A Web API for weather generation and pest development simulation in North America

Preprint · March 2022

DOI: 10.13140/RG.2.2.30606.64326

Mathieu Fortin
Natural Resources Canada

Jacques Regniere
Natural Resources Canada

Rémi Saint-Amant
Natural Resources Canada

A Web API for weather generation and pest development simulation in North America

Mathieu Fortin[1], Jean-François Lavoie[2], Jacques Régnière[3], and Rémi Saint-Amant[3]

[1]Canadian Wood Fibre Centre, Canadian Forest Service, Natural Resources Canada, 580 Booth street, Ottawa, ON, Canada
[2]Canadian Wood Fibre Centre, Canadian Forest Service, Natural Resources Canada, 1055 rue du P.E.P.S., Québec, QC G1V 4C7, Canada
[3]Laurentian Forest Centre, Canadian Forest Service, Natural Resources Canada, 1055 rue du P.E.P.S., Québec, QC G1V 4C7, Canada

28 March 2022

Abstract

Climate is an essential component of environmental models. Over the last two decades, many weather generators have been presented in the literature. Although their implementation into software has been of great help to environmental modellers, their lack of integration into modelling frameworks still represents a challenge for end users. In many cases, end users have to retrieve the climate variables by themselves in order to use an environmental model. In some other cases, the weather generator software is embedded into the modelling framework, but this increases the maintenance effort.

In this paper, we present a different approach: the deployment of a weather generator as a Web API. A few application examples are provided to illustrate the benefits of this implementation. In summary, a Web API facilitates the integration into modelling frameworks, decreases the maintenance effort and avoid interoperability issues due to different programming languages.

Keywords: Weather generator; Web API; Client-server architecture; Interoperability; Software maintenance; Software integration

Software availability

Name of software BioSIM Web API and clients
Developers Jean-François Lavoie, Rémi Saint-Amant and Mathieu Fortin
Available Since 2020
Hardware required No specific requirement
Software required Java, C# and R depending on the client
Availability Clients are available on the web site
Website https://github.com/RNCan/BioSimClient_CSharp/wiki
License LGPL v3
Cost Free
Program languages C# for the Web API and C++, R and Java for the clients
Program size Java, C# and R clients (<200 Kb each)

1 Introduction

Climate change has brought the attention of the scientific community on the effect of climate variables in environmental models. Climate-sensitive models are needed, especially in agriculture and forestry, to assess the impacts of climate change on growth and crops. However, the integration of climate variables into environmental models poses two major challenges. First, meteorological time series (MTS) are recorded in weather stations, that is at punctual geographical locations. In contrast, models have a much larger geographical coverage and MTS must be literally available wall to wall. Secondly, in order to assess the impact of climate change, climate projections must be available. These projections are usually produced by general circulation models (GCM) that have a much coarser resolution than what is needed in environmental modelling.

In spite of these two challenges, a significant number of weather generators have been published in the literature over the last two decades (e.g., Bannayan and Hoogenboom, 2008; Birt et al., 2010; Burton et al., 2013). The wall-to-wall availability of MTS has been addressed through the interpolation or extrapolation of the observed MTS at the nearest weather stations (Jolly et al., 2005). As for regional and local climate projections, these are obtained by downscaling the GCM projections (e.g., Wang et al., 2013).

Although there are many weather generators available in the literature, their implementation can cause some issues to ecological modellers and end users. Most weather generators are implemented as standalone applications programmed in various languages (e.g., Bannayan and Hoogenboom, 2008; Kilsby et al., 2007; Liu et al., 2009). For ecological modellers, this usually implies downloading the standalone application, exporting the geographic coordinates of interest from the modelling framework, processing them in the standalone application and importing the MTS back into the modelling framework. When it comes to implementing climate-sensitive ecological models into software and transferring them to end users, the situation can get worse. Unless the weather generator can be embedded into the same software, end users must download a weather generator – hopefully the same that provided the MTS for the model fitting – and generate the MTS by themselves in order to use the ecological model. Obviously, this additional data processing reduces the probability of successful knowledge transfer to end users.

Embedding a weather generator into the same software that implements the ecological model can be a solution. However, the different programming languages often represent a barrier. Moreover, embedding a software into another has some limitations especially when it comes to updating. Unless the embedded weather generator can update itself, modellers will have to publish a new software version containing the updated weather generator. The end users will have to get aware of these new versions and to download them. Having disparate distributed versions increases the maintenance effort and makes the debugging process more complex.

Another option consists of implementing a client-server architecture. A weather generator can be deployed on a server and the software implementing the ecological model can use a client to retrieve the MTS from the server. Embedding a client into an existing software is simpler than embedding a whole weather generator. It takes less space and requires fewer updates. At the same time, the weather generator on the server can be updated at any time ensuring that the clients are always dealing with the latest version.

The objective of this study was to develop this server-client architecture in order to facilitate the use of weather generators for both environmental modellers and end users of environmental models. More specifically, we used an existing weather generator called BioSIM (Régnière et al., 2017), which is currently used in North America, and we deployed it on a server. The server implementing the weather generator uses a Representational state transfer (REST) Application Programming Interface (API) that reads incoming Hypertext transfer protocol (HTTP) requests from the web. We propose several clients in different languages that can easily be integrated into other software. These clients can be used to produce the HTTP requests and parse the incoming results of these requests.

This paper is structured as follows. First, we introduce the BioSIM weather generator and its implementation into a Web API. Then, some examples of applications are presented. Finally, we discuss the advantages and weaknesses of the client-server architecture in the context of weather generation.

2 The BioSIM weather generator

2.1 Geographical coverage by interpolation

BioSIM is a weather generator initially designed to support the planning of pest management activities (Régnière et al., 1995a,b). It has been in continuous development since the mid 1990s and has been used in many research projects focused on forest ecology, forest fire dynamics and insect pest management (e.g., Tobin et al., 2004; Le Goff et al., 2009; Coulombe et al., 2010).

BioSIM defines an MTS as a series of daily values of the four basic climate variables: minimum air temperature (°C), maximum air temperature (°C), precipitation (mm), and wind speed (km h^{-1}). Depending on the weather station and the time period, relative humidity (%) and solar radiation (W m^{-2}) are also available. If they are not, they are simulated using the MT-CLIM 4.3 model (Glassy and Running, 1994; Thornton and Running, 1999). Snow precipitation (mm) and snow depth accumulation (cm) are also simulated using a model adapted from that of Brown et al. (2003).

The weather generator produces an MTS for a particular location by interpolating from the MTS in the k nearest weather stations. The distance for identifying these k nearest stations is computed as a modified Euclidean distance:

$$d_i = \sqrt{(\Delta x_i)^2 + (\Delta y_i)^2 + 100(\Delta z_i)^2 + (\Delta \text{Shore}_i)^2} \tag{1}$$

where Δx_i, Δy_i, Δz_i, and ΔShore_i are the differences (km) in longitude, latitude, elevation and distance to the shore between weather station i and the point of interest, respectively. Note that the factor 100 in Eq. 1 gives a greater weight to the difference in elevation because of the environmental lapse rate, i.e the average temperature decrease with elevation, which is about 6°C km^{-1} (Barry, 2008, p. 52).

The MTS used in the interpolation can be those observed or they can be simulated through the disaggregation of monthly 30-year normals. The first option is mainly used to produce past MTS whereas the second is the usual way of producing future MTS. Note that future 30-year normals are themselves obtained through the downscaling of regional or general circulation model projections. This downscaling is addressed in Section 2.2.

Before interpolation, an adjustment is performed on the MTS of the k nearest weather stations based on local gradients. The local gradients in minimum temperature, maximum temperature and precipitation are evaluated through multiple linear regressions – one for each climate variable – fitted on the 30-year normals of the 25 nearest climate stations based on a distance function similar to Eq. 1. For instance, for maximum temperature, the local gradient would be evaluated through the following linear regression:

$$\text{T}_{\text{max},i} = \beta_0 + \beta_1 \Delta x_i + \beta_2 \Delta y_i + \beta_3 \Delta z_i + \varepsilon_i \tag{2}$$

Δx_i, Δy_i and Δz_i are the differences in longitude, latitude and elevation between weather station i and the point of interest, respectively, and ε_i is the residual error term.

Once the regressions have been fitted, they are used to adjust the minimum temperature, maximum temperature and precipitation from the nearest weather stations to the coordinates of the point of interest. Then, the values of these variables at the point of interest are calculated as the average of the adjusted values in the nearest climate stations, weighted by the inverse squared distance, i.e. $1/d_i^2$.

2.2 Simulating by disaggregation and future climate projections

Regnière and St-Amant (2007) improved an algorithm initially developed by Regnière and Bolstad (1994) with the aim of producing stochastic realizations of daily minimum and maximum air temperatures and precipitation from 30-year monthly normals. The expected daily temperatures for a particular day are interpolated between the means of two successive months. The algorithm uses the standard deviations of these variables, the second order regressive terms and their cross-correlation as observed during the 30-year period to compute a stochastic deviate that is added to the expectation. MTS generated through this disaggregation of 30-year monthly normals were found to reproduce seasonal patterns as well as the variability and extremes of temperature. The predicted precipitation was not quite as accurate but remained realistic (Regnière and St-Amant, 2007).

Simulating by disaggregation is mainly used to produce future MTS. However, it assumes that future 30-year normals are available for each weather station. BioSIM uses the delta method (Mosier et al., 2014), also referred to as the perturbation method by some authors (Prudhomme et al., 2002; Fowler et al., 2007), to scale climate projections from regional or general circulation models down to the weather stations. The 30-year monthly normals for the period 1981-2010 are taken as reference at each weather station. The difference between these 30-year monthly normals and the projections of a regional or a global circulation model for a future 30-year period is the so-called delta. The locally observed 30-year monthly normals are then updated using the delta in order to produce local future 30-year normals that BioSIM relies on to cover the whole 21st century. Future normals were produced using the delta method and climate projections from the Hadley model, the Canadian global circulation model (GCM4) and the Canadian regional circulation model (RCM4) (Collins et al., 2011; von Salzen et al., 2013; Scinocca et al., 2016) under the representative concentration pathways (RCP) 4.5 and 8.5 (IPCC, 2013, p. 29).

3 Software implementation

BioSIM was implemented in C++ as a standalone application for Windows and it is freely available at `ftp://ftp.cfl.forestry.ca/regniere/software/BioSIM/`. The core of the application is a dynamic link library (DLL) implementing the weather generator. This core is complemented by a series of smaller DLLs that make it possible to derive information from the MTS. The annual, monthly and daily degree-days, the soil moisture index and spruce budworm development count among the output of these additional DLLs. For the sake of clarity, we will refer to the core DLL as the generator and to these additional DLLs as MTS-processing models.

The generator must get access to a dataset of 30-year normals to simulate by disaggregation. It can also access a dataset of observed MTS and generate MTS from these. Given the number of weather stations in North America, the weather generator can hardly be instantiated with all the observed MTS. The MTS are rather broken down into smaller synchronous MTS and a different weather generator is instantiated for each synchronous MTS. The same applies to the 30-year normals. Many weather generators are instantiated with different 30-year normals.

3.1 The BioSIM Web API

3.1.1 Implementation

In the context of this project, we developed a REST API based on the .NET framework. The API is implemented as an Internet Information Services (IIS) application which does the following tasks in order (see Fig. 1):

1. Read incoming HTTP requests;
2. Parse the requests and forward them to one or many instances of the generator;
3. Retrieve the MTS generated by the generator instances;
4. Apply one or many MTS-processing models on the MTS;
5. Send the result back to the client.

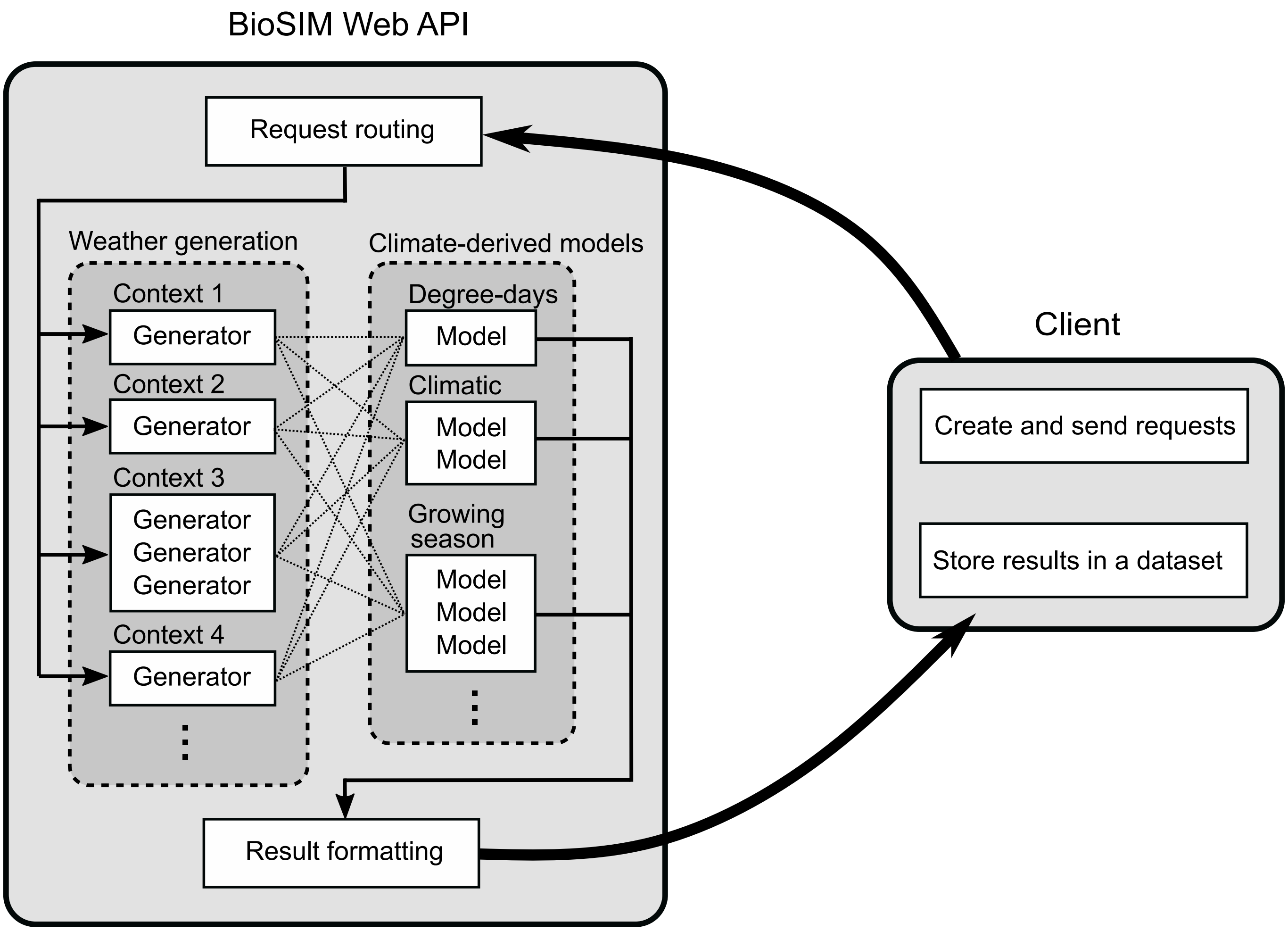

Figure 1: A schematic representation of the BioSIM Web API.

The BioSIM DLLs are embedded in the API as a package. The generator can be loaded with a dataset of 30-year normals and eventually a dataset of daily climate observations. The combination of 30-year normals and daily climate observations is referred to as a context. Upon its initialization, the API creates instances of the generator with different contexts in order to cover the whole 1900-2022 period for observed MTS and as well as the 2022-2100 period for simulation by disaggregation. The list of these contexts is provided in Table 1.

An incoming HTTP request can refer to several contexts. For instance, a request for the period 1976-1986 will deal with generators related to contexts 2 and 3 in Table 1. Upon the uptake of a request, the API determines which contexts are related to the request. For past time series, the range of data within the observed daily MTS defines the time range covered by the context. For example, context 2 covers the 1950-1979 time range. For future time series, the different 30-year normals overlap. The central 10-year period was set as the time range covered by each context associated to future time series. For instance, a request for the period 2041-2060 under RCP 4.5 with the Hadley model would deal with context 7 for the time range 2041-2050 and with context 8 for the time range 2051-2060. The only exception to this is the last context that goes from 2071 to 2100, which was assumed to cover the time range 2081-2100. Once the appropriate contexts have been identified, the API retrieves their associated generators to produce MTS. The MTS from different contexts are concatenated in order to create a single MTS.

Table 1: Different contexts in which the weather generator is instantiated. A context is defined as a combination of 30-year normals and observed daily meteorological time series (MTS).

Context id	Purpose	30-year normals	Observed daily MTS
1	Past time series	Canada-USA 1941-1970	Canada-USA 1900-1949
2		Canada-USA 1951-1980	Canada-USA 1950-1979
3		Canada-USA 1981-2010	Canada-USA 1980-2020
4		Canada-USA 1981-2010	Canada-USA 2021-2022
5	Future time series with Hadley under RCP 4.5	Canada-USA 2011-2040 Hadley RCP 4.5	n/a
6		Canada-USA 2021-2050 Hadley RCP 4.5	n/a
7		Canada-USA 2031-2060 Hadley RCP 4.5	n/a
8		Canada-USA 2041-2070 Hadley RCP 4.5	n/a
9		Canada-USA 2051-2080 Hadley RCP 4.5	n/a
10		Canada-USA 2061-2090 Hadley RCP 4.5	n/a
11		Canada-USA 2071-2100 Hadley RCP 4.5	n/a
12	Future time series with Hadley under RCP 8.5	Canada-USA 2011-2040 Hadley RCP 8.5	n/a
13		Canada-USA 2021-2050 Hadley RCP 8.5	n/a
14		Canada-USA 2031-2060 Hadley RCP 8.5	n/a
15		Canada-USA 2041-2070 Hadley RCP 8.5	n/a
16		Canada-USA 2051-2080 Hadley RCP 8.5	n/a
17		Canada-USA 2061-2090 Hadley RCP 8.5	n/a
18		Canada-USA 2071-2100 Hadley RCP 8.5	n/a
19	Future time series with RCM4 under RCP 4.5	Canada-USA 2011-2040 RCM4 RCP 4.5	n/a
20		Canada-USA 2021-2050 RCM4 RCP 4.5	n/a
21		Canada-USA 2031-2060 RCM4 RCP 4.5	n/a
22		Canada-USA 2041-2070 RCM4 RCP 4.5	n/a
23		Canada-USA 2051-2080 RCM4 RCP 4.5	n/a
24		Canada-USA 2061-2090 RCM4 RCP 4.5	n/a
25		Canada-USA 2071-2100 RCM4 RCP 4.5	n/a
26	Future time series with RCM4 under RCP 8.5	Canada-USA 2011-2040 RCM4 RCP 8.5	n/a
27		Canada-USA 2021-2050 RCM4 RCP 8.5	n/a
28		Canada-USA 2031-2060 RCM4 RCP 8.5	n/a
29		Canada-USA 2041-2070 RCM4 RCP 8.5	n/a
30		Canada-USA 2051-2080 RCM4 RCP 8.5	n/a
31		Canada-USA 2061-2090 RCM4 RCP 8.5	n/a
32		Canada-USA 2071-2100 RCM4 RCP 8.5	n/a
33	Future time series with GCM4 under RCP 4.5	Canada-USA 2011-2040 GCM4 RCP 4.5	n/a
34		Canada-USA 2021-2050 GCM4 RCP 4.5	n/a
35		Canada-USA 2031-2060 GCM4 RCP 4.5	n/a
36		Canada-USA 2041-2070 GCM4 RCP 4.5	n/a
37		Canada-USA 2051-2080 GCM4 RCP 4.5	n/a
38		Canada-USA 2061-2090 GCM4 RCP 4.5	n/a
39		Canada-USA 2071-2100 GCM4 RCP 4.5	n/a
40	Future time series with GCM4 under RCP 8.5	Canada-USA 2011-2040 GCM4 RCP 8.5	n/a
41		Canada-USA 2021-2050 GCM4 RCP 8.5	n/a
42		Canada-USA 2031-2060 GCM4 RCP 8.5	n/a
43		Canada-USA 2041-2070 GCM4 RCP 8.5	n/a
44		Canada-USA 2051-2080 GCM4 RCP 8.5	n/a
45		Canada-USA 2061-2090 GCM4 RCP 8.5	n/a
46		Canada-USA 2071-2100 GCM4 RCP 8.5	n/a

The API has several routes (Table 2). The main route is `BioSimWeather`, which does the weather generation and processes the generated MTS using one or many user-specified MTS-processing models (see Table 3). An example of HTTP request for the annual degree-days over the 2000-2016 period at the eastern end of the Reservoir Gouin (48° 30’ N, 74° 30’ W) in the province of Quebec, Canada, using the main route is:

http://repicea.dynu.net/BioSim/BioSimWeather?lat=48.5&long=-74.5&from=2000&to=2016&model=DegreeDay_Annual&format=JSON

The result is returned in the JavaScript object notation (JSON) format. Multiple MTS can be generated by specifying more than one points in the HTTP request. For example, two MTS are generated at once in the following request:

http://repicea.dynu.net/BioSim/BioSimWeather?lat=48.5%2049.0&long=-74.5%20-76.0&from=2000&to=2016&model=DegreeDay_Annual&format=JSON

Table 2: Routes of the BioSIM Web API. The service is hosted on http://repicea.dynu.net. The complete documentation is available at https://sourceforge.net/p/mrnfforesttools/biosimclient/wiki/BioSIM%20Web%20API%20Documentation/.

Route	Description
`BioSimWeather`	Generate meteorological time series and perform analysis using one or many models
`BioSimNormals`	Provide normals for a given 30-year period
`BioSimModelList`	Provide the list of models for the analysis of meteorological time series
`BioSimModelHelp`	Provide information about a given model
`BioSimModelDefaultParameters`	Provide the parameters of a particular model and their default values

Note that the API allows for a maximum of 10 points per HTTP request. However, the clients described in Section 3.2 can handle a greater number of points by sending multiple requests of 10 points each. The results of the multiple requests are concatenated into a single output. The other routes shown in Table 2 provide further information as to what MTS-processing models are available and how to change their parameters.

The API also implements an automatic update feature. Every day, a new version of the 2021-2022 observed daily MTS dataset is compiled with the latest meteorological observations and it is made available on a dedicated FTP site. The API periodically checks if the version on the FTP site is more recent than the one used in Context 4 (Table 1). Whenever this is the case, the new version of the dataset is downloaded and loaded in place of the previous version. This feature ensures that the Web API always relies on the latest observed MTS.

3.1.2 Scalability and performance

The C# language allows for multithreading. Given that the API accepts several points per request, this feature makes it possible to increase performance. It is possible to instantiate more than one generator for each context listed in Table 1. The different geographical points are then sent into a queue which feeds the different generator instances, which in turn return the MTS to the main thread. The same implementation also applies to MTS-processing models (see Table 3).

Table 3: MTS-processing models by category and topic in the BioSIM Web API.

Variable category and topic	Model names
Climate variables	
Evapotranspiration	`ASCE_ETc_Daily`, `ASCE_ETcEx_Daily`, `ASCE_ETsz_Daily`, `Potential_Evapotranspiration_Annual`, `Potential_Evapotranspiration_Daily`, `Potential_Evapotranspiration_Monthly`, `Standardised_Precipitation_Evapotranspiration_Index`, `Potential_Evapotranspiration_Ex_Annual`, `Potential_Evapotranspiration_Ex_Daily`, and `Potential_Evapotranspiration_Ex_Monthly`
Bud burst	`BudBurst`
Collections of climate variables	`CCBio_Annual`, `CCBio_Monthly`, `Climatic_Annual`, `Climatic_Monthly`, `Climatic_Daily`, `ClimaticEx_Daily`, `ClimaticQc_Annual`, and `TminTairTmax_Daily`
Wind speed and direction	`ClimaticWind_Annual`, `ClimaticWind_Monthly`
Moisture indices	`Climate_Moisture_Index_Monthly`, `Climate_Mosture_Index_Annual`, `Soil_Moisture_Index_Annual`, `Soil_Moisture_Index_Daily`, `Soil_Moisture_Index_Monthly`, `Soil_Moisture_Index_QL_Annual`, `Soil_Moisture_Index_QL_Daily`, and `Soil_Moisture_Index_QL_Monthly`
Degree-days	`DegreeDay_Annual`, `DegreeDay_Monthly`, `DegreeDay_Daily`, `ReverseDegreeDay_Annual`
Growing season	`GrowingSeason`
Snow	`SnowMelt_Monthly`
Vapor pressure	`VaporPressureDeficit_Annual`, `VaporPressureDeficit_Daily`, and `VaporPressureDeficit_Monthly`
Miscellaneous	
Fire indices	`FWI_Annual`, `FWI_Monthly`, `FWI_Daily`, `FWI_Drought_Code_Monthly`, `FWI_Drought_Code_Daily`, `FWI_Drought_Code_Fixe_Monthly`, and `FWI_Drought_Code_Fixe_Daily`
Plant hardiness	`PlantHardinessCanada` and `PlantHardinessUSA`

Although a multithreading approach seemed promising, preliminary trials showed that the marginal gain in adding new threads quickly decreased. In fact, there was little gain or no gain at all in performance when increasing the number of threads from five to 10. This was due to the memory allocation inside the threads. Because the API remained a single-process application, the memory allocation into the different threads is protected by locks which reduce the performance (Embedded Staff, 2006), especially when the threads must allocate large chunks of memory, which was the case here.

To work around this memory issue, we implemented a multiprocess architecture. The different generators and MTS-processing models were instantiated in different processes and linked to the main process, that of the API, through pipes. Because each process is independent and contains a single generator or MTS-processing instance, there is no constraint due to the locks associated to memory allocation. Preliminary results with this architecture showed that marginal gain in performance when adding new processes remained significant.

The performance of the Web API was tested with 100 fictive requests sent on the `BioSimWeather` route. Each request had a random time span between 1950 and 2050, up to 100 locations and

Table 3 (cont.): MTS-processing models by category and topic in the BioSIM Web API. The models denoted by an asterisk are experimental.

Category and topic	Model names
Insects and diseases	
Blue-stain fungus (*Ceratocystis polonica* Moreau)	`BlueStainVariables`
Emerald ash borer (*Agrilus planipennis* Fairmaire)	`EmeraldAshBorer` and `EmeraldAshBorerColdHardiness_Annual`
European elm scale (*Eriococcus spurius* Modeer)	`EuropeanElmScale`*
Fall cankerworm (*Alsophila pometaria* Harris)	`FallCankerworms`*
Gypsy moth (*Lymantria dispar dispar* Linnaeus)	`Gypsy_Moth_Seasonality`
Hemlock looper (*Lambdina fiscellaria* Guenée)	`HemlockLooper`
Hemlock Woolly Adelgid (*Adelges tsugae* Annand)	`HemlockWoollyAdelgid_Annual` and `HemlockWoollyAdelgid_Daily`
Jackpine budworm (*Choristoneura pinus* Freeman)	`Jackpine_Budworm`
Tooth-necked fungus beetle *Laricobius nigrinus* Fender	`LaricobiusNigrinus`*
Mountain pine beetle *Dendroctonus ponderosae* Hopkins	`MPB_Cold_Tolerance_Annual`, `MPB_Cold_Tolerance_Daily` and `MPB_SLR`
Oblique banded leaf roller (*Choristoneura rosaceana* Harris)	`ObliqueBandedLeafroller`
Spring cankerworm (*Paleacrita vernata* Peck)	`SpringCankerworms`*
Eastern spruce budworm (*Choristoneura fumiferana* Clemens)	`Spruce_Budworm_Biology_Annual` and `Spruce_Budworm_Biology`
Spruce beetle (*Dendroctonus rufipennis* Kirby)	`SpruceBeetle`
Western spruce budworm (*Choristoneura freemani* Razowski)	`Western_Spruce_Budworm_annual` and `Western_Spruce_Budworm`
White-marked tussock moth (*Orgyia leucostigma* Smith)	`WhitemarkedTussockMoth`*
White pine weevil (*Pissodes strobi* Peck)	`WhitePineWeevil`*
Yellow-headed spruce sawfly (*Pikonema alaskensis* Rohwer)	`Yellowheaded_Spruce_Sawfly`

randomly asked for either the annual degree days or an extensive array of annual climatic variables. These requests were sent to the Web API running on an Intel(R) Xeon(R) CPU E5-1650 v3 @ 3.50GHz with the generators and MTS-processing models all being instantiated in three processes. The average time to process a single point in each request is shown in Fig. 2. By regression, the average processing time was estimated at 28.8 and 11.4 milliseconds point^{-1} yr^{-1} for the extensive array of climate variables and for the degree-days, respectively.

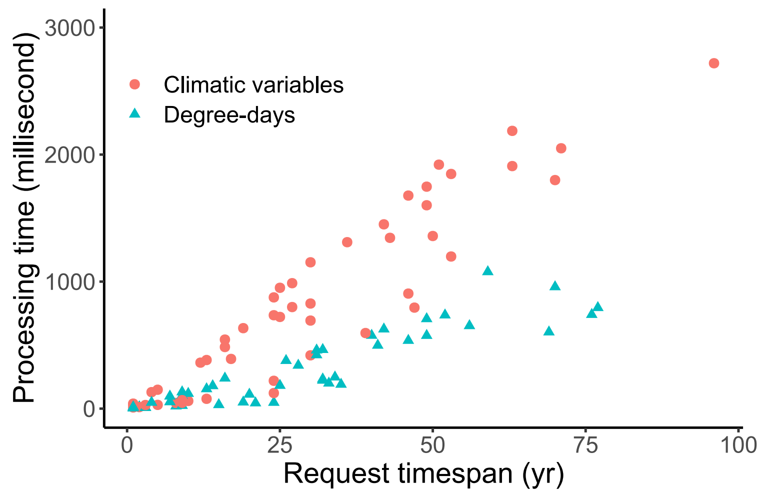

Figure 2: Average time to process a single point in a hundred fictive requests with different time spans.

3.2 C#, Java and R clients

Client applications are available in the C#, Java and R languages. They offer simple static methods that produce the HTTP requests for the Web API (Table 4). The R client actually uses the Java client as back end and depends on the J4R package (see Fortin, 2020) for the interoperatility between R and Java. The R client exposes functions that bear the same names as the static methods of the Java client (Table 4). The C# client is a direct translation of the Java client. Using one of the three clients has the advantage of breaking requests with a large number of points into smaller requests and concatenating the results, which are then stored into an instance that provide accessors to easily retrieve the climate variables.

Table 4: Methods and functions exposed in the C#, Java and R clients. Given the naming convention in C#, the methods bear the same names but start with an uppercase letter.

Method or function	Description
`generateWeather`	Generate meteorological time series and analyze them using a particular model
`getNormals`	Provide monthly normals, which can be aggregated over some months
`getAnnualNormals`	Provide annual normals
`getMonthlyNormals`	Provide monthly normals without any aggregation
`getModelList`	Provide the list of available models
`getModelHelp`	Provide the information about a particular model
`getModelDefaultParameters`	Provide the default parameter values for a particular model

4 Application examples

To illustrate the potential applications of the Web API, we provide four examples. The first three are based on the R client whereas the last one uses the Java client. The R scripts to reproduce the first three examples can be found in the Supporting Information.

4.1 Spring frost and tomato seedlings

Tomato seedlings are sensitive to spring frost. Damage can occur at temperatures below -2°C (Drown, 1985). Therefore, it would be helpful to identify the first calendar day for which the minimum temperature remained consistently above this threshold over the last 30 years. Let us focus on a location close to Ottawa, Canada (45°25' N, 76°01' W). Using the R client, we can easily retrieve the minimum temperature from early May to the end of June for all the years from 1991 to 2021 and plot them against the calendar day (Fig. 3). It turns out that the minimum temperature never went below -2°C after May 29 during the last 30 years and this date could be considered as safe to avoid spring frost when planting tomato seedlings outdoors, although other factors will likely affect the subsequent growth and harvest.

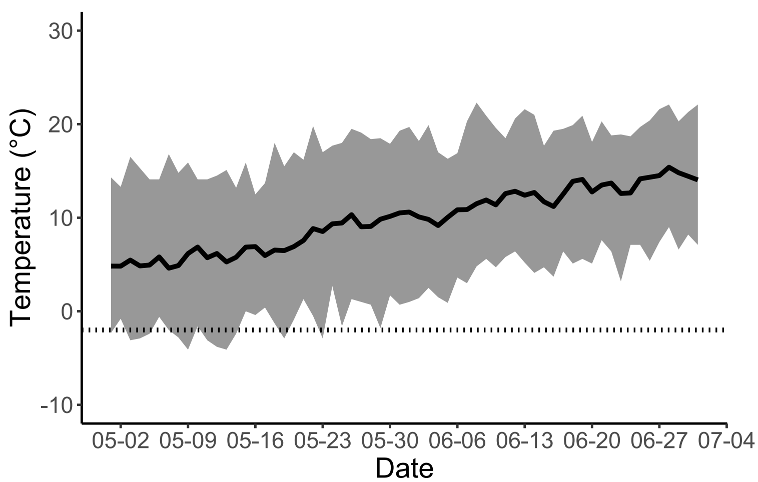

Figure 3: Minimum daily temperatures near Ottawa, Canada, from May 1st to July 1st over the 1991-2021 period. The solid line represents the 30-year average minimum temperature. The shaded area delineates the maximum and minimum values observed over the period. The dotted line represents the -2°C threshold for frost damage.

4.2 Spruce budworm phenology

Spruce budworm is a major defoliator in the province of Quebec, Canada. It feeds mainly on balsam fir (*Abies balsamea* (L.) Mill.) and white spruce (*Picea glauca* (Moench) Voss), and to a lesser extent on red spruce (*Picea rubens* Sarg.) and black spruce (*Picea mariana* (Mill.) BSP). Outbreaks occur approximately once every 40 years (Boulanger and Arsenault, 2004). During outbreaks, four consecutive years of moderate to severe defoliation significantly increase the mortality of balsam fir trees (Pothier and Mailly, 2006).

Spruce budworm larvae emerge in late April to mid-May. Their population dynamics depend on many factors including foliage dynamics as well. Régnière (1982) developed a process-oriented model to predict the seasonal development of spruce budworm populations. This model, which has undergone several improvements since its initial publication (Régnière et al., 2012), was implemented as a MTS-processing model in BioSIM and it is part of the Web API.

Montmorency Experimental Forest (47°19' N, 71°09' W) is located north of Quebec city, Canada. This area is dominated by one of the host species, namely balsam fir. The predicted proportion of the population that reaches different successive development stages during spring and summer 2023 can be obtained by generating the MTS for this location and processing it using the MTS-processing model (Fig. 4). Less than 10% of the initial population of overwintering larvae survive to the adult stage.

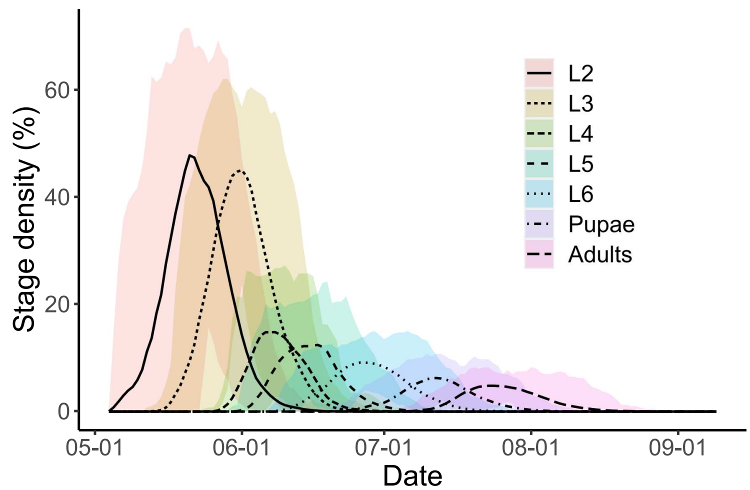

Figure 4: Predicted proportion of the spruce budworm population in the successive development stages during spring and summer 2023 at Montmorency Experimental Forest. The shaded areas delineate the minimum and maximum predicted values for each stage. L2-L6 represent the successive larval stages, before they turn into pupae and finally adults.

4.3 Stochastic predictions of maximum temperature

Heat waves are a concern for some populations as they are associated with health issues (Patz et al., 2014). We simulated the annual highest temperature over the 2022-2100 period for the municipality of Chibougamau, Canada (49°56' N, 74°23' W). The weather generation relies on the stochastic disaggregation of 30-year normals. Consequently, generating several MTS gives a better idea of the variability around the mean trend. In this example we produced 10 realizations of MTS for this period under two future climate scenarios: RCP 4.5 and RCP 8.5 (Fig. 5). The simulation shows that annual maximum temperature peaked at 35°C after 2060 under RCP 4.5. In contrast, peaks as high as 37°C could be reached after 2060 under RCP 8.5.

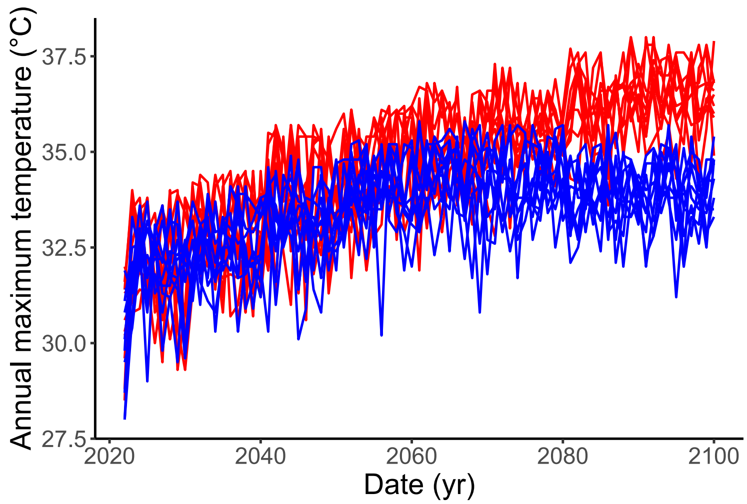

Figure 5: Ten realizations of annual maximum temperatures near Chibougamau, Canada, over the 2022-2100 period under two representative concentration pathways (RCP), 4.5 (blue) and 8.5 (red).

4.4 Integration of climate variables into forest growth simulation

CAPSIS is a Java-programmed platform for forest growth simulators (Dufour-Kowalski et al., 2012). One of its simulators is currently used to support forest management planning on publicly-owned forest lands in the province of Quebec. This simulator, ARTEMIS, follows the typical architecture of individual-based models in forestry (Porté and Bartelink, 2002): it predicts tree mortality, the growth of survivors and the recruits of new individuals. All these submodels were statistically fitted to a large dataset of permanent-plot data, some of which were sensitive to 30-year temperature and precipitation normals (Fortin and Langevin, 2012). These 30-year normals were produced by BioSIM.

The CAPSIS platform aims at providing user-friendly growth simulators. Assuming that the users will download BioSIM and retrieve the 30-year normals by themselves before simulating with ARTEMIS will certainly discourage many users, without mentioning the possible errors in manipulating the data. This issue was raised in the introduction of this paper. To avoid such a situation, the Java client was embedded in the CAPSIS platform. Upon the initialization of the ARTEMIS simulator, the geographical coordinates of the points of interest are sent to the Web API which sends the 30-year temperature and precipitation normals back to the CAPSIS platform (Fig. 6). The large-area growth projections in Melo et al. (2019) and Fortin et al. (2021) relied on the Java client to retrieve the climate variables before running the simulations with ARTEMIS.

We used a random plot from the fourth campaign of Quebec's provincial forest inventory. Plot 1507602703 is located in a forest management unit near the municipality of Chibougamau. It was measured in 2015 following the inventory protocol (MFWP, 2016): it covers an area of 400 m^2 in which all trees with diameter at breast height (1.3 m in height) greater than 9 cm were tallied. Black spruce was the dominant species, accompanied by jack pine (*Pinus banksiana* Lamb.) in a smaller proportion.

This plot was imported in the CAPSIS platform and its growth was simulated with the

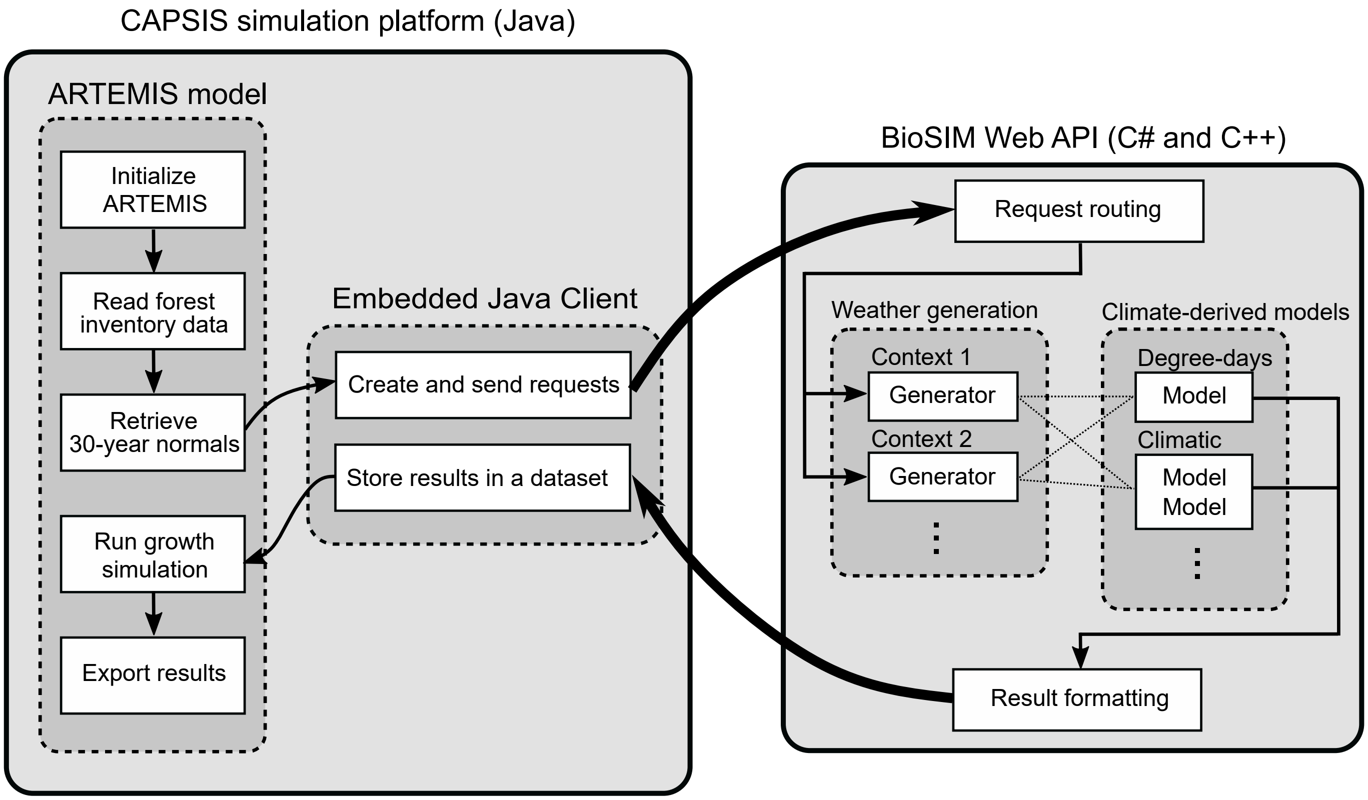

Figure 6: Integration of the Java client in the flowchart of the ARTEMIS forest growth model hosted on the CAPSIS simulation platform.

ARTEMIS simulator for the 2015-2105 period under three climate scenarios: no change, RCP 4.5 and RCP 8.5. The simulation showed that this plot has a greater all-species standing volume at the end of the 21st century under RCP 4.5 and 8.5 compared to the no change scenario (Fig. 7). This simulation does not account for any natural or human-made disturbance though.

5 Discussion and Conclusions

There are numerous weather generators available from the literature in addition to those we already mentioned (e.g., Ivanov et al., 2007; Semenov, 2008; Wang et al., 2016; Peleg et al., 2017). All these weather generators share the same ideas of downscaling global projections and spatially interpolating MTS. The methods used to perform these two tasks differ across the weather generators. For instance, many downscaling techniques exist and they can be classified into four categories: regression methods, weather pattern approaches, stochastic weather generators, and limited-area climate models (Wilby and Wigley, 1997). They all have limitations that can affect the reliability of the predictions in some contexts (Wilby et al., 2002). Our objective was not to present the pros and cons of BioSIM with respect to other weather generators. It can be said that BioSIM has advantages and limitations compared to other simulators.

Our focus was more on how weather generators can become more accessible and better integrated into a work flow. There are plenty of examples of weather generators being used to provide MTS as inputs for models in environmental sciences and in particular in forestry and hydrology

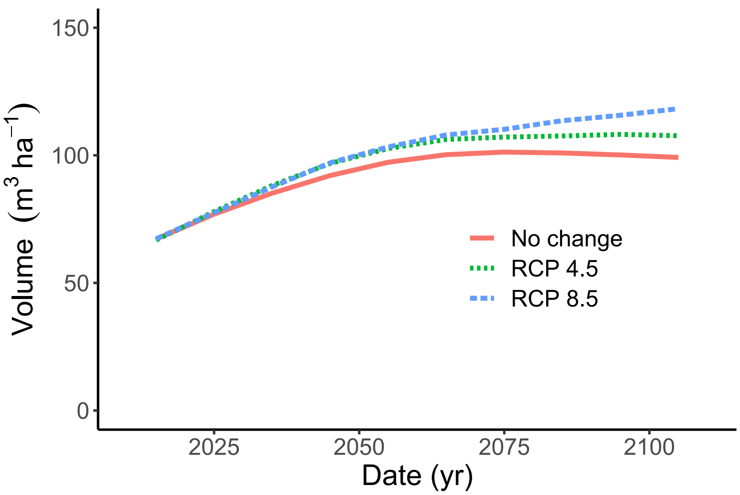

Figure 7: Predicted volume growth in plot 1507602703 under three climate scenarios: no change, RCP 4.5 and RCP 8.5.

(e.g., Le Goff et al., 2009; Fatichi et al., 2011; Schlabing et al., 2014; Melo et al., 2019). Efforts have been made to provide user-friendly access to these weather generators through standalone or web applications (e.g., Schlabing et al., 2014; Wang et al., 2016). However, to the best of our knowledge, these always imply additional steps in the work flow, namely saving the MTS generated by the weather generator to disk and reimporting them into the simulation framework.

Our solution to avoid these additional steps was to implement the weather generator into a Web API and to provide some clients that can be used for a proper integration into a simulation framework. Our application examples show that climate variables can easily be retrieved in R and Java. Acknowledging that R is a popular language (TIOBE, 2022), and especially in the scientific community, our Web API makes it easy to integrate BioSIM into the work flow of an R script.

In addition to facilitating the use of a weather generator, such a Web API has at least two other benefits. First, it ensures that the users are always using the latest version of the weather generator. As we pointed out in the introduction, the Web API avoids version conflicts since the clients are always connected to the latest version of BioSIM, namely that embedded in the Web API. Secondly, it also facilitates the interoperability between the languages since the communication between the client and the Web API is based on character strings. Our application example on the integration of climate variables into forest growth simulation is a good example of interoperability (Fig. 6). As long as the client can produce HTTP requests and parse the results, the language does not matter. For instance, the language of two of our clients is different from that of the Web API and this did not cause any problem. REST APIs such as the one shown in this study have become a standard for interoperability between languages and devices (Di Martino et al., 2017).

The downside of a weather generator Web API is the computational burden. Generating MTS is not trivial. Depending on the time span and the computer CPU speed, BioSIM can take up to half a second to produce an MTS for a single point. Having the weather generator on a server means that there is only one application running for potentially many clients. Having many clients sending multiple requests at the same time is likely to increase the computational time. We managed to control this delay by limiting the number of points in a single request to 10 and by having a multiprocess implementation on the Web API side. However, each process instantiates a generator or a MTS-processing model and these instances, especially the generators, require a share of RAM. This results in a greater RAM utilisation. Moreover, a multi-process implementation requires the CPU to have a correspondingly larger number of cores. As a consequence, the deployment of a weather generator Web API requires a greater capacity than that of most personal computers.

Another limitation of such a Web API is that it does not allow for rollback to a previous version of the BioSIM application. By having the application on a server and making it accessible through a Web API, we ensure that the users are all using the latest version. However, some users might want to keep using a particular version of the BioSIM application for their own reasons. This is impossible with the current Web API. A possible workaround consists of having different Web API, one for each version of the BioSIM application, but this would imply additional resources. Given that the number of users interested in previous versions of BioSIM is limited, we do not intent to create Web API for these versions. This could be envisaged if a new version of BioSIM was to be released.

We acknowledge that BioSIM has weaknesses just as any other weather generator. Depending on the context, this weather generator can be better or worse than some other weather generator. In order to avoid the pitfall of individual models, it is recommended to use multimodel ensembles instead of a single model, especially when generating regional and global climate projections (Krishnamurti et al., 2000). Following this line of thought, we strongly encourage the implementation of other weather generators into Web API like ours, so that end-users can get access to a large array of weather generators within their simulation framework.

Acknowledgements

The authors are grateful to the Data Hub team (Natural Resources Canada) for the support in the migration of a preliminary version of the Web API into the .NET environment. This project was funded by the Canadian Forest Service through its internal scientific program "Developing Sustainable Fibre Solutions".

Abbreviations

API: Application programming interface
CPU: Central processing unit
DLL: Dynamic link library
FTP: File transfer protocol
GCM: General circulation model
HTTP: Hypertext transfer protocol
IIS: Internet information services
JSON: JavaScript object notation
MTS: Meteorological time series
RAM: Random access memory
RCP: Representative concentration pathway
REST: Representational state transfer

References

Bannayan, M. and Hoogenboom, G. (2008). Weather analogue: A tool for real-time prediction of daily weather data realizations based on a modified k-nearest neighbor approach. *Environmental Modelling & Software*, 23:703–713.

Barry, R. G. (2008). *Mountain Weather and Climate. Third Edition.* Cambridge University Press, Cambridge, UK.

Birt, A. G., Valdez-Vivas, M. R., Feldman, R. M., Lafon, C. W., Cairns, D., Coulson, R. N., Tchakarian, M., Xi, W., and Guldin, J. M. (2010). A simple stochastic weather generator for ecological modeling. *Environmental Modelling & Software*, 25:1252–1255.

Boulanger, Y. and Arsenault, D. (2004). Spruce budworm outbreaks in eastern Quebec over the last 450 years. *Canadian Journal of Forest Research*, 34:1035–1043.

Brown, R. D., Brasnett, B., and Robinson, D. (2003). Gridded North American monthly snow depth and snow water equivalent for GCM evaluation. *Atmosphere-Ocean*, 41(1):1–14.

Burton, A., Glenis, V., Jones, M. R., and Kilsby, C. G. (2013). Models of daily rainfall cross-correlation for the United Kingdom. *Environmental Modelling & Software*, 49:22–33.

Collins, W. J., Bellouin, N., Doutriaux-Boucher, M., Gedney, N., Halloran, P., Hinton, T., Hugues, J., Jones, C. D., Joshi, M., Liddicoat, S., Martin, G., O'Connor, F., Rae, J., Senior, C., Sitch, S., Totterdell, I., Wiltshire, A., and Woodward, S. (2011). Development and evaluation of an Earth-System model - HadGEM2. *Geoscientific Model Development*, 4:1051–1075.

Coulombe, S., Bernier, P. Y., and Raulier, F. (2010). Uncertainty in detecting climate change impact on the projected yield of black spruce (*Picea mariana*). *Forest Ecology and Management*, 259:730–738.

Di Martino, B., Esposito, A., Maisto, S. A., and Nacchia, S. (2017). A semantic IoT framework to support RESTful devices' API interoperability. In *IEEE 14th International Conference on Networking, Sensing and Control (ICNSC)*, pages 78–83.

Drown, G. T. (1985). Factors affecting ice nucleation in tomato plants. Master's thesis, Oklahoma State University.

Dufour-Kowalski, S., Courbaud, B., Dreyfus, P., C., M., and de Coligny, F. (2012). Capsis: an open software framework and community for forest growth modelling. *Annals of Forest Science*, 69:221–233.

Embedded Staff (2006). Tackling memory allocation in multicore and multithreaded applications. Embedded. Technical article. Available online at www.embedded.com/tackling-memory-allocation-in-multicore-and-multithreaded-applications/.

Fatichi, S., Ivanov, V. Y., and Caporali, E. (2011). Simulation of future climate scenarios with a weather generator. *Advances in Water Resources*, pages 448–467.

Fortin, M. (2020). Executing native Java code in R: an approach based on a local server. *PeerJ Computer Science*, 6:e300.

Fortin, M. and Langevin, L. (2012). Stochastic or deterministic single-tree models: is there any difference in growth predictions? *Annals of Forest Science*, 69(2):271–282.

Fortin, M., Sattler, D., and Schneider, R. (2021). An alternative simulation framework to evaluate the sustainability of annual harvest on large forest estates. *Canadian Journal of Forest Research*, Submitted.

Fowler, H. J., Blenkinsop, S., and Tebaldi, C. (2007). Linking climate change modelling to impacts studies: recent advances in downscaling techniques for hydrological modelling. *International Journal of Climatology*, 27:1547–1578.

Glassy, J. M. and Running, S. W. (1994). Validating diurnal climatology logic of the MT-CLIM model across a climatic gradient in Oregon. *Ecological Applications*, 4(2):248–257.

IPCC (2013). *Climate Change 2013: the Physical Science Basis. Working Group I Contribution to the Fifth Assessment Report of the International Panel on Climate Change.* Cambridge University Press, Cambridge, United Kingdom.

Ivanov, V. Y., Bras, R. L., and Curtis, D. C. (2007). A weather generator for hydrological, ecological, and agricultural applications. *Water Resources Research*, 43(10):W10406.

Jolly, W. M., Graham, J. M., Michaelis, A., Nemani, R., and Running, S. W. (2005). A flexible, integrated system for generating meteorological surfaces derived from point sources across multiple geographic scales. *Environmental Modelling & Software*, 20:873–882.

Kilsby, C. G., Jones, P. D., Burton, A., Ford, A. C., Fowler, H. J., Harpham, C., James, P., Smith, A., and Wilby, R. L. (2007). A daily weather generator for use in climate change studies. *Environmental Modelling & Software*, 22:1705–1719.

Krishnamurti, T. N., Kishtawal, C. M., Zhang, Z., LaRow, T., Bachiochi, D., Williford, E., Gadgil, S., and Surendran, S. (2000). Multimodel ensemble forecasts for weather and seasonal climate. *Journal of Climate*, 13(23):4196–4216.

Le Goff, H., Flannigan, M. D., and Bergeron, Y. (2009). Potential changes in monthly fire risk in the eastern Canadian boreal forest under future climate change. *Canadian Journal of Forest Research*, 39:2369–2380.

Liu, J., Williams, J. R., Wang, X., and Yang, H. (2009). Using MODAWEC to generate daily weather data for the EPIC model. *Environmental Modelling & Software*, 24:655–664.

Melo, L. C., Schneider, R., and Fortin, M. (2019). The effect of natural and anthropogenic disturbances on the uncertainty of large-area forest growth forecasts. *Forestry*, 92:231–241.

MFWP (2016). *Norme d'inventaire écoforestier. Placettes-échantillons temporaires. Edition 2016.* Government of Quebec, Ministry of Forests, Wildlife and Parks.

Mosier, T. M., Hill, D. F., and Sharp, K. V. (2014). 30-Arcsecond monthly climate surfaces with global land coverage. *International Journal of Climatology*, 34:2175–2188.

Patz, J. A., Frumkin, H., Holloway, T., Vimont, D. J., and Haines, A. (2014). Climate change: Challenge and opportunities for global health. *JAMA*, 312(15):1565–1580.

Peleg, N., Fatichi, S., Paschalis, A., Molnar, P., and Burlando, P. (2017). An advanced stochastic weather generator for simulating 2-D high-resolution climate variables. *Journal of Advances in Modeling Earth Systems*, 9:1595–1627.

Porté, A. and Bartelink, H. H. (2002). Modelling mixed forest growth: a review of models for forest management. *Ecological Modelling*, 150:141–188.

Pothier, D. and Mailly, D. (2006). Stand-level prediction of balsam fir mortality in relation to spruce budworm defoliation. *Canadian Journal of Forest Research*, 36:1631–1640.

Prudhomme, C., Reynard, N., and Crooks, S. (2002). Downscaling of global climate models for flood frequency analysis: where are we now? *Hydrological Processes*, 16:1137–1150.

Régnière, J. (1982). A process-oriented model of spruce budworm phenology (lepidoptera: Tortricidae). *The Canadian Entomologist*, 114(9):811–825.

Regnière, J. and Bolstad, P. (1994). Statistical simulation of daily air temperature patterns in eastern North America to forecast seasonal events in insect pest management. *Environmental Entomology*, 23(6):1368–1380.

Régnière, J., Cooke, B., and Bergeron, V. (1995a). BioSIM: A computer-based decision support tool for seasonal planning of pest management activities. Information report lau-x-116, Natural Resources Canada, Canadian Forest Service, Laurentian Forestry Centre.

Régnière, J., Duval, P., and Saint-Amant, R. (2012). Predicting insect distribution under climate change from physiological responses: spruce budworm as an example. *Biological Invasions*, 14(8):1571–1586.

Régnière, J., Lavigne, D., and Staples, A. (1995b). Analyse de la performance de BioSIM, un outil de planification saisonnière de la lutte antiparasitaire, au Nouveau-Brunswick, en 1992 et 1993. Information report lau-x-115f, Natural Resources Canada, Canadian Forest Service, Laurentian Forestry Centre.

Régnière, J., Saint-Amant, R., Béchard, A., and Moutaoufik, A. (2017). BioSIM 11 – User's manual. Update of Information Report LAU-X-137, Natural Resources Canada, Canadian Forest Service, Laurentian Forestry Centre.

Regnière, J. and St-Amant, R. (2007). Stochastic simulation of daily air temperature and precipitation from monthly normals in North America north of Mexico. *International Journal of Biometeorology*, 51:415–430.

Schlabing, D., Frassi, M. A., Eder, M. M., Rinke, K., and Bárdossy, A. (2014). Use of a weather generator for simulating climate change effects on ecosystems: A case study on Lake Constance. *Environmental Modelling & Software*, 61:326–338.

Scinocca, J. F., Kharin, V. V., Jiao, Y., Qian, M. W., Lazare, M., Solheim, L., and Flato, G. M. (2016). Coordinated global and regional climate modeling. *Journal of Climate*, 29:17–35.

Semenov, M. A. (2008). Simulation of extreme weather events by a stochastic weather generator. *Climate Research*, 35:203–212.

Thornton, P. E. and Running, S. W. (1999). An improved algorithm for estimating incident daily solar radiation from measurements of temperature, humidity, and precipitation. *Agricultural and Forest Meteorology*, 93:211–228.

TIOBE (2022). TIOBE index for January 2022. Available online at https://www.tiobe.com/tiobe-index/. Accessed on January 11, 2022.

Tobin, P. C., Sharov, A. A., Liebhold, A. A., Leonard, D. S., Roberts, E. A., and Learn, M. R. (2004). Management of the gypsy moth through a decision algorithm under the STS project. *American Entomologist*, 50(4):200–209.

von Salzen, K., Scinocca, J. F., McFarlane, N. A., Li, J., Cole, J. N. S., Plummer, D., Verseghy, D., Reader, M. C., Ma, X., Lazare, M., and Solheim, L. (2013). The Canadian fourth generation atmospheric global climate model (CanAM4). Part i: Representation of physical processes. *Atmosphere-Ocean*, 51(1):104–125.

Wang, T., Hamann, A., Spittlehouse, D., and Carroll, C. (2016). Locally downscaled and spatially customizable climate data for historical and future periods for North America. *PLoS ONE*, 11(6):e0156720.

Wang, X., Huang, G., Lin, Q., Nie, X., Cheng, G., Fan, Y., Li, Z., Yao, Y., and Suo, M. (2013). A stepwise cluster analysis approach of downscaled climate projection – A Canadian case study. *Environmental Modelling & Software*, 49:141–151.

Wilby, R. L., Dawson, C. W., and Barrow, E. M. (2002). SDSM – a decision support tool for the assessment of regional climate change impacts. *Environmental Modelling & Software*, 17:147–159.

Wilby, R. L. and Wigley, T. M. L. (1997). Downscaling general circulation model output: a review of methods and limitations. *Progress in Physical Geography*, 21(4):530–548.