DM06

Best Practices On Reprogramming A Legacy Global Water Model

In just 15 minutes, you can contribute valuable feedback to improve our research software WaterGAP. **Reserach software is defined as software that encompasses various elements such as source code files, algorithms, computational workflows, and executables developed during the research process or for a research objective**.

Background:

The global hydrological model, WaterGAP, has been in development for over 25 years, created primarily by PhDs and Postdocs with limited software development experience. Recognizing the challenges posed by poor code quality, we've undertaken a reprogramming initiative to improve readability, modifiability, reusability, and maintainability.

Study Focus:

This survey which aims at unlocking **user perspectives on a reprogrammed model** is a crucial component of the study "Best Practices on Reprogramming a Legacy Global Water Model". Our study aims to share real-case experiences in the reprogramming process, providing insights from inception to the final code.

Your Role:

We're keen to hear your perspectives on the reprogrammed model, focusing on readability, modification, and potential contributions to model development. Your feedback will greatly contribute to the long-term research software sustainability.
Note: **sustainability here is defined as the process of developing and maintaining software that continues to meet its purpose over time**

Participation is anonymous, voluntary, and you can stop anytime. The results will be anonymized and published in the peer-reviewed journal Geoscientific Model Development.

Best wishes,
Emmanuel Nyenah, Petra Döll, and Robert Reinecke

DM01

1. What phase of your career are you currently in?

- ◯ Student (undergraduate, Bachelor/Master or similar)
- ◯ Scientific staff
- ◯ PhD (candidate / student)
- ◯ Postdoc
- ◯ Group leader / junior professor
- ◯ Associate professor
- ◯ Full professor
- ◯ Other (scientific) ______
- ◯ Other (non-scientific) ______

DM02

2. How long have you been working in your research field?

- ◯ Less than 1 year
- ◯ 1-5 years
- ◯ 6-10 years
- ◯ 11-15 years
- ◯ 16+ years

DM03

3. To which field within the geosciences does your research mainly belong?

Please select one or more options.

- ☐ Hydrology
- ☐ Climate Science
- ☐ Geophysics
- ☐ Environmental Science
- ☐ Water Quality
- ☐ Ecology
- ☐ Soil Science
- ☐ Geomorphology
- ☐ Earth System Sciences
- ☐ Other (please specify) ______

US07

4. Do you have any background in Hydrology ?

- None
- Novice
- Intermediate
- Advanced

DM05

5. What is the focus of your research

Select option that describes your typical daily work routine.

- I use existing computational models (e. g. run with desired input data) for research without developing them myself
- I develop and use computational models for my research
- I improve our understanding of processes by conducting experiments in the field or lab.
- I develop computational models but do not conduct any research.
- I use existing results of computational models for my research
- Other (please specify)

DM04

6. What geographic scale are you working on?

- Global
- Continental
- < Million km^2
- < 1000 km^2
- < km^2
- < m^2
- < cm^2
- Mixed

- Does not apply to me

US10

In the following, we kindly ask you to keep in mind that **research software** includes source code files, algorithms, computational workflows, and executables developed during the research process or for a research objective.

Plotting scripts are not research software.

US01

7. How often do you use research software?

This involves reading software documentation and installing and running the software.

- Every day
- Multiple times per week
- Once a week
- Once a month
- Less than once a month

- I do not use research software in my own research.

US04

8. Do you own a research software?

- Yes, I own the research software
- No, my supervisor owns the research software
- No, the research software belongs to the instituion I work for

---

- Does not apply to me

US03

9. How often do you develop code for research software in your research practice?

This involves adding and/or modifying code in the source code of research software.
Note: This could be your own research software or research software developed by someone else

- Every day
- Multiple times per week
- Once a week
- Once a month
- Less than once a month

- I used to develop research software in the past but not anymore.
- Does not apply to me

US05

10. How many years of programming experience do you have?

US06

11. What kind of programming languages do you use?

Please select one or more options.

- [ ] Python
- [ ] R
- [ ] MATLAB
- [ ] Fortran
- [ ] C++
- [ ] Other (please specify)

- [ ] None

US08

12. How did you learn to program software?

Please select one or more options.

- [ ] Computer Science degree
- [ ] Courses during undergraduate studies (e.g., BA / Bsc)
- [ ] Courses during postgraduate studies (e.g., MA / MSc / PhD)
- [ ] Workshops
- [ ] Online-courses
- [ ] Self-taught /autodidact

- [ ] I do not know how to program

US09

13. Do you practice any of the following methods / concepts as part of your scientific programming routine?

Please select one or more options

- [ ] Object-oriented programming
- [ ] Test-driven development
- [ ] The use of software architecture and software design patterns (such as Model-Controller-View (MVC), Abstract factory, Decorator or similar)
- [ ] Other (please specify)

- [ ] Does not apply to me

UP06

This section explores the research software called **WaterGAP**. It is a **global hydrological model** that has been active since 1996. WaterGAP quantifies human use of groundwater and surface water, as well as water flows, storage, and overall water resources across all land areas on Earth. This model has been instrumental in assessing water resources and water stress levels both historically and in future scenarios, particularly in the context of climate change. Its contribution extends to enhancing our understanding of variations in continental water storage, with a focus on overexploitation and depletion of water resources.

The following are questions on the source code of WaterGAP which will potentially help us improve source code readability, modification, and ease of use in addition to getting potential contributions to model development.

UP02

14. Can you identify what equation is used in the code snippet below?

Please select the right answer.
The image below shows a function which computes:

- ◯ potential evapotranspiration according to Hargreaves for land and surface water bodies.
- ◯ potential evapotranspiration according to Priestley-Taylor for land and surface water bodies.
- ◯ actual evapotranspiration for land and surface water bodies.
- ◯ actual evapotranspiration according to Penman-Monteith for land and surface water bodies.
- ◯ Other (please specify)

- ◯ The function is too complex to understand
- ◯ I dont know what a function in programming is

```python
@njit(cache=True)
def priestley_taylor(temperature, pt_coeff_humid_arid,
                     net_radiation, openwater_net_radiation,
                     x, y ):
    """
    Compute Priestly-Taylor potential evapotranspiration.

    Parameters
    ----------
    temperature : float
        Daily air tempeature, Units : [K]
    pt_coeff_humid_arid : flaot
        Priestley-Taylor coefficient  for humid and arid cells (alpha), Units: [-]
    net_radiation : float
        Net radiation  according to Müller Schmied et al., 2016., Units: [Wm-2]
    openwater_net_radiation : float
        Open water radiation  according to Müller Schmied et al., 2016.,
        Units: [Wm-2]
     x : int
         Latitude index of cell
     y : int
         Longitude index of cell

    Returns
    -------
    potential_evap : float
        Potential evapotranspiration, Units: [mm/day]
    openwater_pot_evap : float
        Open water potential evapotranspiration, Units: [mm/day]

    """
    # Index (x, y) to  print out varibales of interest
    # e.g.  if x==65 and y==137: print(net_radiation)
    # =====================================================================
    # Slope of the saturation kPa°C-1
    # =====================================================================
    # Converting temperature to degrees celcius
    covert_to_degree = 273.15
    conv_temperature = temperature - covert_to_degree

    # Actual name: Slope of the saturation, Units: kPa°C-1
    slope_of_sat_num = 4098 * (0.6108 * np.exp((17.27 * conv_temperature) /
                                           (conv_temperature + 237.3)))

    slope_of_sat_den = (conv_temperature + 237.3)**2

    slope_of_sat = slope_of_sat_num / slope_of_sat_den

```

```
    # ======================================================================
    # Psychrometric constant  kPa°C-1
    # ======================================================================
    # Actual name: Atmospheric pressure,    Units: kPa
    atm_pressure = 101.3

    # Actual name: Latent heat, Units: MJkg-1
    latent_heat = np.where(conv_temperature > 0,
                           (2.501 - (0.002361 * conv_temperature)), 2.835)

    #  Actual name: Psychrometric constant  Unit kPa°C-1
    psy_const = (0.0016286 * atm_pressure) / latent_heat

    # ======================================================================
    #  Priestley-Taylor Potential evapotranspiration (mm/day)
    #  (Eq. 7 in Müller Schmied et al 2021.)
    # ======================================================================
    # Priestley-Taylor coefficient  for potential evapotranspiration(α)
    # Following Shuttleworth (1993), α is set to 1.26 in humid
    # and to 1.74 in (semi)arid cells
    # Humid-arid calssification based on Müller Schmied et al. 2021

    # Coverting net radiation to mm/day
    # Note!!!, I deliberately did not attach "self"  here so I dont
    # convert the final net radiation output to  mm/day.
    net_radiation = (net_radiation * 0.0864) / latent_heat

    # Actual name: Potential evapotranspiration,    Units:  mmd-1
    potential_evap = pt_coeff_humid_arid * ((slope_of_sat * net_radiation)
                                            / (slope_of_sat + psy_const))

    # Accounting for negative net radiation and setting them to zero
    potential_evap = np.where(net_radiation <= 0, 0, potential_evap)

    # ======================================================================
    # Priestley-Taylor open water potential evapotranspiration (mm/day)
    # ======================================================================
    # Coverting net radiation to mm/day
    openwater_net_radiation = \
        (openwater_net_radiation * 0.0864) / latent_heat

    # Actual name: Open water potential evapotranspiration, Units:  mmd-1
    openwater_pot_evap = \
       pt_coeff_humid_arid * ((slope_of_sat * openwater_net_radiation) /
                              (slope_of_sat + psy_const))

    # Accounting for negative net radiation and setting them to zero
    openwater_pot_evap = np.where(openwater_net_radiation <= 0, 0,
                                  openwater_pot_evap)
    return potential_evap, openwater_pot_evap
```

UP04

15. Is the additional documentation helpful in understanding the function shown previously?

The image below shows the theoretical explanation of the function (code snippet) previously shown.

Documentation shown below can be also found on an online interactive web page:
hydrologyfrankfurt.github.io/ReWaterGAP/api_docs/vertical_water_balance/radiation_evapotranspiration.html#radiation-evap

	Strongly Agree				Disagree	No idea
The documentation explains the previous functions clearly.	○	○	○	○	○	○
I now know the purpose of the previous function.	○	○	○	○	○	○
Documentation is too complex to undertstand	○	○	○	○	○	○

Potential Evapotranspiration

`radiation_evapotranspiration.priestley_taylor(temperature, pt_coeff_humid_arid, net_radiation, openwater_net_radiation, x, y)` [source]

Compute Priestly-Taylor potential evapotranspiration.

Parameters:

temperature : ***float***

Daily air tempeature, Units : [K]

pt_coeff_humid_arid : ***flaot***

Priestley-Taylor coefficient for humid and arid cells (alpha), Units: [-]

net_radiation : ***float***

Net radiation according to Müller Schmied et al., 2016., Units: [Wm−2]

openwater_net_radiation : ***float***

> Open water radiation according to Müller Schmied et al., 2016., Units: [Wm−2]

x : ***int***

Latitude index of cell

y : ***int***

Longitude index of cell

Returns:

potential_evap : ***float***

Potential evapotranspiration, Units: [mm/day]

openwater_pot_evap : ***float***

Open water potential evapotranspiration, Units: [mm/day]

The potential evapotranspiration $E_{pot}\,[mm/d]$ is calculated with the **Priestley–Taylor** equation according to Shuttleworth (1993) [4], as:

$$E_{pot} = \alpha\left(\frac{S_a R}{S_a + g}\right)$$

α is set to 1.26 in humid and to 1.74 in (semi)arid cells (see Appendix B in Müller et al. [2]). R is the net radiation $[mm/d]$ that depends on land cover (Table C2, Müller et al. [2]). S_a is the slope of the saturation vapor pressure–temperature relationship, and g is the psychrometric constant $[\frac{kPa}{^\circ C}]$.

Note

All grid cells with an aridity index AI < 0.75 are defined as semiarid/arid grid cells. Furthermore, all grid cells north of 55° N are defined as humid grid cells. For further information on this see Müller et al. [2] Appendix B.

Slope of the saturation and psychrometric constant

s_a is the slope of the saturation vapor pressure-temperature relationship $[\frac{kPa}{^\circ C}]$ defined as:

$$s_a = \frac{4098(0.6108e^{\frac{17.27T}{T+237.3}})}{(T+237.3)^2}$$

where $T\,[^\circ C]$ is the daily air temperature.

The the psychrometric constant $g\,[\frac{kPa}{^\circ C}]$ is defined as:

$$g = \frac{0.0016286 p_a}{l_h}$$

where p_a is atmospheric pressure of the standard atmosphere $(101.3kPa)$, and l_h is latent heat $[\frac{MJ}{kg}]$. Latent heat is calculated as:

$$l_h = \begin{cases} 2.501 - 0.002361T, & \text{if } T > 0 \\ 2.501 + 0.334, & \text{otherwise} \end{cases}$$

UP08

Questions **16** and **17** relate to the code snippet previously shown (also shown below) and associated documentation on potential evapotranspiration according to Priestley-Taylor.

UP03

16. How confident are you to change atmospheric pressure constant from 101.3kPa to 101.325kPa in the equation you identified ?

- ○ Not confident
- ○ Slightly confident
- ○ Moderately confident
- ○ Confident
- ○ Very confident
- ○ Not sure

UP07

17. Which line of code would you modify ?.

Write line number here

```
@njit(cache=True)
def priestley_taylor(temperature, pt_coeff_humid_arid,
                     net_radiation, openwater_net_radiation,
                     x, y ):
    """
    Compute Priestly-Taylor potential evapotranspiration.

    Parameters
    ----------
    temperature : float
        Daily air tempeature, Units : [K]
    pt_coeff_humid_arid : flaot
        Priestley-Taylor coefficient  for humid and arid cells (alpha), Units: [-]
    net_radiation : float
        Net radiation  according to Müller Schmied et al., 2016., Units: [Wm-2]
    openwater_net_radiation : float
        Open water radiation  according to Müller Schmied et al., 2016.,
        Units: [Wm-2]
     x : int
         Latitude index of cell
     y : int
         Longitude index of cell

    Returns
    -------
    potential_evap : float
        Potential evapotranspiration, Units: [mm/day]
    openwater_pot_evap : float
        Open water potential evapotranspiration, Units: [mm/day]

    """
    # Index (x, y) to  print out varibales of interest
    # e.g.  if x==65 and y==137: print(net_radiation)
    # =====================================================================
    # Slope of the saturation kPa°C-1
    # =====================================================================
    # Converting temperature to degrees celcius
    covert_to_degree = 273.15
    conv_temperature = temperature - covert_to_degree

    # Actual name: Slope of the saturation, Units: kPa°C-1
    slope_of_sat_num = 4098 * (0.6108 * np.exp((17.27 * conv_temperature) /
                                               (conv_temperature + 237.3)))

    slope_of_sat_den = (conv_temperature + 237.3)**2

    slope_of_sat = slope_of_sat_num / slope_of_sat_den

```

```
    # =====================================================================
    # Psychrometric constant  kPa°C-1
    # =====================================================================
    # Actual name: Atmospheric pressure,    Units: kPa
    atm_pressure = 101.3

    # Actual name: Latent heat, Units: MJkg-1
    latent_heat = np.where(conv_temperature > 0,
                           (2.501 - (0.002361 * conv_temperature)), 2.835)

    #  Actual name: Psychrometric constant  Unit kPa°C-1
    psy_const = (0.0016286 * atm_pressure) / latent_heat

    # =====================================================================
    #  Priestley-Taylor Potential evapotranspiration (mm/day)
    #  (Eq. 7 in Müller Schmied et al 2021.)
    # =====================================================================
    # Priestley-Taylor coefficient  for potential evapotranspiration(α)
    # Following Shuttleworth (1993), α is set to 1.26 in humid
    # and to 1.74 in (semi)arid cells
    # Humid-arid calssification based on Müller Schmied et al. 2021

    # Coverting net radiation to mm/day
    # Note!!!, I deliberately did not attach "self"  here so I dont
    # convert the final net radiation output to  mm/day.
    net_radiation = (net_radiation * 0.0864) / latent_heat

    # Actual name: Potential evapotranspiration,    Units:  mmd-1
    potential_evap = pt_coeff_humid_arid * ((slope_of_sat * net_radiation)
                                            / (slope_of_sat + psy_const))

    # Accounting for negative net radiation and setting them to zero
    potential_evap = np.where(net_radiation <= 0, 0, potential_evap)

    # =====================================================================
    # Priestley-Taylor open water potential evapotranspiration (mm/day)
    # =====================================================================
    # Coverting net radiation to mm/day
    openwater_net_radiation = \
        (openwater_net_radiation * 0.0864) / latent_heat

    # Actual name: Open water potential evapotranspiration, Units:  mmd-1
    openwater_pot_evap = \
       pt_coeff_humid_arid * ((slope_of_sat * openwater_net_radiation) /
                              (slope_of_sat + psy_const))

    # Accounting for negative net radiation and setting them to zero
    openwater_pot_evap = np.where(openwater_net_radiation <= 0, 0,
                                  openwater_pot_evap)
    return potential_evap, openwater_pot_evap
```

CE02

Join Our Community

WaterGAP, freely available on GitHub, encourages collaboration among developers, fostering transparency and broadening participation in source code development. This model source code is developed, hosted and maintained on GitHub github.com/HydrologyFrankfurt/ReWaterGAP

Community engagement ensures WaterGAP meets diverse user needs, addressing real-world water challenges. Through our mailing list (will be communited on our Github page), our community enjoys benefits like updates, workshops, and conferences announcements.
Our workshops and conferences will offer practical training sessions aimed at enhancing users' comprehension of the model and enabling them to effectitvely use WaterGAP for their reseach.

CE01

18. Community Engagement

	Very Likely				Very Unlikely	Not sure
How likely is it that you will engage with a user community related to the WaterGAP model (e.g. via mailing list and workshop)	○	○	○	○	○	○

CE04

19. What has prohibited you from engaging with other community projects ?

CE03

Outlook for WaterGAP

WaterGAP can be run everywhere independent of host institutions (Goethe University Frankfurt and University of Bochum). We hope to provide sustainable research software which not only improves our understanding of the global water cycle but can be integrated with other models (e.g. groundwater or economic models) to drive impactful research.

MI01

20. What will you suggest to help improve the use, comprehension, and modification of this software to improve its long-term sustainability?

Thank you for completing this questionnaire!

We would like to thank you very much for helping us.

Your answers were transmitted, you may close the browser window or tab now.

M.Sc. Emmanuel Nyenah, Goethe University Frankfurt – 2024