as array dimensions are of length one. In contrast, arrays containing forecast variables have not one but two time dimensions: one for the time when the forecast was made and the other for the time for which the values are predicted. This use of two time dimensions makes it possible to follow the evolution of a series of forecasts made over a period of time for a particular future date, as for example when forecasts of crop yields at a particular location are refined over time as more information becomes available.

We adopt a standardized naming scheme and organization for pSIMS input files that incorporates metadata about the (row, col) tuple that denotes a file's location in the global grid in both the file name and directory structure. Thus, for example, a file named /agcfsr.15min/0456/0859/0456_0859.psims.nc contains data about the site at row 456 and column 859 in a global 15 arcminute grid (which is in Southern Mozambique at latitude −23.875 and longitude 34.625). A consequence of this organization is that each terminal directory holds data for a single site. This organization can improve parallel input/output performance on shared filesystems and minimizes clutter while browsing directory trees.

3. Specifying and running a simulation campaign

We refer in general to a set of simulations run with pSIMS as a "simulation campaign." A campaign typically involves one model (pDSSAT, pAPSIM, or pCenW), one species (maize, soy, pine, etc.), and one region run for dozens or hundreds of seasons and 10–100 different management or parameter configurations. A simulation campaign requires climate and soils data for the target region, along with management and technology inputs (e.g., genetic parameters for different crop cultivars) that may vary widely by model and experiment. A given simulation experiment, such as the one described in Section 5, typically includes many simulation campaigns to consider many crops or to explore uncertainty from different climate forcings. A campaign is specified by the contents of three files that encode experiment details and other options for the requested simulations: the parameter control file, scenario template file, and master campaign file.

The parameter control file includes all parameters that are passed to pSIMS for a given campaign: pointers to data, command line executables, command line options for translator apps, specification of the space-time grid and scenarios, and variables to extract from the output. See Listing 1 for an example parameter file for a pDSSAT campaign that simulates maize in Eastern/Southern Africa at 15 arcminute resolution, the results of which are included in Section 5.

The scenario template is a standardized JSON file in the AgMIP crop experiment (ACE) format (Porter et al., 2014) that records only one of the many experiments (sometimes called scenarios or "treatments" in DSSAT parlance) for a given campaign. The template includes all required data sections for a model run, such as management events, and uses user-specified default values for all inputs.

The NetCDF4-format *master campaign file* specifies all inputs that vary over the set of experiments that define a given campaign, whether these elements vary in space or not. The file has dimensions of latitude, longitude, and scenario, where "scenario" is a general concept that users can exploit in various ways to simulate different management settings, environmental conditions, or input datasets (scenarios translate directly to "treatments" in pDSSAT campaigns). This file is used by the campaign translator to substitute parameters into the experiment template file (see Section 4) and can be used to adjust parameters in any combination of the dimensions. For example, variables with spatial dimensions but no

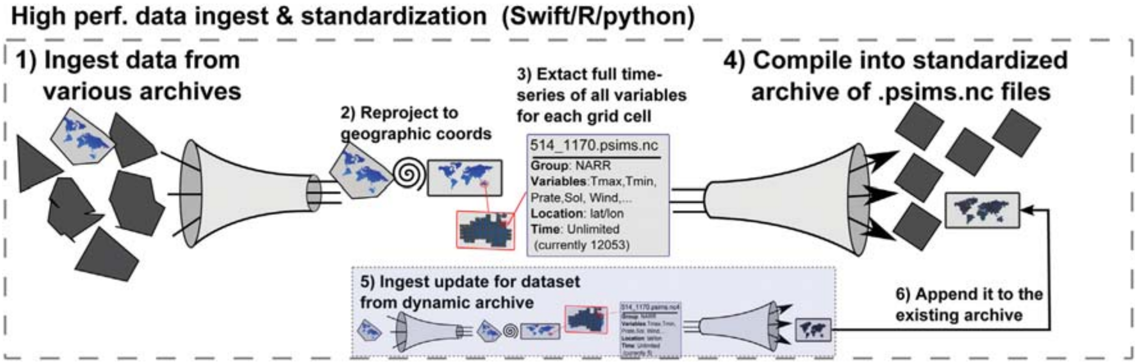
High perf. data ingest & standardization (Swift/R/python)
1) Ingest data from various archives
2) Reproject to geographic coords
3) Extact full time-series of all variables for each grid cell
514_1170.psims.nc
Group: NARR
Variables: Tmax, Tmin, Prate, Sol, Wind,...
Location: lat/lon
Time: Unlimited (currently 12053)
4) Compile into standardized archive of .psims.nc files
5) Ingest update for dataset from dynamic archive
6) Append it to the existing archive

Fig. 2. Expanded schematic of datatypes and processing pipeline from Fig. 1. Steps in the pipeline are: 1) Data ingest from arbitrary sources in various formats and datatypes. 2) If necessary, data is transformed to the standard pSIMS geographic projection. 3) For each land grid-cell in the input data, the full time series of data is extracted (in parallel) and converted to the .psims.nc format. 4) The resulting set of .psims.nc files are organized into an archive for long-term storage. 5) If the input dataset is still being updated, we ingest and process updates at regular intervals and 6) append the updates to the .psims.nc archive.

scenario dimension are applied to all scenarios at the appropriate (latitude, longitude) pair, while variables with a scenario dimension but no spatial dependence are applied uniformly at all locations.

4. The pSIMS codebase

pSIMS runs in a UNIX environment and requires common dependencies such as Python and NetCDF operators (NCO; http://nco.sourceforge.net/) along with some less common dependencies such as the Swift parallel scripting language (Section 7). Additional software packages may be needed to compile or run a particular model, such as the Free Pascal Compiler (http://www.freepascal.org/) for CenW and the Boost C++ libraries (http://www.boost.org/) and Mono (http://www.mono-project.com) for APSIM.

The pSIMS code base also includes a set of translator apps, software utilities that are used to accomplish a wide variety of tasks within the framework, ranging from simple data reformatting to statistical data processing (e.g., applying statistical perturbations or bias-corrections to a given input climate dataset). Some translator apps are used for offline processing, such as converting climate data into psims.nc format, while others are used while processing a campaign. Users can easily incorporate custom translator apps into the framework using any number of software packages, use multi-model translator utilities developed in the AgMIP framework (Porter et al., 2014), or use existing utilities distributed with pSIMS. Here we describe some key translator types and give examples of apps that are distributed with pSIMS v0.9 (Fig. 3). We do not attempt an exhaustive accounting of offline utilities, but rather focus on the translator apps that are key to simulation campaign execution and to incorporating new models into the framework.

Campaign translator apps extract parameters from the master campaign file and populate the necessary fields in the scenario template file to produce a JSON file (in the standard AgMIP ACE format) that contains all management and scenario info needed for the simulation. pSIMS v0.9 is distributed with a single model-agnostic campaign translator utility (camp2json.py).

Experiment translator apps convert from the JSON-based model-agnostic AgMIP experiment format into the model-specific file(s) that are needed for a simulation. Besides this "experiment" file generated by the campaign translator, it also pulls in the relevant soil data (in generic ACE soil data format) from the pre-computed archive. pSIMS v0.9 is distributed with customized python apps for DSSAT and APSIM formats (jsons2dssat.py and jsons2apsim.py) which are closely related to the AgMIP QUADUI data translators (it was convenient in early work to create our own versions, but in future releases we expect to support the native AgMIP family of experiment translators as well).

Weather translator apps convert .psims.nc climate files to the model-specific format needed for a particular simulation, converting units, deriving combined variables, and (if requested) perturbing or bias-correcting the data series in the psims.nc file.

Output translator apps convert data from the custom ASCII output formats that the models produce into standard site-based NetCDF files that use the psims.nc standards. Each file contains all the user-requested variables (as defined in the parameter file, see Listing 1 for example) for a given point in a compact self-describing file that can be conveniently merged into spatial NetCDF files.

5. A multi-model multi-scale example assessment study with pDSSAT and pAPSIM

To demonstrate how pSIMS facilitates multi-model multi-scale assessments of crop growth and climate impacts, we describe four pSIMS campaigns that we conducted for maize in Africa from 1980 to 2010 (Fig. 4) using climate forcings from AgCFSR. The four campaigns involve two models (pDSSAT and pAPSIM), each simulating a) the full continent at 0.5° spatial resolution (10,301 grid cells for 30 years) and b) the Southern/Eastern African countries of Zimbabwe, Malawi, Zambia, Tanzania, Mozambique, Ethiopia, Burundi, Rwanda, Uganda, Kenya, and Somalia at 0.25° spatial resolution (7778 grid cells, again for 30 years). These campaigns are small compared to long-duration 0.5° global climate impact runs that we have performed (56,537 land grid cells, 150 years), but convey the versatility of the framework. All runs were conducted with the same fertilizer inputs, generated by combining organic and chemical fertilizer data from three sources (Mueller et al., 2012; Potter et al., 2010; Foley et al., 2011) and similar sowing dates (pAPSIM uses a fixed sowing date each year while pDSSAT has a variable planting rule confined to a period around the observed value) based on the SAGE (Sacks et al., 2010) and MIRCA2000 (Portmann et al., 2010) crop calendar datasets (extrapolated globally using environmental considerations).

We match cultivar coefficients as closely as possible between the two models and map them to the same spatial grid to reproduce observed maturity dates from crop calendars. For this purpose, we defined 10 generic hybrid and 10 generic open-pollinated cultivars in consultation with crop model experts, using open pollinated variants for grid cells dominated by subsistence maize farming, according to You and Wood (2006). We distribute these cultivar definitions with the pSIMS software.

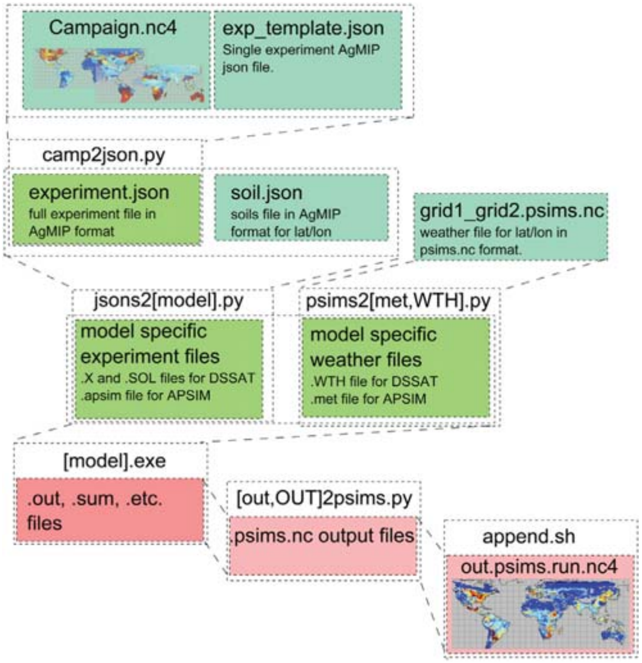

Fig. 3. A sketch of the simulation campaign framework (labeled 4–6 in Fig. 1) including the basic translator apps released with pSIMS v0.9 and input/output filenames for DSSAT v4.5 and APSIM v7.5. *camp2json.py* takes user-specified input files (Campaign.nc4 and exp_template.json) to generate generic experiment files (containing all necessary management inputs) in the model-agnostic AgMIP JSON format. *jsons2[model].py* combines these with pre-computed soil profiles in AgMIP format and converts to model specific input files. *psims2[met,WTH].py* generates model-specific weather files. Once the impact model is executed, all the raw input and output files are collected into a compressed archive to ensure future reproducibility, and a user specified output processing routine (e.g., *out2psims.py*) is called to extract the desired variable subset and convert it to standard psims output format. Finally, once all simulations for a given campaign have completed, the append utility collects each variable in highly-compressed and portable NetCDF v4 files.

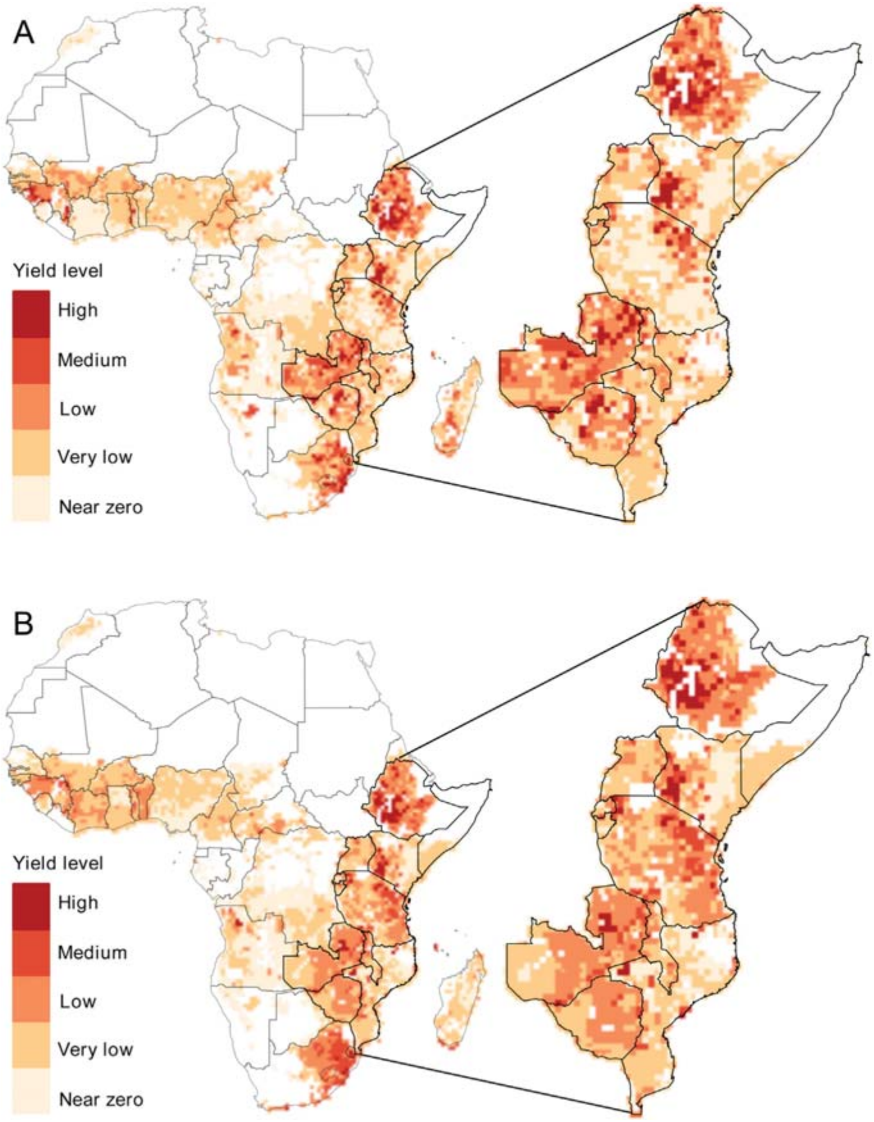

Fig. 4. Example outputs of a multi-model multi-scale pSIMS simulation for A) pDSSAT and B) pAPSIM. Plots show 1980–2010 mean simulated yields for Africa maize at 0.5° resolution, alongside the same simulations run at 0.25° resolution for Southern/Eastern Africa (Zimbabwe, Malawi, Zambia, Tanzania, Mozambique, Ethiopia, Burundi, Rwanda, Uganda, Kenya, Somalia). Grid cells with zero harvested maize area (according to the MIRCA2000 dataset) are not shown.

A single simulation with APSIM v7.5 (six management configurations for 31 years each) on a Dual 8-core Intel 2.6 GHz “Sandy Bridge” Xeon E5-2670 processor took 118 s, increasing to 184 s when 16 such simulations are run simultaneously on a single node. The 10.3K tasks of the full Africa run would thus take about 14.1 days on a single core, but when run in parallel on 96 compute nodes (1536 total cores) take only about 21 min, for a speedup of about 1000 times. Run-time for a single DSSAT v4.5 simulation (with the same dimensions and on the same node-type) is about 19 s, while run-time for 16 DSSAT simulations on a single node is typically about 20–25 s (2.3 days in serial or 2.2–2.8 min in parallel on 96 nodes; a speedup of about 1300 times).

6. An example assessment with pCenW

pSIMS has been applied to study climate change impacts on plantation forestry in New Zealand using the CenW forest growth simulation model (Kirschbaum, 1999). Site-specific parameters include a fertility index based on a national nitrogen surface, water-holding capacity, and percent fine soil (silt and clay). The CenW model can be configured for many tree species; the pSIMS-parallelized CenW used here, pCenW, is set up for *pinus radiata* (the most common commercial timber species in New Zealand, and indeed worldwide). pCenW runs each site individually with a daily time step. It can run any simulation period but has only been calibrated for managed forests with rotation ages between 15 and 50 years. Output can be generated at any time scale down to daily records, allowing for complex calculations and a better understanding for how trees in a particular location are likely to grow.

Fig. 5 shows estimates of merchantable timber (m^3/ha) for New Zealand *pinus radiata* stands 30 years after planting in 1972, 2012, and 2052, estimated from pCenW simulations. The simulations were run at the 0.05° (approx. 5 km) grid cell level using climate data from the HadleyCM3 model under the.SRES A1B scenario. These simulations indicate that timber volume on current plantation area in NZ is expected to increase over time, driven entirely by changes in climate (i.e., excluding CO_2 fertilization effects). This increase in stand volume is likely to result in greater levels of forest

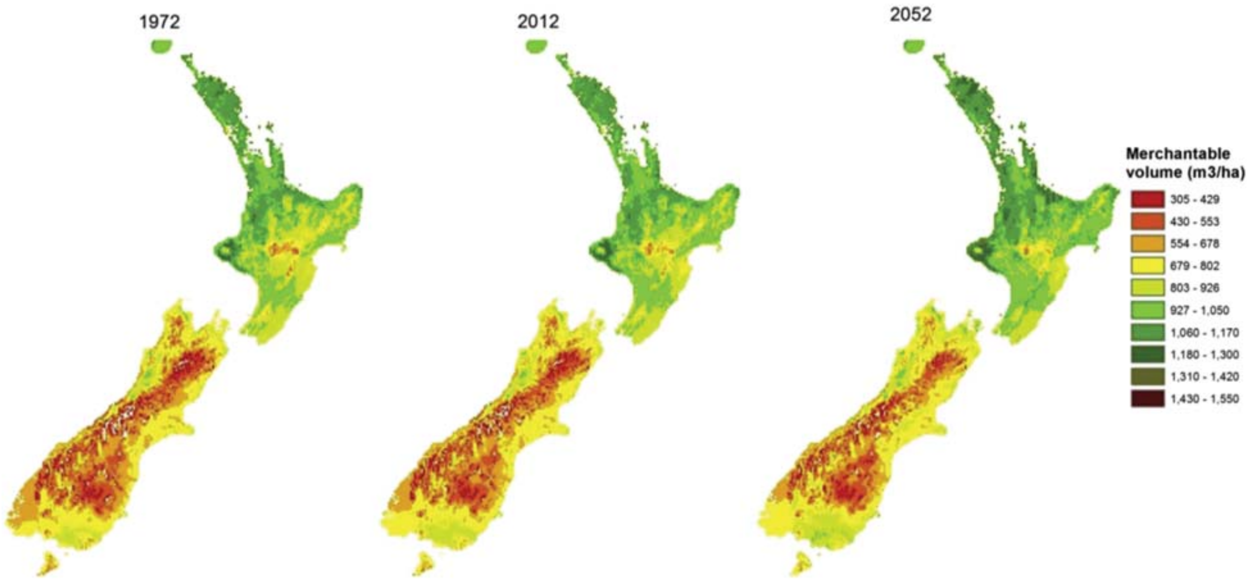

Fig. 5. pCenW estimate of merchantable timber volume (m^3/ha) for New Zealand *pinus radiata* stands 30 years after planting in 1972, 2012, and 2052, under climate model HadCM3 and scenario SRES A1B.

carbon sequestration for the country overall, though economic adaptations to faster tree growth (such as a reduction in the optimal harvest age) may affect these trends.

7. Use of *Swift* parallel scripting

Each pSIMS simulation campaign typically requires the execution of O(10^4–10^5) small serial jobs, one per grid cell, each of which uses one CPU core for anywhere from 30 s to many minutes. The Swift parallel scripting language (Wilde et al., 2009) makes it straightforward to write and execute pSIMS runs, using highly portable and system-independent scripts such as the example in Listing 2. Space does not permit a detailed description of Listing 2, but in brief, it 1) defines a machine-independent interface to the model executable, 2) loads a list of geographic locations on which that executable is to be run, and 3) defines a set of simulations at these locations.

The Swift language is implicitly parallel, high-level, and functional. It automates the difficult and science-distracting tasks of distributing tasks and data across multiple remote systems, and of retrying failing tasks and restarting failing workflow runs. Its runtime can manage efficiently the execution of many small single-core or multi-core jobs, dynamically packing those jobs tightly onto multiple computing nodes to maximize system utilization. Swift automates node acquisition; inter-job data dependencies; throttling; scheduling and dispatch of work to cores; and retry of failing jobs.

The key to Swift's ability to execute large numbers of small tasks efficiently on large parallel computers is its use of a two-level scheduling strategy. Internally, Swift launches a pilot job called a "coaster" on each node within a resource pool (Hategan et al., 2011). The Swift runtime then manages the dispatch of application invocations, plus any data that these tasks require, to those coasters, which manage their execution on compute nodes. As tasks finish, Swift schedules more work to those nodes, achieving high CPU utilization even for fine-grained workloads. Individual tasks can be serial, OpenMP, MPI, or other parallel applications. Swift makes computing location independent, allowing us to run pSIMS on a variety of grids, supercomputers, clouds, and clusters, with the same scripts used on multiple distributed sites and diverse resources. Fig. 6 shows a typical execution scenario, in which pSIMS is run across two University of Chicago campus resources: the UChicago Campus Computing Cooperative (UC3) (Bryant, 2012) and the UChicago Research Computing Center (UCRCC).

We have run pSIMS on UC3, Open Science Grid (Pordes et al., 2007), and UCRCC; the XSEDE clusters Ranger and its successor Stampede; the Amazon Elastic Compute Cloud (EC2) and several other clusters and supercomputers. In production mode (i.e., excluding testing and prototype stages) pSIMS has been used for more than 100 large-scale simulation campaigns of DSSAT, CenW, and APSIM. These campaigns have already totaled over 5.6 million individual DSSAT runs, each of 30–150 years and including 10–100 independent scenarios, and a growing number of runs with other models such as CenW and APSIM (Table 1).

8. Discussion

The parallel System for Integrating Impacts Models and Sectors (pSIMS) is a new framework for efficient implementation of large-

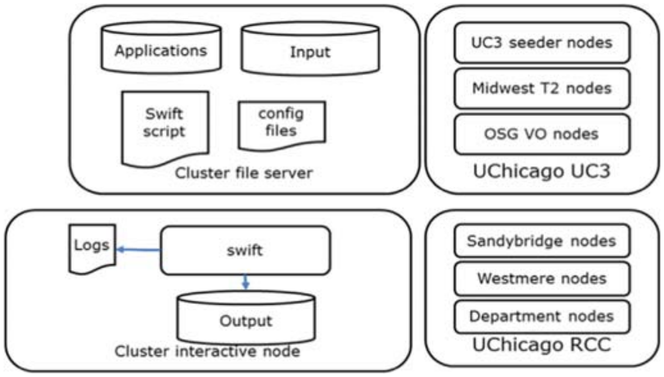

Fig. 6. Typical Swift configuration for pSIMS execution.

```
 model          dssat45
 out_file       out.psims.dssat45.agcfsr.15min.seafr.maize

// Data pointers
 campaign       campaigns/pdssat/seafr
 refdata        campaigns/pdssat/common
 weather        data/clim/agcfsr.15min
 soils          data/soils/hwsd200.seafr.15min
 bintransfer    bin/DSCSM045.EXE,tapps/pdssat/psims2WTH.py,tapps/pdssat/OUT2psims.py,
                 bin/camp2json.py,tapps/pdssat/jsons2dssat.py

// Command line inputs for translation utilities
 tappwth        psims2WTH.py -o GENERIC1.WTH -v tmin,tmax,rsds,pr,wind
 tappcamp       camp2json.py -c Campaign.nc4 -e exp_template.json -o experiment.json
 tappinp        jsons2dssat.py -x X1234567.MZX -s soil.json -e experiment.json -S SOIL.SOL

// Execution and other basic options
 executable     DSCSM045.EXE A X1234567.MZX
 ref_year       1980          // First year for which climate data is provided
 num_years      31            // Total number of years to simulate
 delta          15            // Grid cell spacing measured (in arcminutes)
 scens          12            // Number of scenarios simulated

// Output file generation and processing options
 outtypes       .WTH,.MZX,.SOL,.OUT,.json,.txt  // File types to include in output tarball
 lat_zero       15            // Northern edge of the data in output nc4 file (in degrees)
 lon_zero       21.75         // Western edge of the data in output nc4 file (in degrees)
 num_lats       168           // Total number of latitudes in output nc4 file
 num_lons       119           // Total number of longitude in output nc4 file
 variables      ADAT,MDAT,HWAM,ETCP,LAIX        // Name of each variable in .out file
 var_units      "DAS,DAS,kg/ha,mm,frac"         // Comma separated list of variable units
 long_names     "AnthesisDate,MaturityDate,Yield,GrowingSeasonET,MaxLeafAreaIndex"
 postprocess    OUT2psims.py -i Summary.OUT
```

Listing 1. Annotated example parameter file for a pDSSAT campaign for maize in Southern/Eastern Africa.

```
// 1) Define RunpSIMS function as an interface to the pSIMS program

   // a) Specify function prototype
app (file tar_out, file part_out) RunpSIMS (file scenario[], file weather[], file
      soils[], file common[], file binary[],string latidx, string lonidx, file param)
   // b) Specify how to invoke the pSIMS executable
                                   { RunpSIMS latidx lonidx @param @tar_out; }

// 2) Define the post-processing functions
app (file outfile) append (file inputArray[], file param)
                  { append 1 1 @param "./parts"@outfile; }
app (file outfile) merge (int var_num, file inputArray[], file param)
                  { merge var_num "1" "./var_files" @param;}
app (file outfile) combine (file inputArray[], file scenarioInput[], file param)
                  { combine @param; }

// 3) Read the list of grids over which computation is to be performed and other common file inputs
string grid[] = readData("gridList.txt");
file part_outs[][];
file scen[] <filesys_mapper; location=@arg("campaign"), pattern="*">;
file common[] <filesys_mapper; location=@arg("refdata"), pattern="*">;
file binary[] <fixed_array_mapper; files=@arg("bintransfer")>;
file params <"params.psims">;

// 4) Invoke RunpSIMS once for each grid; different calls are concurrent
foreach g,i in grid {
 string gridN[] = @strsplit(g, "/");

 // a) Map the various grid-dependent input and output files to Swift variables
 // Input files
 file weather[]<filesys_mapper;location=@strcat(@arg("weather"),"/",grid[i]), pattern="*">;
 file soils[] <filesys_mapper; location=@strcat(@arg("soils"),"/",grid[i]), pattern="*">;

 // Output files
 file tar_out <single_file_mapper; file=@strcat("output/", grid[i], "output.tar.gz")>;
 file part_out <single_file_mapper; file=@strcat("parts/", grid[i], ".psims.nc")>;

 // b) Call RunpSIMS
 (tar_out, part_out) = RunpSIMS(scen,weather,soils,common,binary,gridN[0],gridN[1],params);
   part_outs[@toInt(gridN[0])][@toInt(gridN[1])] = part_out;
}

// 5) Call post-processing functions append, merge, and combine (not shown due to space constraints)
```

Listing 2. Annotated pSIMS Swift script.

scale assessments of climate vulnerabilities, impacts, and adaptations across multiple sectors and at unprecedented scales. pSIMS includes an extensible, high-performance data ingest and processing pipeline that generates a standardized collection of management, environmental, and climate datasets based on portable and efficient datatypes, as well as a code base to enable large-scale, high-resolution simulations of the impacts of changing climate on primary production (agriculture, livestock, and forestry) using many site-based climate impact models.

These new capabilities are enabled by the use of high-performance computing, which in turn is harnessed by the Swift parallel scripting language. The pSIMS framework also contains data translation tools that can handle the input and output formats used in various models; specifications for integrating translators developed in AgMIP; and tools for aggregation and scaling of simulation outputs to arbitrary spatial and temporal scales relevant for decision support, validation, and downstream model coupling. This framework has been used for high-resolution crop yield and

Table 1
Summary of campaign execution by project, including the total number of jobs in each campaign, the total number of simulation units (jobs × scenarios × years), the total model CPU time, and the total size of the generated outputs.

Project	Campaigns	Sim Units (Billion)	CPU hours (K)	Jobs (M)	Output data (TBytes)
NARCCAP USA (pDSSAT)	16	1.3	13	1.9	0.47
ISI-MIP Global (pDSSAT)	80	11.8	216	4.38	4.14
Prediction 2012 (pDSSAT)	2	0.2	2	0.24	0.5
NZ climate impacts (pCenW)	2	0.4	3	0.1	0.6
GGCMI Phase 1 (pDSSAT and pAPSIM; ongoing)	~96	~12	~200	~25	~1

climate impact assessments at the US and global levels (Elliott et al. 2014a,b, Elliott et al., 2013; Rosenzweig et al., 2014; Glotter et al., 2014).

Acknowledgments

We thank Pierre Riteau and Kate Keahey for help running pSIMS on clouds; Stephen Welch of Kansas State and Dan Stanzione, John Fonner, and Matthew Vaughn of TACC, for help executing Swift workflows on Ranger and Stampede, and under the iPlant portal; Ravi Madduri of Argonne and UChicago for help placing pSIMS under the Galaxy portal; and Cheryl Porter and Chis Villalobos of University of Florida for help with AgMIP translators. Thanks also to countless DSSAT, APSIM, and CenW experts that have answered questions and helped with parameterizations, including Ken Boote, Jim Jones, Cheryl Porter, Senthold Asseng, Sotirios Archontoulis, Peter Thorburn, and Miko Kirschbaum. This work was supported in part by the National Science Foundation under grants SBE-0951576 and GEO-1215910. Swift is supported in part by NSF grant OCI-1148443. Computing resources used included the XSEDE Stampede machine at TACC, the University of Chicago Computing Cooperative, the University of Chicago Research Computing Center, and the Beagle system funded by NIH grant S10 RR029030-01.

References

AgMIP source code repository. Available from: https://github.com/agmip (accessed 01.04.13).

Bryan, B.A., 2013. High-performance computing tools for the integrated assessment and modelling of social–ecological systems. Environ. Model. Softw. 39, 295–303.

Bryant, L., 2012. UC3: a Framework for Cooperative Computing at the University of Chicago Open Science Grid Computing Infrastructures Community Workshop. University of California Santa Cruz. http://1.usa.gov/ZXum6q.

Eaton, B., Gregory, J., Drach, B., Taylor, K., Hankin, S., Caron, J., Signell, R., Bentley, P., Rappa, G., Höck, H., Pamment, A., Juckes, M., 2010. NetCDF Climate and Forecast (CF) Metadata Conventions, version 1.5. Lawrence Livermore National Laboratory. http://cf-pcmdi.llnl.gov/documents/cf-conventions/1.5/cf-conventions.html.

Elliott, J., Deryng, D., Muller, C., Frieler, K., Konzmann, M., Gerten, D., Glotter, M., Florke, M., Wada, Y., Eisner, S., Folberth, C., Foster, I., Gosling, S., Haddeland, I., Khabarov, N., Ludwig, F., Masaki, Y., Olin, S., Rosenzweig, C., Ruane, A., Satoh, Y., Schmid, E., Stacke, T., Tang, Q., Wisser, D., 2014a. Constraints and potentials of future irrigation water availability on global agricultural production under climate change. Proc. Natl. Acad. Sci. U.S.A. 111 (9), 3239–3244.

Elliott, J., Sharma, B., Best, N., Glotter, M., Dunn, J.B., Foster, I., Miguez, F., Mueller, S., Wang, M.Q., 2014b. A spatial modeling framework to evaluate domestic biofuel-induced potential land use changes and emissions. J. Environ. Sci. Technol. 48 (4), 2488–2496 http://dx.doi.org/10.1021/es404546r.

Elliott, J., Glotter, M., Best, N., Boote, K.J., Jones, J.W., Hatfield, J.L., Rosenzweig, C., Smith, L.A., Foster, I., 2013. Predicting agricultural impacts of large-scale drought: 2012 and the case for better modeling. RDCEP Working Paper No. 13-01. http://ssrn.com/abstract=2222269.

Foley, J.A., et al., 2011. Solutions for a cultivated planet. Nature 478 (7369), 337–342.

Glotter, M., Elliott, J., McInerney, D., Best, N., Kelly, D., Foster, I., Moyer, E., 2014. Robustness of agricultural impacts projections to downscaling approaches for climate inputs. Proc. Natl. Acad. Sci. U.S.A. in press.

Hategan, M., Wozniak, J., Maheshwari, K., 2011. Coasters: uniform resource provisioning and access for clouds and grids. Fourth IEEE International Conference on Utility and Cloud Computing (UCC '11), Washington, DC, USA, 2011, IEEE Computer Society, 114–121.

G Hoogenboom, CH Porter, PW Wilkens, KJ Boote, LA Hunt, JW Jones, et al. The decision support system for Agrotechnology Transfer (DSSAT): past, current and future developments. In: Program and Summaries, 40th Biological Systems Simulation Conference, Maricopa, AZ, pp. 13–15, 2010.

Jones, J.W., Hoogenboom, G., Porter, C.H., Boote, K.J., Batchelor, W.D., Hunt, L.A., Wilkens, P.W., Singh, U., Gijsman, A.J., Ritchie, J.T., 2003. The DSSAT cropping system model. Eur. J. Agron. 18 (3–4), 235–265.

Keating, Brian A., Carberry, P.S., Hammer, G.L., Mervyn, E. Probert, Robertson, M.J., Holzworth, D., Huth, N.I., et al., 2003. An overview of APSIM, a model designed for farming systems simulation. Eur. J. Agron. 18 (3), 267–288.

Kirschbaum, M.U.F., 1999. CenW, a forest growth model with linked carbon, energy, nutrient and water cycles. Ecol. Model. 118 (1), 17–59.

Malik, T., Best, N., Elliott, J., Madduri, R., Foster, I., 2011. Improving the Efficiency of Subset Queries on Raster Images. In: HPDGIS '11: Second International Workshop on High Performance and Distributed Geographic Information Systems, Chicago, Illinois, USA. ACM, pp. 34–37.

Mueller, N.D., et al., 2012. Closing yield gaps through nutrient and water management. Nature, 1–4.

Nichols, J., Kang, S., Post, W., et al., 2011. HPC-EPIC for high resolution simulations of environmental and sustainability assessment. Comput. Electron. Agric. 79, 112–115.

Paskin, N., 2005. Digital Object Identifiers for scientific data. Data Sci. J. 4, 12–20.

Pordes, R., Petravick, D., Kramer, B., Olson, D., Livny, M., Roy, A., Avery, P., Blackburn, K., Wenaus, T., Würthwein, F., Foster, I., Gardner, R., Wilde, M., Blatecky, A., McGee, J., Quick, R., 2007. The open science grid. Scientific Discovery through Advanced Computing (SciDAC) Conference.

Porter, C.H., Villalobos, C., Holzworth, D., Nelson, R., White, J.W., Athanasiadis, I.N., Janssen, S., Ripoche, D., Cufi, J., Raes, D., Zhang, M., Knapen, R., Sahajpal, R., Boote, K.J., Jones, J.W., 2013. Harmonization and translation of crop modeling data to ensure interoperability. Environ. Model. Softw. 62, 495–508.

Portmann, F.T., Siebert, S., Döll, P., 2010. MIRCA2000-Global monthly irrigated and rainfed crop areas around the year 2000: a new high-resolution data set for agricultural and hydrological modeling. Glob. Biogeochem. Cycles 24, Gb1011. http://dx.doi.org/10.1029/2008gb003435.

Potter, P.A., et al., 2010. Characterizing the spatial patterns of global fertilizer application and manure production. Earth Interact. 14 (2), 1–22.

Rosenzweig, C., Jones, J.W., Hatfield, J.L., Ruane, A.C., Boote, K.J., Thorburn, P., Antle, J.M., Nelson, G.C., Porter, C., Janssen, S., Asseng, S., Basso, B., Ewert, F., Wallach, D., Baigorria, G., Winter, J.M., 2013. The agricultural model intercomparison and improvement project (AgMIP): protocols and pilot studies. Agric. For. Meteorol. 170 (0), 166–182.

Rosenzweig, C., Elliott, J., et al., 2014. Assessing agricultural risks of climate change in the 21st century in a global gridded crop model intercomparison. Proc. Natl. Acad. Sci. U.S.A. 111 (9), 3268–3273.

Rutledge, G.K., Alpert, J., Ebisuzaki, W., 2006. NOMADS: a climate and weather model archive at the national Oceanic and Atmospheric Administration. Bull. Am. Meteorol. Soc. 87 (3), 327–341.

Sacks, W.J., Deryng, D., Foley, J.A., Ramankutty, N., 2010. Crop planting dates: an analysis of global patterns. Global Ecol. Biogeogr. 19, 607–620 http://dx.doi.org/10.1111/j.1466-8238.2010.00551.x.

Vital, J.-A., Gaurut, M., Lardy, R., Viovy, N., Soussana, J.-F., Bellocchi, G., Martin, R., 2013. High-performance computing for climate change impact studies with the Pasture Simulation model. Comput. Electron. Agric. 98, 131–135.

Wilde, M., Foster, I., Iskra, K., Beckman, P., Zhang, Z., Espinosa, A., Hategan, M., Clifford, B., Raicu, I., 2009. Parallel scripting for applications at the Petascale and beyond. IEEE Comput. 42 (11), 50–60.

Williams, D.N., Ananthakrishnan, R., Bernholdt, D.E., Bharathi, S., Brown, D., Chen, M., Chervenak, A.L., Cinquini, L., Drach, R., Foster, I.T., Fox, P., Fraser, D., Garcia, J., Hankin, S., Jones, P., Middleton, D.E., Schwidder, J., Schweitzer, R., Schuler, R., Shoshani, A., Siebenlist, F., Sim, A., Strand, W.G., Su, M., Wilhelmi, N., 2009. The Earth system grid: enabling access to multi-model climate simulation data. Bull. Am. Meteorol. Soc. 90 (2), 195–205.

You, Liangzhi, Wood, Stanley, 2006. An entropy approach to spatial disaggregation of agricultural production. Agric. Syst. 90 (1), 329–347.

Zhao, G., Bryan, B.A., King, D., Song, X., Yu, Q., 2012. Parallelization and optimization of spatial analysis for large scale environmental model data assembly. Comput. Electron. Agric. 89, 94–99.

Zhao, Gang, Bryan, Brett A., King, Darran, Luo, Zhongkui, Wang, Enli, Bende-Michl, Ulrike, Song, Xiaodong, Yu, Qiang, 2013. Large-scale, high-resolution agricultural systems modeling using a hybrid approach combining grid computing and parallel processing. Environ. Model. Softw. 41, 231–238.