Supplement of Geosci. Model Dev., 10, 873–888, 2017
http://www.geosci-model-dev.net/10/873/2017/
doi:10.5194/gmd-10-873-2017-supplement

Geoscientific Model Development
Open Access

Supplement of

Exploring new topography-based subgrid spatial structures for improving land surface modeling

Teklu K. Tesfa and Lai-Yung Ruby Leung

Correspondence to: Teklu K. Tesfa (teklu.tesfa@pnnl.gov)

Supplementary Materials

Tables

Table 1s: Algorithm applied to derive elevation-based SUs using the Local method.

Algorithm 1: Local method of subbasin discretization. Array BRK_i = [Elv_{min}, $Elv_{0.2}$, $Elv_{0.5}$, $Elv_{0.8}$, and Elv_{max}] denotes elevation values at the initial class breaks, where Elv_{min}, $Elv_{0.2}$, $Elv_{0.5}$, $Elv_{0.8}$, and Elv_{max} refer to the minimum elevation, elevation values at relative areas of 0.2, 0.5, and 0.8, and maximum elevation of the subbasin, respectively. Array R_i = [R_1, R_2, R_3, R_4] denotes the values of elevation range between consecutive BRK_i. Variables BRK_f denotes the final values of elevation at class breaks. Variable thr denotes the value of elevation threshold (100 m). Variable n denotes the number of Rs with values less than thr. Function GetFinalBRKs() denotes a function used to determine BRK_f by recursively merging Rs less than the thr with the neighboring Rs recursively.

```
For each Subbasin:
    Derive a hypsometric curve
    Determine elevation values at the BRK_i
    Calculate values of R_i between consecutive BRK_i
    Determine n

    If n == 0 // All values of R_i greater than the thr
        BRK_f = BRK_i
        R_f = R_i
    Else if R1 >= thr and R2 < thr and R3 < thr and R4 >= thr:
        If (R2 + R3) >= thr:
            BRK_f = [Elv_min, Elv_0.2, Elv_0.8, Elv_max] // Keep the body as separate class
        Else:
            BRK_f = [Elv_min, Elv_0.5, Elv_max] // Split the body into the head and tail
    Else if R1 >= thr and R2 < thr and R3 >= thr and R4 >= thr:
        BRK_f = [Elv_min, Elv_0.2, Elv_0.8, Elv_max] // Keep the body as separate class
    Else if R1 >= thr and R2 >= thr and R3 < thr and R4 >= thr:
        BRK_f = [Elv_min, Elv_0.2, Elv_0.8, Elv_max] // Keep the body as separate class
    Else:
        BRK_f = GetFinalBRKs(BRK_i, R_i, thr) // Call the recursive function

    Return BRK_f
```

Table 2s: Algorithm to determine the final class break values (BRK_f) by merging elevation ranges with less than the threshold value to the neighboring elevation ranges recursively.

Algorithm 2: To determine the final values of class breaks using recursive function GetFinalBRKs(). BRK_i, R_i, n, thr, denote the same variables as in Algorithm 1 (Table 1). Variables i and nn denote an index values of BRKs and the number of all Rs, respectively.

```
Function GetFinalBRKs(BRK_i, R_i, thr):
    Determine n
    Determine nn // number of all Rs
    Determine i // index of Rs with less than thr
    If n > 0 and nn > 1:
        Get the index (i)
        If i == 0: // R is at the beginning of the array
            R_i[i + 1] = R_i[i + 1] + R_i[i] // merge R with the next neighbor
            Update BRK_i
            Call GetFinalBRKs(BRK_i, R_i, thr) // This is a recursive call
        Else if i == nn: // R is at the end of the array
            R_i[i - 1] = R_i[i - 1] + R_i[i] //merge R with the previous neighbor
            Update BRK_i
            Call GetFinalBRKs(BRK_i, R_i, thr) //Recursive call
        Else: // merge with the smaller negibor
            If R_i[i - 1] > R_i[i + 1]
                R_i[i + 1] = R_i[i + 1] + R_i[i] // merge R with the next neighbor
                Update BRK_i
                Call GetFinalBRKs(BRK_i, R_i, thr) // This is a recursive call
            Else:
                R_i[i - 1] = R_i[i - 1] + R_i[i] // merge R with the previous neighbor
                Update BRK_i
                Call GetFinalBRKs(BRK_i, R_i, thr) // This is a recursive call

    Return BRK_i
```

Table 3s: Comparing the SUs of the Global method generated using 3% area threshold and Subbasin representations against the original PRISM grid representation using statistical summary of precipitation and surface temperature calculated over the study domain

Representation	**Precipitation (mm)**		**Temperature (C°)**	
	Average	**Standard deviation**	**Average**	**Standard deviation**
Subbasin	669.036	459.479	7.179	2.525
Non-geo-located subgrid units using the Global method	728.95	509.79	7.09	2.71
Original PRISM Grid	717.021	519.523	6.935	2.681

Table 4s: Comparing the SUs of the Global method generated using 3% area threshold and Subbasin representations against the original NDVI grid representations using statistical summary of spring and summer NDVI values calculated over the study domain

Representation	**NDVI values (spring)**		**NDVI values (summer)**	
	Average	**Standard deviation**	**Average**	**Standard deviation**
Subbasin	5804.00	1735.03	5128.87	1967.29
Non-geo-located subgrid units using the Local method	5909.03	1881.78	5351.59	2115.09
Original NDVI Grid	5810.02	2159.39	5207.00	2342.65

Figures

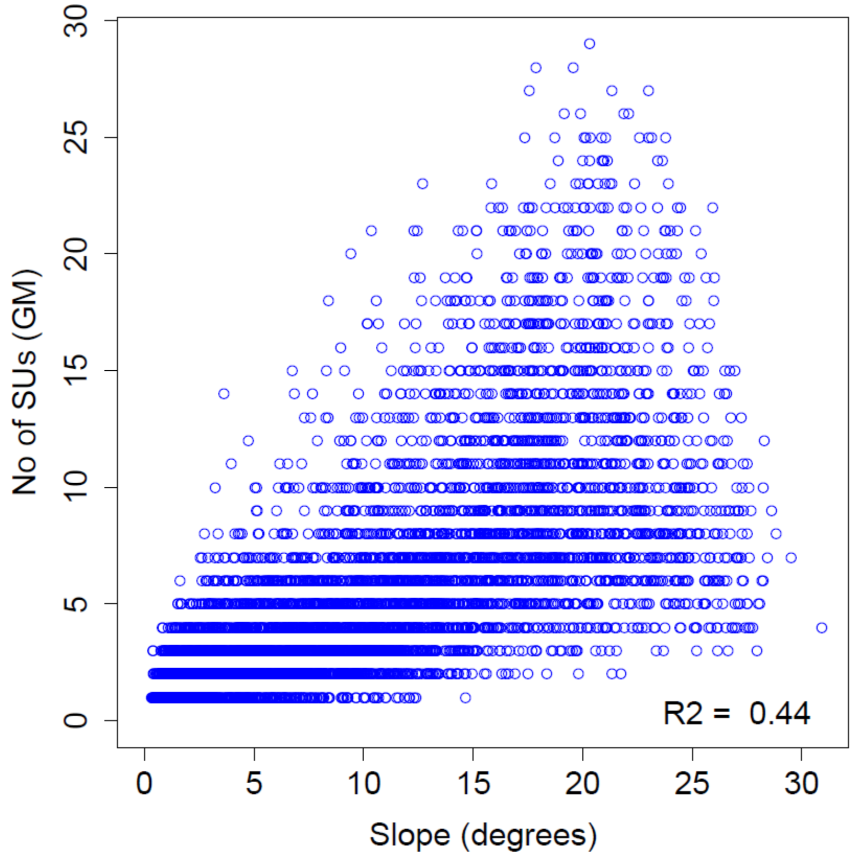

Figure 1s: Number of geo-located subgrid units per subbasin from the Global method based on the combination of topographic elevation and slope at area threshold value of 1% compared against values of average slope of the subbasins.

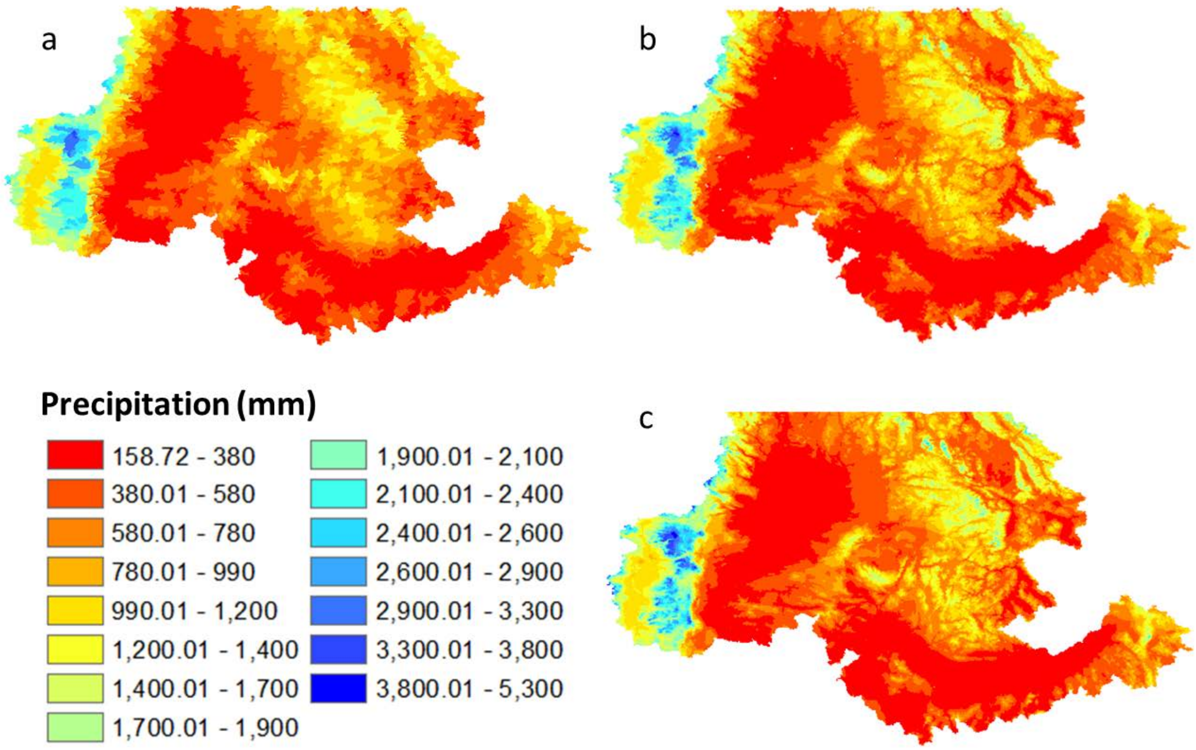

Figure 2s: PRISM 30 year normal precipitation represented using the subbasins (a) and non-geo-located SUs from the Global method using 3% area threshold (b) compared to those of the original PRISM grids (c). The 535 Canadian territory of the study area is not represented in the PRISM dataset.

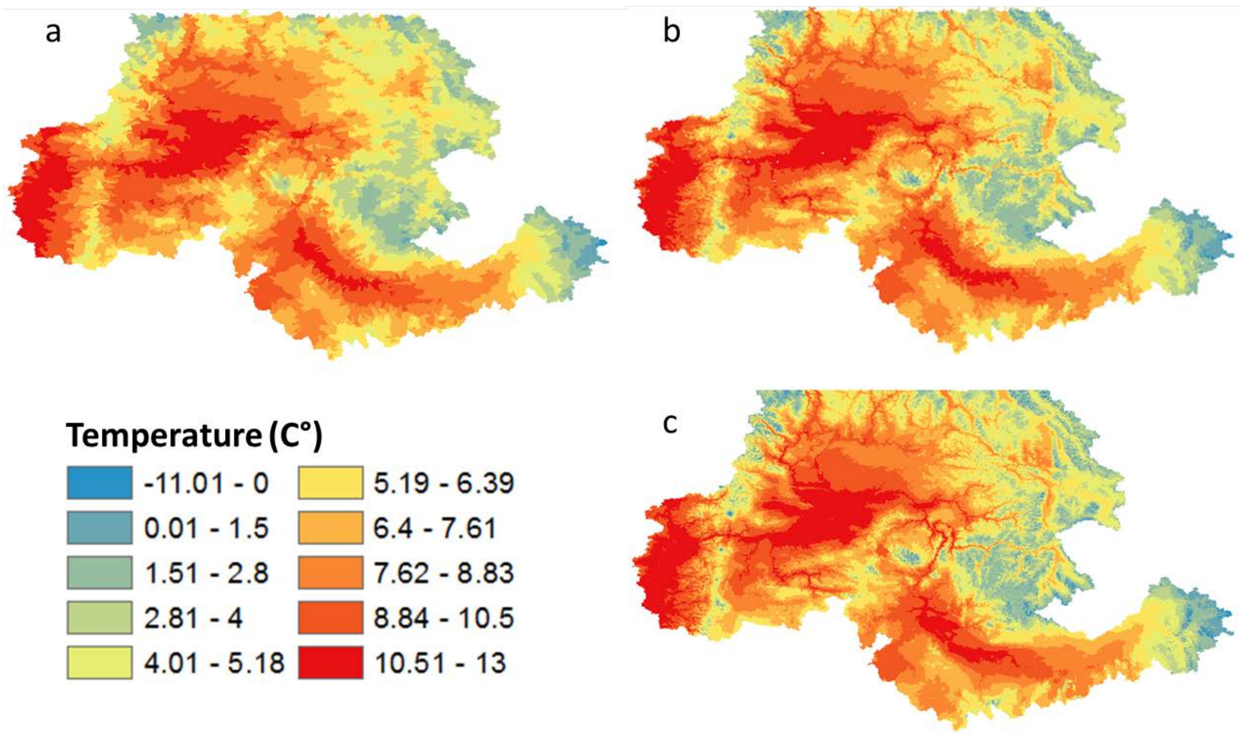

Figure 3s: PRISM 30 year normal precipitation represented using the subbasins (a) and non-geo-located SUs from the Global method using 3% area threshold (b) compared to those of the original PRISM grids (c). The 535 Canadian territory of the study area is not represented in the PRISM dataset