



Native NetCDF climate forcing in SWAT+ 62: implementation, numerical equivalence, and input performance

Celray James Chawanda^{*1,2}, Jeffrey Arnold¹, Raghavan Srinivasan¹, Olaf David³, and Jose Pablo Teran Orsini²

5 ¹Texas A&M AgriLife Research, 720 East Blackland Road, Temple, Texas 76502, USA

²Department of Water and Climate, Vrije Universiteit Brussel, Pleinlaan 2, 1050 Ixelles, Belgium

³Department of Civil and Environmental Engineering, Colorado State University, Fort Collins, Colorado 80523, USA

*Correspondence: Celray James Chawanda (celray.chawanda@tamu.edu)

Abstract. Large-scale hydrological models need climate forcing at thousands of locations. The Soil and Water Assessment
10 Tool Plus (SWAT+) stores measured weather as separate text files for each station and variable, so file counts, storage and
initialisation time grow with each station. Existing NetCDF tools for SWAT+ prepare data outside the model and still write
these text files. Reading one compressed dataset inside the model would remove this burden without changing how SWAT+
simulates hydrology. We developed a native NetCDF-4 pathway and a converter for existing station files, and tested the
15 pathway's effect on model results, storage and input time. The reader runs at initialisation and fills the existing climate
arrays, so the daily process code is unchanged; one entry in file.cio selects it, and existing text-input projects run as before.
Paired runs of an uncalibrated Zambezi River basin model with 1,148 stations over 1980–1985 produced identical output
files, and all 15,098,496 daily forcing values matched at 32-bit precision. At 700 stations, the NetCDF inputs needed 83.84
% less storage (235.6 MB against 1,457.6 MB). In the fixed-domain, warm-cache benchmark, process time through
20 completion of climate loading was 5.8 s for NetCDF and 53.6 s for text; text was faster only at 5 stations. Each test used one
compiler and one machine; the reader expects the converter's schema and loads the prepared domain into memory. Because
the saving recurs in every run, calibration, sensitivity analysis and ensemble studies of regional to global SWAT+ models
gain most, and each project's climate forcing becomes one self-describing file that is simpler to share, archive and
reproduce.

1 Introduction

25 Climate forcing in the Soil and Water Assessment Tool Plus (SWAT+) is commonly represented by station-specific text
files. A complete measured-weather station can require separate files for precipitation, temperature, solar radiation, relative
humidity, and wind speed. This design is easy to inspect and edit, but the number of files grows in direct proportion to the
number of stations. Regional and continental SWAT+ projects can consequently contain thousands of small weather files,
even though the restructuring from SWAT2012 to SWAT+ substantially reduced fragmentation in other model inputs
30 (Bieger et al., 2017; Plunge et al., 2024). Opening and parsing these files repeatedly can increase initialisation time during



calibration, sensitivity analysis, and ensemble simulation, while the file count complicates project transfer, versioning, and archiving. Similar file-system effects have been reported in other large environmental modelling workflows (Yepes-Arbós et al., 2022; Sarmany et al., 2024).

The problem is particularly relevant to current SWAT+ applications. SWAT and SWAT+ are used for basin-scale water quantity and quality studies (Arnold et al., 2012; Bieger et al., 2017), and newer workflows support high-resolution continental and global model construction (Chawanda et al., 2020, 2025). Consolidated input files have previously been used in SWAT to reduce the cost of repeated runs (Yen et al., 2014). SWATprepR can load NetCDF weather and still writes the separate station files the model reads (Plunge et al., 2024). The development here is a reader inside SWAT+: it loads one compressed dataset during initialisation and fills the climate arrays the process routines already use.

NetCDF is widely used to store multidimensional geoscientific data. NetCDF-4 uses an HDF5 storage layer and supports lossless compression, chunking, and a portable binary representation (Delaunay et al., 2019). The converter also records variable attributes under the Climate and Forecast metadata conventions (Hassell et al., 2017). Performance still depends on data layout, access pattern, compression settings, software libraries, storage hardware, and cache state. Use of NetCDF alone does not determine input speed (Huang et al., 2014; Wong et al., 2015; Yepes-Arbós et al., 2022). The implementation requires a separate description and benchmark.

This paper presents a development and technical evaluation of native NetCDF climate input in SWAT+ (commit e7fe08a). The objectives are to (1) document the NetCDF schema and the source-code pathway used to connect it to the existing SWAT+ climate structures, (2) provide a conversion workflow for existing SWAT+ projects, (3) test whether changing the storage format changes selected model outputs, and (4) quantify file-count, storage, and initialisation-time differences for station networks containing up to 700 stations. The contribution is deliberately specific to SWAT+. We discuss implementation choices that may be relevant elsewhere, but we do not claim that the measured scaling transfers unchanged to other models or computing environments.

2 Model development

2.1 Legacy SWAT+ climate input and design requirements

At initialisation, SWAT+ reads the weather-station configuration and resolves station names to the measured precipitation, temperature, solar-radiation, humidity, and wind records. The measurement readers allocate variable-specific arrays, and *climate_control* subsequently copies values for the current day into each *weather_station%weat* record. Downstream hydrological and water-quality routines consume this common weather state and do not read the input files directly.

The NetCDF development was designed around three requirements. First, existing text-input projects and the daily process routines had to remain unchanged. Second, all measured variables had to be available through the existing SWAT+ climate



arrays so that the new pathway introduced no NetCDF-specific branches in process code. Third, conversion from a working text-input project had to be automated and reversible. The resulting implementation changes data ingestion during initialisation but does not change the equations, parametrisations, routing, or output routines of SWAT+.

2.2 NetCDF schema and station manifest

- 65 The input dataset uses time, lat, and lon dimensions and the short variable names already used by SWAT+: pcp, tmin, tmax, slr, hmd, and wnd (Fig. 1). The variables represent daily precipitation, minimum and maximum air temperature, solar radiation, relative humidity, and wind speed, respectively. Coordinate variables store latitude in degrees north, longitude in degrees east, and time as days since a stated reference date on a Gregorian calendar. The converter writes 32-bit floating-point climate values to a NetCDF-4 file, applies lossless zlib compression and shuffling, and records units and descriptive attributes. The dataset created for this study follows a time–latitude–longitude layout and declares the Climate and Forecast metadata convention used by the converter.
- 70

NetCDF-4 climate dataset

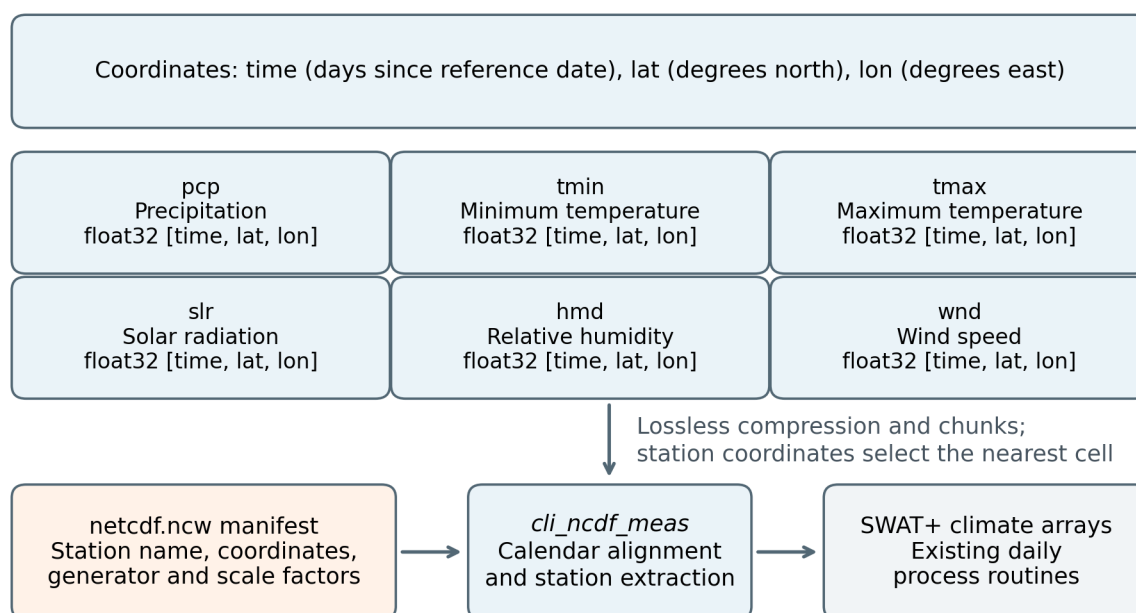


Figure 1: NetCDF climate-input structure used by the SWAT+ implementation. The fixed variable names are precipitation (pcp), minimum and maximum air temperature (tmin and tmax), solar radiation (slr), relative humidity (hmd), and wind speed (wnd).



75 A text manifest named `netcdf.new` connects SWAT+ weather stations to the NetCDF dataset. Each manifest row contains the SWAT+ station name, weather-generator reference, latitude, longitude, elevation, and optional variable-specific scale factors. The climate entry in `file.cio` selects `netcdf.new`, while the climate-path entries identify the NetCDF file. This explicit switch preserves the legacy configuration and permits the two input pathways to be tested with otherwise identical projects.

The implemented reader expects the schema and variable names documented here. It is not intended as a general Climate and
80 Forecast conventions reader for arbitrary archives. In particular, the evaluated workflow uses daily Gregorian records, a latitude–longitude grid, and valid data at every grid cell selected by the station manifest. Variable units must match the SWAT+ climate structures. Manifest factors permit multiplicative scaling; the reader does not infer unit conversions or apply additive offsets.

The Zambezi file used for the paired runs contains 16,437 daily records from 1 January 1980 through 31 December 2024 on
85 a Gregorian calendar, with no gaps in the time coordinate (days since 1970-01-01). The grid is 28 latitude by 48 longitude cells at 0.5°, from 21.75° S to 8.25° S and from 17.25° E to 40.75° E. The six climate variables are float32, stored with chunks of 200 time steps by 10 latitude cells by 12 longitude cells, zlib level 9, shuffle, and Fletcher32 checksums. Their fill value is -9999. Relative humidity is stored as a fraction. The file declares CF-1.7. The simulation uses only 1980–1985 of that file. The reader still allocates the full time and grid arrays.

90 The tested loader accepts the calendars standard, gregorian, and proleptic_gregorian. It aligns the series by offsetting the first time value to 1 January of the simulation start year, then reads one record per day and counts years with leap days. It does not check that later time coordinates are daily or free of gaps. The nearest grid cell is the smallest Euclidean distance in degrees of latitude and longitude. On a tie, the first cell in scan order is kept. A manifest scale factor of -99 selects weather generation for that variable. Any other factor, including the value 1 used for every climate variable at all 1,148 stations in
95 this manifest, keeps the station on the measured-gauge path. The potential-evapotranspiration column in that manifest is null.

2.3 Source-code integration

The run controller checks the climate entry in `file.cio` during initialisation. When the entry is `netcdf.new`, the legacy measured-climate readers are bypassed and the NetCDF-specific loader is called. The modified `cli_staread` routine reads the manifest, allocates the weather stations, stores coordinates and scale factors, and prepares the same station-index fields used
100 by the legacy pathway. NetCDF support is a build option: the CMake option `ENABLE_NETCDF`, off by default, compiles the reader and links the NetCDF C library. A build without it reads text input only and has no NetCDF dependency; `NetCDF_Climate_Input.md` in the source archive describes both builds.

The `cli_ncdf_meas` routine interfaces with the NetCDF C library through Fortran `iso_c_binding`. It opens the specified dataset, reads the coordinate arrays and the six climate variables for the whole domain, and copies the nearest cell into the
105 standard SWAT+ measurement structures. Manifest scale factors are applied during that copy. Negative precipitation, solar



radiation and wind values are set to zero, and humidity is limited to the range 0 to 1. Temperature is not clipped. Once initialisation is complete, the precipitation controller, *climate_control*, and the model time loop use the populated structures and do not read the NetCDF file again.

This front-loaded design confines the new dependency to initialisation and minimises changes to mature process code. It also explains the expected equivalence: if the converter preserves the text values and the NetCDF loader reconstructs the same measurement arrays, both pathways present the same daily weather state to the model. The trade-off is memory. The evaluated reader loads the climate variables for the prepared domain before extracting station series; memory demand therefore grows with the spatial and temporal dimensions of that file. Users should prepare a domain-specific dataset. The current implementation is not suited to an unrestricted global archive.

115 2.4 SWAT+ NetCDF Converter

The SWAT+ NetCDF Converter reads an existing SWAT+ weather-station definition and the associated measured-weather files, constructs the latitude–longitude grid, writes the NetCDF variables and metadata, creates *netcdf.new*, and updates a copy of *file.cio*. It does not interpolate the source observations: values are assigned to the grid cells corresponding to the input station coordinates, while cells without station data remain missing. If multiple stations occupy the same grid cell at the chosen resolution, that collision must be resolved by selecting a finer resolution or preprocessing the station set. The original text-input project is retained so that paired runs can be made.

The NetCDF file for the numerical test was written with converter version 1.0 (Chawanda, 2025). The current release, version 1.2 (Chawanda, 2026a), builds a grid that covers every station and stops when two stations at different coordinates share a cell. Because the converter writes the schema expected by the model reader, it is the supported migration path.

125 3 Evaluation design

3.1 Zambezi River basin test case

The development was evaluated with a SWAT+ model of the Zambezi River basin produced with the CoSWAT workflow (Chawanda et al., 2025). The model uses ASTER elevation data (Yue et al., 2017), FAO soil data (Fischer et al., 2008), European Space Agency land-cover data (Defourny et al., 2017), and the Global Reservoir and Dam database (Lehner et al., 2011). The delineation and physical configuration were used as supplied in the archived project. Daily 20CRv3–ERA5 data from the ISIMIP3a repository supplied the climate forcing at 0.5° resolution (Lange et al., 2025). The model was not calibrated because the evaluation concerns numerical consistency between input pathways, not the hydrological skill of the Zambezi configuration.



The same climate records were represented in two ways: the conventional collection of SWAT+ text files and a NetCDF-4
135 dataset created from those files. Apart from the climate-input selection and its associated files, the model configuration,
executable, simulation period, parameters, and requested outputs were held constant.

The paired runs used the same 1,148 climate stations. The simulation window was 1 January 1980 through 31 December
1985 (2,192 days). Printed outputs cover 1981–1985. The NetCDF file itself continues through 31 December 2024. In both
runs, the basin row of file.cio has a null carbon field appended so that the executable reads the following records correctly;
140 all other input files match the archived project. Both copies of codes.bsn set carbon = 0, so *carbon_bsn_read* returns before
opening the carbon file.

All reported simulations and timing experiments used SWAT+ 62 commit e7fe08a, archived with the data (Chawanda,
2026b). Intel ifx 2025.0.4 compiled the Release executable with -fpp -free -fpe0 -traceback -O. The source also compiles
with GNU Fortran 16.2.1; all runs in this study used the Intel executable. The generated static-link flag was removed to link
145 the same object files against installed shared libraries; the model source was unchanged. The runtime NetCDF C library was
version 4.10.1. Each paired run used a fresh container and working directory.

3.2 Numerical-equivalence test

Paired simulations were compared for every top-level text output and for selected basin, hydrologic response unit (HRU),
channel, and aquifer series. The selected series were basin-average precipitation, evapotranspiration (ET), and surface
150 runoff; the same three variables for HRUs 20, 50, 100, and 300; streamflow for channels 40, 80, 200, and 400; and recharge
for aquifers 11, 32, 61, and 201. Monthly series were compared for the basin, HRU, channel, and aquifer objects. Annual
aquifer recharge was compared as well. Each monthly series in the comparison contains 60 values, and each annual aquifer
series contains 5 values, which is the printed window 1981–1985 after the simulation year 1980 is omitted. Agreement was
summarised with the maximum absolute difference, the maximum relative difference, root-mean-square error (RMSE),
155 percent bias (PBIAS), Nash–Sutcliffe efficiency (NSE), Kling–Gupta efficiency (KGE), and the coefficient of determination
(R^2). These statistics are numerical-consistency diagnostics. They are not an evaluation against observations. Forcing values
were compared separately, at 32-bit precision, before the reader’s lower-bound clip. Maximum relative differences use the
text value with a denominator floor of 10^{-12} , as recorded in the analysis scripts.

3.3 Storage and initialisation benchmark

160 The benchmark used 15 station configurations: 5, 25, 50, 100, 150, 200, 250, 350, 400, 450, 500, 550, 600, 650 and 700
stations. The subsets were spread across the Zambezi region by greedy distance selection and iterative swaps in latitude–
longitude coordinates. Both formats cover the same 1980–2024 record.



The NetCDF scaling datasets were built from each text case on the fixed 28 latitude by 48 longitude grid used by the numerical test, at 0.5° spacing, with a script included in the data archive (Chawanda, 2026b). The station subsets were loaded independently; they are not nested within the 700-station case. All six float32 variables use chunks of 200 time steps by 10 latitude cells by 12 longitude cells, zlib level 9, shuffle and Fletcher32. Unselected cells contain the fill value -9999. Every selected station lies within the grid bounds and has a distinct cell. After writing, every selected daily value for 1980–2024 was compared with its corresponding text record. Across the 15 configurations this covered 491,137,560 station–variable–day values, with zero differences. The fixed spatial and temporal dimensions separate changes in station occupancy from domain size, and every case uses the same chunk layout.

Timing uses a separate early-stopping benchmark built from the same model object files, with an instrumented *proc_read* that stops after climate loading. All other model objects are reused. The primary measurement is the benchmark process duration inside the container, including preparation before climate loading and excluding Docker startup and hydrological simulation. A separate *SYSTEM_CLOCK* timer records the climate read alone.

The main repeat runs on an AMD Threadripper PRO 3995WX machine with 64 physical cores and 256 GiB installed memory, Debian 13, Linux 6.12.94, and a WD_BLACK SN850X NVMe drive with ext4. Each process is pinned to logical CPU 10 and uses one OpenMP thread. A fresh container and clean working directory are used for each repetition. One excluded warm-up per format and station count establishes the warm-cache protocol, and format order alternates between repetitions. Thirty measured repetitions are recorded per format and station count. Fresh containers do not empty the host filesystem cache. An additional three repetitions per format at 700 stations request eviction of the climate files through *POSIX_FADV_DONTNEED*; this is an advisory eviction experiment and does not establish a guaranteed cold cache.

Figure 4 reports the median and the minimum-to-maximum range of all 30 measured repetitions in each station/format group; no repetition is excluded. In every repetition the summed user and system CPU time was within 0.08 s of the process duration. This is consistent with predominantly CPU-bound execution under the warm-cache protocol; the measurements do not separately resolve input/output and scheduling waits.

Storage was measured as the aggregate size of the climate-input files at each station count. Both totals include the shared weather-generator file. The text total also includes the station precipitation, temperature, solar-radiation, humidity and wind files, their variable lists, and the weather-station definition. The NetCDF total also includes the compressed dataset and *netcdf.ncw*.



190 4 Results

4.1 Numerical consistency

On Intel ifx 2025.0.4, the text and NetCDF runs of e7fe08a finished with return code 0. All 91 top-level output text files were byte-identical between the paired runs, covering 16,962,190 lines per run. The 27 selected series contain 1,400 values (Fig. 2). The maximum absolute difference and the maximum relative difference were 0. RMSE and PBIAS were 0. NSE was 1. KGE and R^2 are 1 at four decimal places; a few of the stored statistics differ from 1 only in the last binary digit of a floating-point calculation on identical inputs. The forcing comparison covered all 15,098,496 values at 1,148 stations for 1980–1985, which is six variables times 2,192 days, and found no differences at float32 precision and no missing sentinels. This is agreement of the two ingestion paths on one compiler, one model configuration, and one forcing file. It is not bitwise identity across compilers, and it is not a test of hydrological skill.

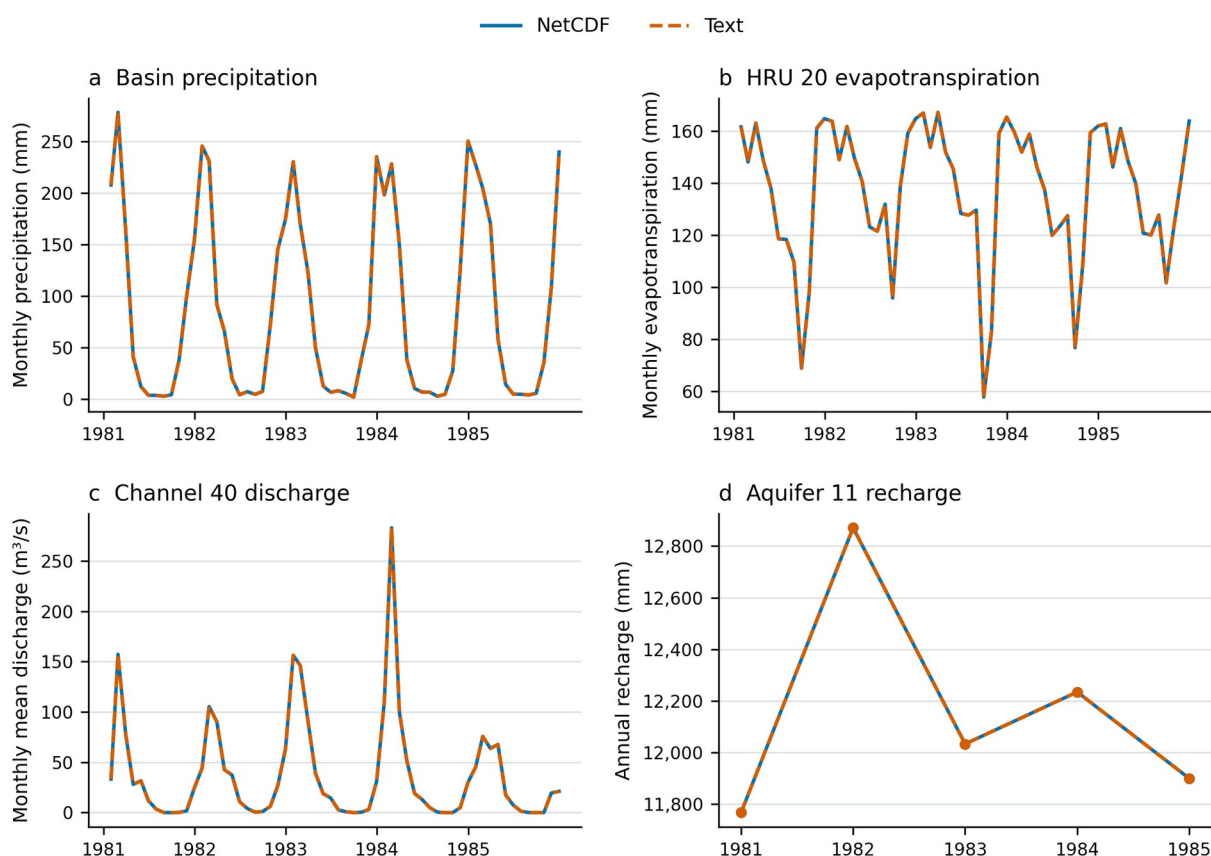
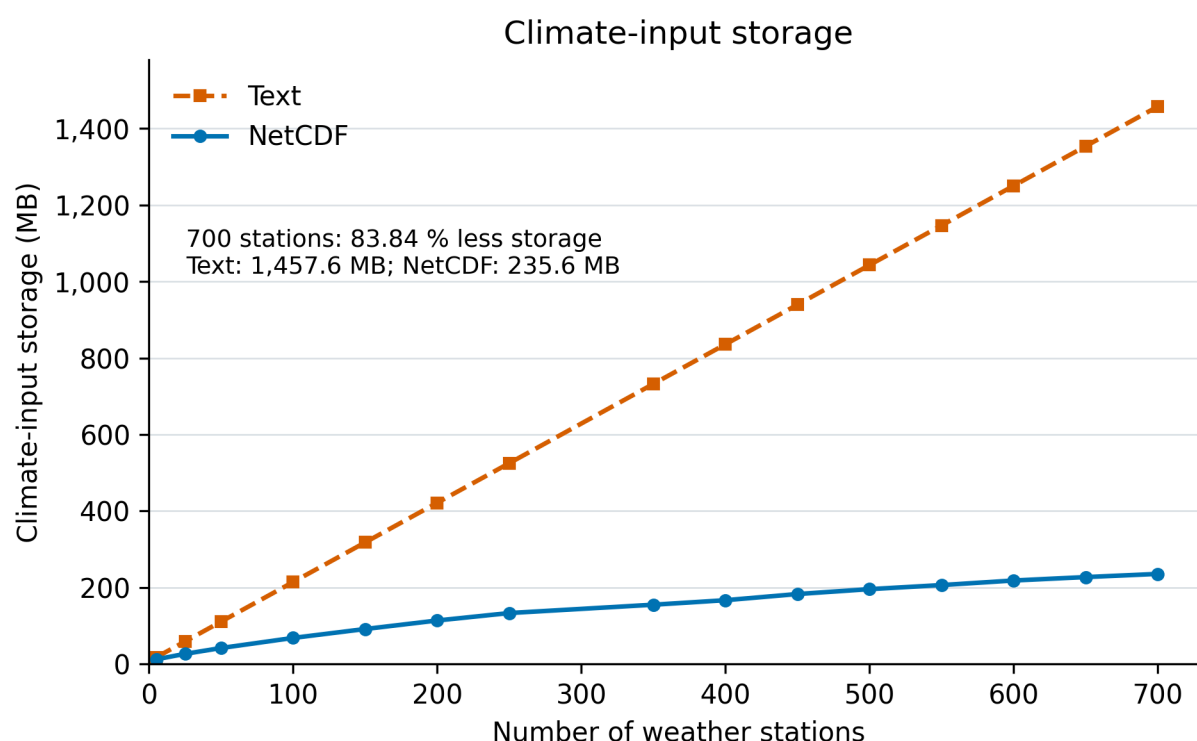


Figure 2: Representative output series from the successful text and NetCDF runs of the uncalibrated Zambezi model. The lines coincide. Basin precipitation, HRU evapotranspiration and channel flow are monthly; aquifer recharge is annual. The printed window is 1981–1985.



4.2 File count and storage

205 At 700 stations, the text climate inputs occupied 1,457,622,593 bytes in 3,507 files. The NetCDF inputs occupied 235,565,093 bytes in three files: the dataset, the station manifest and the shared weather-generator file. The NetCDF total is 16.16 % of the text total, a reduction of 83.84 %. In decimal megabytes (10^6 bytes), the totals are 1,457.6 MB and 235.6 MB. Both representations preserve the selected station records at float32 precision. The storage difference reflects binary representation and lossless compression of this dataset, with the stated domain and chunk layout.



210

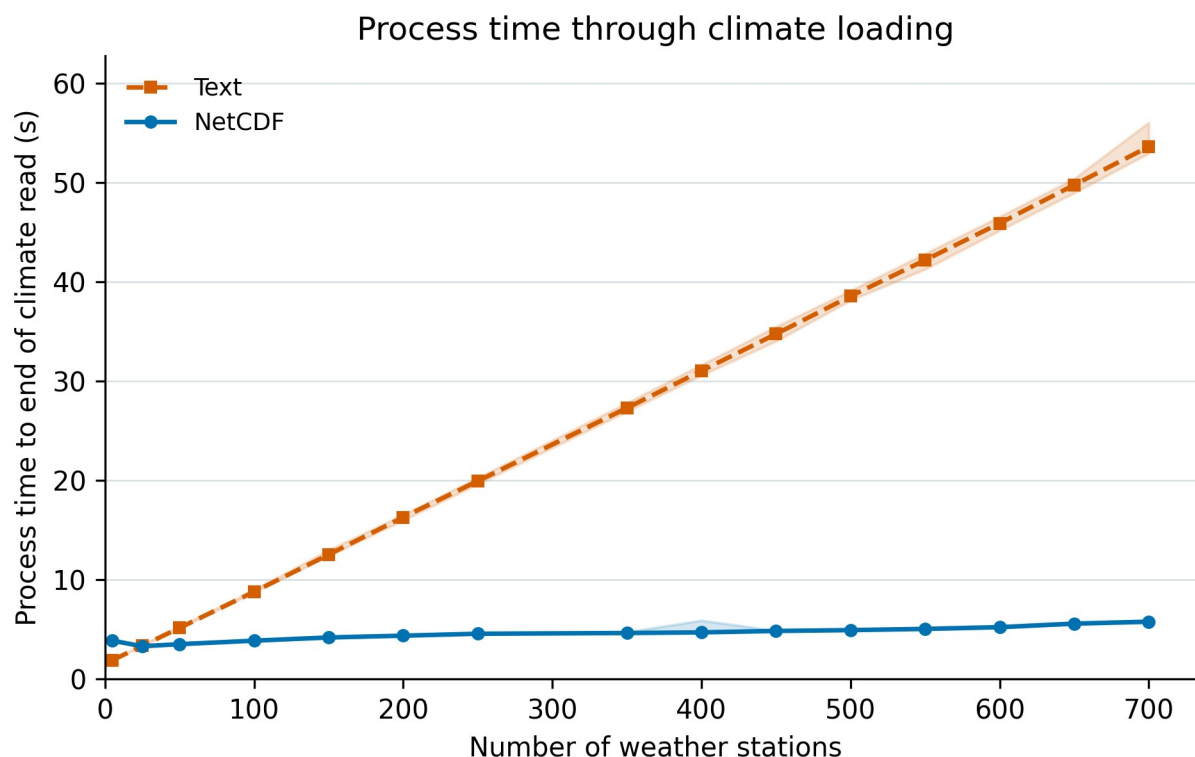
Figure 3: Climate-input storage for the NetCDF datasets and the station text files. Every selected station has a distinct grid cell and all 1980–2024 values match its text records at float32. Both totals include the shared weather-generator file; MB = 1,000,000 bytes.

4.3 Climate initialisation time

215 Benchmark process time through completion of climate loading increased approximately linearly for text inputs, from 1.9 s at 5 stations to 53.6 s at 700 stations (Fig. 4). This is consistent with the growing number of file-open and text-parsing operations. The corresponding NetCDF process time remained between 3.3 and 5.8 s across all 15 configurations. NetCDF was slower at 5 stations (3.9 s against 1.9 s), the two routes were equal within 0.1 s at 25 stations, and NetCDF was faster from 50 stations onwards. At 700 stations the NetCDF process took 5.8 s, 47.9 s less than the text process and about one



220 ninth of its time. The separate climate-only timer gave medians of 4.1 s for NetCDF and 52.0 s for text; preparation before climate loading accounts for the remaining process time. Asking the system to remove the climate files from its cache before a run added at most 0.4 s at 700 stations.



225 **Figure 4: Benchmark process time from startup through completion of climate loading at 5 to 700 stations. Median and minimum-to-maximum range of 30 warm-cache runs per format and station count on a fixed domain; no measured runs are excluded. The duration includes preparation before climate loading and excludes Docker startup and hydrological simulation.**

5 Discussion

230 The identical outputs follow from the reader's position in the model. It runs once, before the simulation loop, and writes into the arrays the text readers fill, so the daily process code receives the same weather state whichever route supplied it. The forcing comparison establishes preservation of every tested value at the 32-bit precision of the climate arrays, and the byte-identical output files demonstrate that changing the input route preserves the simulated results for the paired runs. For an established project whose paired runs produce byte-identical outputs, the format change itself requires no recalibration, and existing validation of those outputs remains applicable. The comparison here establishes equivalence for the tested Zambezi configuration. Input format can therefore be chosen for its storage and performance benefits within the evaluated workflow.



235 Most of the storage reduction comes from binary encoding, and less of it from compression. The text files spend about 21
 bytes on each daily value, because every value is written as formatted characters with repeated year and day columns. The
 NetCDF dataset spends about 3.3 bytes per value, below the 4 bytes of an uncompressed 32-bit number, even though it also
 stores the empty cells of the grid. Lossless compression thus adds a gain on top of the binary format, and the size of that gain
 depends on the data (Delaunay et al., 2019); long runs of dry days, for example, compress well. The saving should therefore
 240 carry over to other basins and forcing products, while its exact size will vary.

The two timing curves differ because the two routes scale with different things. The text route opens and parses five files per
 station, so its cost grows with the number of stations: about 74 ms per station on top of a shared model start-up of about 1.4
 s. The NetCDF route reads the whole prepared domain once, about 133 million values here, and then copies each station's
 series, so its cost depends mainly on the size of the domain and the record length. The observed marginal slope is about 3 ms
 245 per additional station; this benchmark does not isolate the cost of station copying from other processing. This explains why
 text was faster for the smallest networks and why the routes cross at about 25 to 30 stations. It also means that the advantage
 depends on how densely stations fill the domain: a sparse network inside a large grid pays for reading cells it does not use.
 Selective reads could reduce this overhead and memory use, depending on chunk layout and station distribution. The same
 dependence of NetCDF performance on layout and access pattern has been reported for other models (Huang et al., 2014;
 250 Wong et al., 2015; Yepes-Arbós et al., 2022).

The flat NetCDF curve applies to the conditions of this benchmark: a fixed domain and record length, one chunk layout and
 compression level, a local NVMe drive and a warm cache. Other chunk shapes, compression levels, domains, record lengths,
 file systems or cache conditions can give different results, and a shared high-performance computing file system remains to
 be tested. The benchmark compares the two implemented input routes, including their existing preparation steps. It does not
 255 compare NetCDF with consolidated text files, a database, uncompressed NetCDF, Zarr or parallel NetCDF, and it does not
 separate the costs of decompression, coordinate search, allocation and array copying; these comparisons are possible
 extensions.

The evaluated workflow uses the converter schema (Sect. 2.2) and complete daily weather records, with a scale factor of 1
 for every manifest row. Further implementation details are provided with the archived source code (Chawanda, 2026b).

260 The practical value lies where SWAT+ applications are moving. Gridded forcing produces dense station networks, such as
 the 1,148 stations of the 0.5° Zambezi setup used here, and continental and global setups such as CoSWAT (Chawanda et al.,
 2025) have many more. In those settings the saving recurs at every initialisation, so it grows with the number of runs: at 700
 stations, 1,000 serial runs of a calibration or ensemble would spend about 13 h less in initialisation through completion of
 climate loading. This is arithmetic on the measured difference, and parallel execution, shared file systems and the length of
 265 the hydrological simulation will change the time actually saved. Yen et al. (2014) reached the same conclusion for
 consolidated SWAT inputs in repeated runs; the pathway here keeps the existing model structures and uses a standard



format, so the same file can also be inspected and processed with common NetCDF tools. The demonstrated migration path converts existing station files once and reuses the resulting NetCDF dataset in subsequent runs. Workflows that already use NetCDF forcing, such as SWATprepR (Plunge et al., 2024), could avoid intermediate station files by producing the reader's supported schema directly.

The change also affects how projects are shared and reproduced. One self-describing file with units and coordinates replaces thousands of station files, which makes a project easier to move, version and archive, and fewer small files reduce the file-system load reported for other large modelling workflows (Yepes-Arbós et al., 2022; Sarmany et al., 2024). Because stations are found by coordinates, the pathway also removes a failure mode of the text route, where a station list edited out of lexical order can cause a measurement reference to be missed. These benefits apply to projects of any size, and because the text route is unchanged and the NetCDF route is selected per project, users can adopt it where it helps without affecting existing work.

6 Conclusions

This paper added a native NetCDF-4 climate input to SWAT+, with a converter that turns existing station weather files into the format the reader expects. The reader runs during initialisation, fills the standard climate arrays and leaves the daily process code unchanged, so existing text-input projects run as before and each project chooses its input route in file.cio. In paired runs of the uncalibrated Zambezi model, the text and NetCDF routes produced identical output files, and every forcing value matched at 32-bit precision.

At 700 stations, the NetCDF inputs needed 83.84 % less storage than the text files (235.6 MB against 1,457.6 MB, including the shared weather-generator file), and benchmark process time through completion of climate loading was 5.8 s against 53.6 s under the fixed-domain, warm-cache protocol; text was faster only at the smallest configuration. The text cost grows with every station, while the NetCDF cost depends mainly on the size of the domain, so the advantage increases with the density of the station network. This matters most for large-scale modelling. Regional, continental and global SWAT+ setups driven by gridded forcing have thousands of stations, and their calibration, sensitivity and ensemble studies repeat the climate read in every run, so time savings accumulate across repeated runs. Storage savings apply to each retained copy of the climate inputs and accumulate with run count only when runs keep separate input copies. The storage values describe one dataset and one compression configuration.

The pathway is a focused input option. Its supported workflow is conversion of existing SWAT+ daily weather files into the documented schema, followed by native reading into the established climate structures, and the text pathway remains available, including in builds without NetCDF support, which need no NetCDF library (Sect. 2.3). Future work should reduce memory use through selective reads, broaden metadata and calendar support, and evaluate performance across computing and storage architectures.



Code and data availability

The implementation and all reported experiments use SWAT+ 62 commit e7fe08a47b96aac8a4904d4c19340088eb6dc2e0
300 from the netCDF-Support branch of <https://github.com/celray/swatplus>. The source snapshot is archived as model/swatplus-e7fe08a-source.zip in the data package (Chawanda, 2026b), with the experiment executables in model/executables/ and the build record in model/build_record.json. The continuing model repository is <https://github.com/swat-model/swatplus>, and the NetCDF pathway is intended to be merged into it. The source distribution includes the LGPL-2.1 licence.

The SWAT+ NetCDF Converter version 1.0, used for the numerical-test file, is archived on Zenodo (Chawanda, 2025;
305 <https://doi.org/10.5281/zenodo.17656407>). Version 1.2, with the grid correction, is archived in the same record (Chawanda, 2026a; <https://doi.org/10.5281/zenodo.23285415>). The source distribution includes the MIT licence; continuing development is at <https://github.com/celray/swat2netcdf>.

The Zambezi model, the paired-run inputs and execution records, the 15 scaling cases, all timing records, the output comparison tables and the analysis and figure scripts are archived on figshare (Chawanda, 2026b;
310 <https://doi.org/10.6084/m9.figshare.34383972.v1>). The original ISIMIP3a atmospheric forcing is available from Lange et al. (2025; <https://doi.org/10.48364/ISIMIP.982724.3>).

Author contributions

CJC conceived the study, developed the NetCDF implementation and conversion software, performed the experiments and analysis, prepared the figures, and wrote the initial manuscript. JA contributed to conceptualisation and manuscript review.
315 RS contributed to manuscript review. OD and JPTO contributed to software methodology and manuscript review.

Competing interests

The authors declare that they have no conflict of interest.

Acknowledgements

The authors thank the Blackland Research and Extension Center at Texas A&M AgriLife Research for institutional support.

320 Financial support

This research received no specific funding.



References

- Arnold, J. G., Moriasi, D. N., Gassman, P. W., Abbaspour, K. C., White, M. J., Srinivasan, R., Santhi, C., Harmel, R. D., van Griensven, A., Van Liew, M. W., Kannan, N., and Jha, M. K.: SWAT: Model use, calibration, and validation, Trans. ASABE, 55, 1491–1508, <https://doi.org/10.13031/2013.42256>, 2012.
- 325 Bieger, K., Arnold, J. G., Rathjens, H., White, M. J., Bosch, D. D., Allen, P. M., Volk, M., and Srinivasan, R.: Introduction to SWAT+, a completely restructured version of the Soil and Water Assessment Tool, J. Am. Water Resour. As., 53, 115–130, <https://doi.org/10.1111/1752-1688.12482>, 2017.
- Chawanda, C. J.: SWAT+ NetCDF Converter, version 1.0, Zenodo [code], <https://doi.org/10.5281/zenodo.17656407>, 2025.
- 330 Chawanda, C. J.: SWAT+ NetCDF Converter, version 1.2, Zenodo [code], <https://doi.org/10.5281/zenodo.23285415>, 2026a.
- Chawanda, C. J.: SWAT+ NetCDF climate input: model, data and analysis scripts, figshare [code and data set], <https://doi.org/10.6084/m9.figshare.34383972.v1>, 2026b.
- Chawanda, C. J., George, C., Thiery, W., van Griensven, A., Tech, J., Arnold, J., and Srinivasan, R.: User-friendly workflows for catchment modelling: towards reproducible SWAT+ model studies, Environ. Modell. Softw., 134, 104812, <https://doi.org/10.1016/j.envsoft.2020.104812>, 2020.
- 335 Chawanda, C. J., van Griensven, A., Nkwasa, A., Teran Orsini, J. P., Jeong, J., Choi, S.-K., Srinivasan, R., and Arnold, J. G.: CoSWAT Model v1: a high-resolution global SWAT+ hydrological model, Hydrol. Earth Syst. Sci., 29, 6901–6916, <https://doi.org/10.5194/hess-29-6901-2025>, 2025.
- Defourny, P., Lamarche, C., Bontemps, S., De Maet, T., Van Bogaert, E., Moreau, I., Brockmann, C., Boettcher, M., Kirches, G., Wevers, J., and Santoro, M.: Land Cover CCI: Product User Guide, version 2.0, European Space Agency and UCL-Geomatics, https://maps.elie.ucl.ac.be/CCI/viewer/download/ESACCI-LC-Ph2-PUGv2_2.0.pdf (last access: 12 October 2026), 2017.
- Delaunay, X., Courtois, A., and Gouillon, F.: Evaluation of lossless and lossy algorithms for the compression of scientific datasets in netCDF-4 or HDF5 files, Geosci. Model Dev., 12, 4099–4113, <https://doi.org/10.5194/gmd-12-4099-2019>, 2019.
- 345 Fischer, G., Nachtergaele, F., Prieler, S., van Velthuisen, H. T., Verelst, L., and Wiberg, D.: Global Agro-ecological Zones Assessment for Agriculture (GAEZ 2008), IIASA, Laxenburg, Austria, and FAO, Rome, Italy, 2008.
- Hassell, D., Gregory, J., Blower, J., Lawrence, B. N., and Taylor, K. E.: A data model of the Climate and Forecast metadata conventions (CF-1.6) with a software implementation (cf-python v2.1), Geosci. Model Dev., 10, 4619–4646, <https://doi.org/10.5194/gmd-10-4619-2017>, 2017.



- 350 Huang, X. M., Wang, W. C., Fu, H. H., Yang, G. W., Wang, B., and Zhang, C.: A fast input/output library for high-resolution climate models, *Geosci. Model Dev.*, 7, 93–103, <https://doi.org/10.5194/gmd-7-93-2014>, 2014.
- Lange, S., Quesada-Chacón, D., Mengel, M., Treu, S., and Büchner, M.: ISIMIP3a atmospheric climate input data, version 1.3, ISIMIP Repository [data set], <https://doi.org/10.48364/ISIMIP.982724.3>, 2025.
- Lehner, B., Liermann, C. R., Revenga, C., Vörösmarty, C., Fekete, B., Crouzet, P., Döll, P., Endejan, M., Frenken, K.,
 355 Magome, J., Nilsson, C., Robertson, J. C., Rödel, R., Sindorf, N., and Wisser, D.: High-resolution mapping of the world's reservoirs and dams for sustainable river-flow management, *Front. Ecol. Environ.*, 9, 494–502, <https://doi.org/10.1890/100125>, 2011.
- Plunge, S., Szabó, B., Strauch, M., Čerkasova, N., Schürz, C., and Piniewski, M.: SWAT+ input data preparation in a scripted workflow: SWATprepR, *Environ. Sci. Eur.*, 36, 53, <https://doi.org/10.1186/s12302-024-00873-1>, 2024.
- 360 Sarmany, D., Valentini, M., Maciel, P., Geier, P., Smart, S., Aguridan, R., Hawkes, J., and Quintino, T.: MultiIO: a framework for message-driven data routing for weather and climate simulations, in: Proceedings of the Platform for Advanced Scientific Computing Conference (PASC '24), Zurich, Switzerland, 3–5 June 2024, 1–12, <https://doi.org/10.1145/3659914.3659938>, 2024.
- Wong, D. C., Yang, C. E., Fu, J. S., Wong, K., and Gao, Y.: An approach to enhance pnetCDF performance in
 365 environmental modeling applications, *Geosci. Model Dev.*, 8, 1033–1046, <https://doi.org/10.5194/gmd-8-1033-2015>, 2015.
- Yen, H., Ahmadi, M., White, M. J., Wang, X., and Arnold, J. G.: C-SWAT: the Soil and Water Assessment Tool with consolidated input files in alleviating computational burden of recursive simulations, *Comput. Geosci.*, 72, 221–232, <https://doi.org/10.1016/j.cageo.2014.07.017>, 2014.
- Yepes-Arbós, X., van den Oord, G., Acosta, M. C., and Carver, G. D.: Evaluation and optimisation of the I/O scalability for
 370 the next generation of Earth system models: IFS CY43R3 and XIOS 2.0 integration as a case study, *Geosci. Model Dev.*, 15, 379–394, <https://doi.org/10.5194/gmd-15-379-2022>, 2022.
- Yue, L., Shen, H., Zhang, L., Zheng, X., Zhang, F., and Yuan, Q.: High-quality seamless DEM generation blending SRTM-1, ASTER GDEM v2 and ICESat/GLAS observations, *ISPRS J. Photogramm.*, 123, 20–34, <https://doi.org/10.1016/j.isprsjprs.2016.11.002>, 2017.