First steps of climate modelling

marcellinus jerricho

29/11/2021

In this activity, we will continue to build our own model for the temperature of the surface of the Earth and various layers of its atmosphere. Following the development cycle for modeling, we started with a simplified initial setup and will incrementally add more features to make the model more realistic.

Adding More Layers of Atmosphere

In reality, we know that the composition and temperature of the atmosphere varies with height. To improve our model, we would like to include multiple layers of atmosphere, each with its own emissivity/absorptivity and temperature. We can extend our single-layer atmosphere model using the diagram in Figure 2.

For a 2-layer atmosphere model, the amount of radiation absorbed by the surface of the Earth is equal to the portion of radiation of the Sun that is not reflected away plus the amount of radiation emitted downward by the lower layer of atmosphere plus the portion of radiation emitted downward by the upper layer of the atmosphere that is not absorbed by the lower layer of atmosphere on its way down to the surface. Mathematically, this looks like

$$A_{surf} = (1 - \alpha)R_{sun} + R_{atm1} + (1 - \epsilon_1)R_{atm2}.$$

Similarly, the amount of radiation absorbed by the lower layer of the atmosphere is equal to a portion of the radiation emitted by the Earth's surface plus a portion of the radiation emitted downward by the upper layer of atmosphere. Mathematically, this looks like

$$A_{atm1} = \epsilon_1 R_{surf} + \epsilon_1 R_{atm2}.$$

Finally, the amount of radiation absorbed by the upper layer of the atmosphere is equal to a portion of the radiation emitted upward by the lower layer of atmosphere plus a portion of the radiation from the Earth's surface that did not get absorbed by the lower layer of atmosphere. Mathematically, this looks like

$$A_{atm2} = \epsilon_2 R_{atm1} + \epsilon_2(1 - \epsilon_1)R_{surf}.$$

Similar to the previous iteration, we know that $R_{surf} = \sigma T_{surf}^4$, $R_{atm1} = 2\epsilon_1\sigma T_{atm1}^4$, and $R_{atm2} = 2\epsilon_2\sigma T_{atm2}^4$, which means we can write out formulas for the rates of change of the temperatures of the surface and each layer of atmosphere in terms of their current temperatures and the parameters:

$$\dot{T}_{surf}(t) = \frac{A_{surf}(t) - R_{surf}(t)}{C_{surf}} = \frac{(1-\alpha)R_{sun} + \epsilon_1\sigma T_{atm1}(t)^4 + (1-\epsilon_1)\epsilon_2\sigma T_{atm2}(t)^4 - \sigma T_{surf}(t)^4}{C_{surf}} \tag{1}$$

$$\dot{T}_{atm1}(t) = \frac{A_{atm1} - R_{atm1}}{C_{atm1}} = \frac{\epsilon_1\sigma T_{surf}(t)^4 + \epsilon_1\epsilon_2\sigma T_{atm2}(t)^2 - 2\epsilon_1\sigma T_{atm1}(t)^4}{C_{atm}} \tag{2}$$

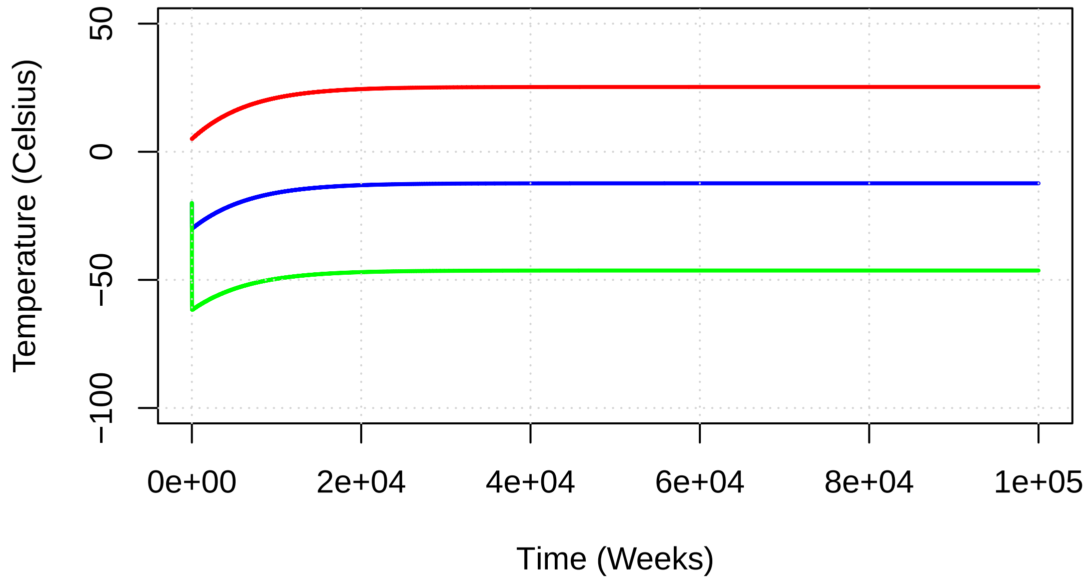
$$\dot{T}_{atm2}(t) = \frac{A_{atm2} - R_{atm2}}{C_{atm2}} = \frac{\epsilon_2\epsilon_1\sigma T_{atm1}(t)^4 + \epsilon_2(1-\epsilon_1)\sigma T_{surf}(t)^4 - 2\epsilon_2\sigma T_{atm2}(t)^4}{C_{atm2}}. \tag{3}$$

5th Iteration: Double layer atmosphere with constant albedo

Update the new dynamic model to include a second layer of atmosphere with absorptivity equal to 0.5.

```r
# Define a function to calculate the rates of change of the different
# temperatures in terms of the parameters and current temperatures.
change_rates <- function(T_surf, T_atm1, T_atm2, C_surf, C_atm1, C_atm2,
                         R_sun, alpha, sigma, epsilon1, epsilon2){

   A_surf <- (1-alpha)*R_sun + epsilon1*sigma*T_atm1^4
                             + epsilon2*(1-epsilon1)*sigma*T_atm2^4
   R_surf <- sigma*T_surf^4

   A_atm1 <- epsilon1*sigma*T_surf^4 + epsilon2*epsilon1*sigma*T_atm2^4
   R_atm1 <- 2*epsilon1*sigma*T_atm1^4

   A_atm2 <- epsilon2*(1-epsilon1)*sigma*T_surf^4 + epsilon2*epsilon1*sigma*T_atm1^4
   R_atm2 <- 2*epsilon2*sigma*T_atm2^4

   T_dot_surf <- (A_surf - R_surf) / C_surf
   T_dot_atm1 <- (A_atm1 - R_atm1) / C_atm1
   T_dot_atm2 <- (A_atm2 - R_atm2) / C_atm2

   return( c(T_dot_surf, T_dot_atm1, T_dot_atm2) )
}

# Set up the relevant parameters.
R_sun <- 342 #W/m^2 or J/s/m^2
sigma <- 5.67 * 10^(-8) #W/m^2/K^4 or J/s/m^2/K^4
alpha <- 0.3 #fraction
epsilon1 <- 0.8 #fraction
epsilon2 <- 0.5 #fraction
C_surf <- 1.176 * 10^(10) #J/K
C_atm <- 1.4 * 10^7 #J/K

# Assume the heat capacities of the two layers of atmosphere are equal.
C_atm1 <- 0.5 * C_atm #J/K
C_atm2 <- 0.5 * C_atm #J/K

# Set up the initial temperatures.
T_surf_0 <- 5 #C
T_atm1_0 <- -20 #C
T_atm2_0 <- -20 #C

# Set up a sequence of times to calculate the temperatures. Since the temperature
# of the Earth changes slowly, increment the times by weeks instead of seconds.
num_steps <- 100000
times_weeks <- seq(num_steps)

# Since the units for the rates of change are J/s and K/s, we
# need to express convert weeks to seconds for the calculations.
times <- times_weeks * 60*60*24*7

# Create dummy lists to store the temperatures over time.
```

```r
Ts_surf <- matrix(0,1,num_steps)
Ts_atm1 <- matrix(0,1,num_steps)
Ts_atm2 <- matrix(0,1,num_steps)

# Insert the initial temperatures into the storage lists.
Ts_surf[1] <- T_surf_0
Ts_atm1[1] <- T_atm1_0
Ts_atm2[1] <- T_atm2_0

# Iteratively fill in the temperatures in the storage lists.
for(t in 2:num_steps){

  # Calculate the time step.
  delta <- times[t] - times[t-1]

  # Convert the previous temperatures from Celsius to Kelvin.
  T_surf_K <- Ts_surf[t-1] + 273
  T_atm1_K <- Ts_atm1[t-1] + 273
  T_atm2_K <- Ts_atm2[t-1] + 273

  # Input the previous temperatures into the change_rates function.
  rates <- change_rates(T_surf_K, T_atm1_K, T_atm2_K, C_surf,
                        C_atm1, C_atm2, R_sun, alpha,
                        sigma, epsilon1, epsilon2)

  # Calculate the current temperatures using the previous temperatures,
  # time step, and rates of change.
  Ts_surf[t] <- Ts_surf[t-1] + delta * rates[1]
  Ts_atm1[t] <- Ts_atm1[t-1] + delta * rates[2]
  Ts_atm2[t] <- Ts_atm2[t-1] + delta * rates[3]
}

# Plot the results
plot(times_weeks,Ts_surf,type="l",ylim=c(-100,50),col="red",lwd=2,
     xlab="Time (Weeks)",ylab="Temperature (Celsius)")
lines(times_weeks,Ts_atm1,type="l",col="blue",lwd=2)
lines(times_weeks,Ts_atm2,type="l",col="green",lwd=2)
grid()
```




```r
# Print the final temperatures
message(sprintf("Temperature of Surface (in deg Cel): %3.1f",Ts_surf[num_steps]))
```

```
## Temperature of Surface (in deg Cel): 25.3
```

```r
message(sprintf("Temperature of Lower Atmosphere (in deg Cel): %3.1f",Ts_atm1[num_steps]))
```

```
## Temperature of Lower Atmosphere (in deg Cel): -12.3
```

```r
message(sprintf("Temperature of Upper Atmosphere (in deg Cel): %3.1f",Ts_atm2[num_steps]))
```

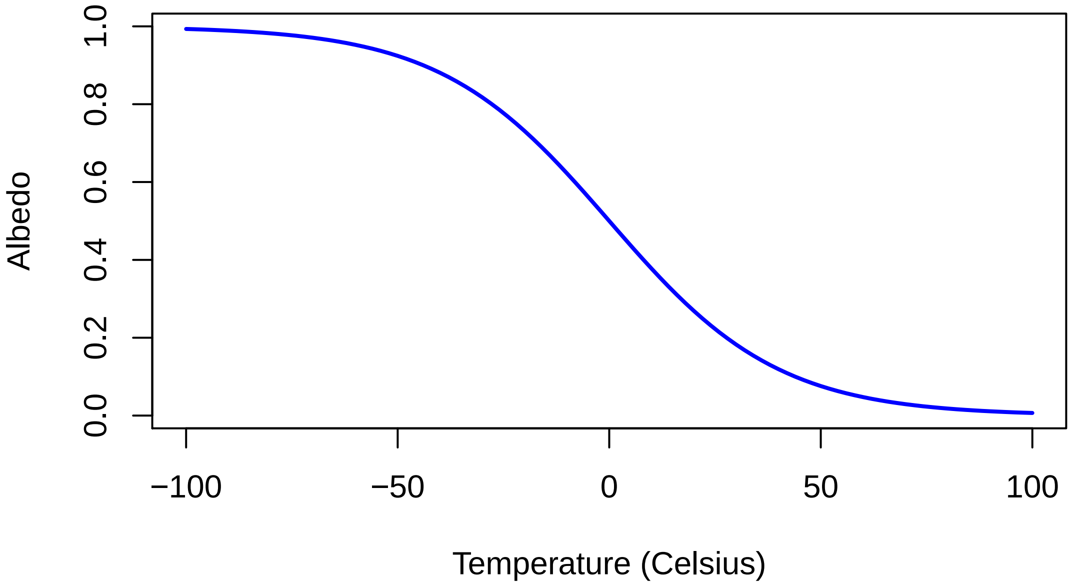
```
## Temperature of Upper Atmosphere (in deg Cel): -46.4
```

Additional Benefits

Modifying our approach from a static to a dynamic model allowed us to examine the long-term behavior of a multilayered atmosphere more easily, but there are additional upshots to this approach! Some of the parameters that we assumed to be constant can be expressed as functions of temperature as well. For instance, because much of the albedo for the Earth's surface comes from ice and snow cover, we should expect the Earth's surface to have less albedo when its temperature is hot and more albedo when its temperature is cold. Since albedo ranges between 0 and 1, an expression for albedo as a function of the Earth's surface temperature might look something like the following:

$$\alpha(T_{surf}) = \frac{1}{1 + e^{(1/20)T_{surf}}}$$

whose graph is plotted below:



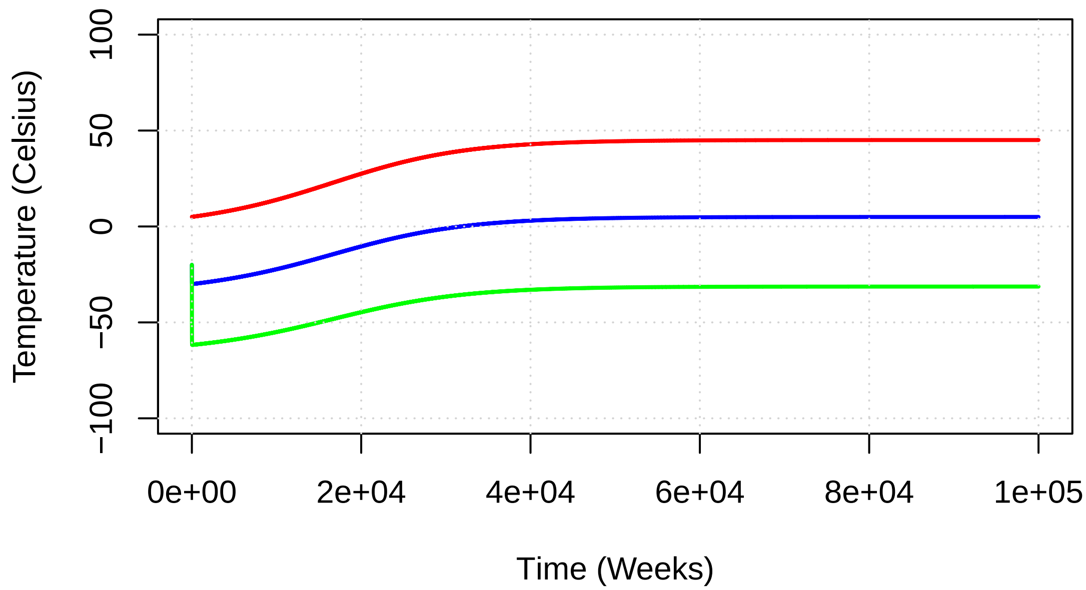
6th Iteration: Temperature-dependent albedo

Update the model to include temperature dependent albedo.

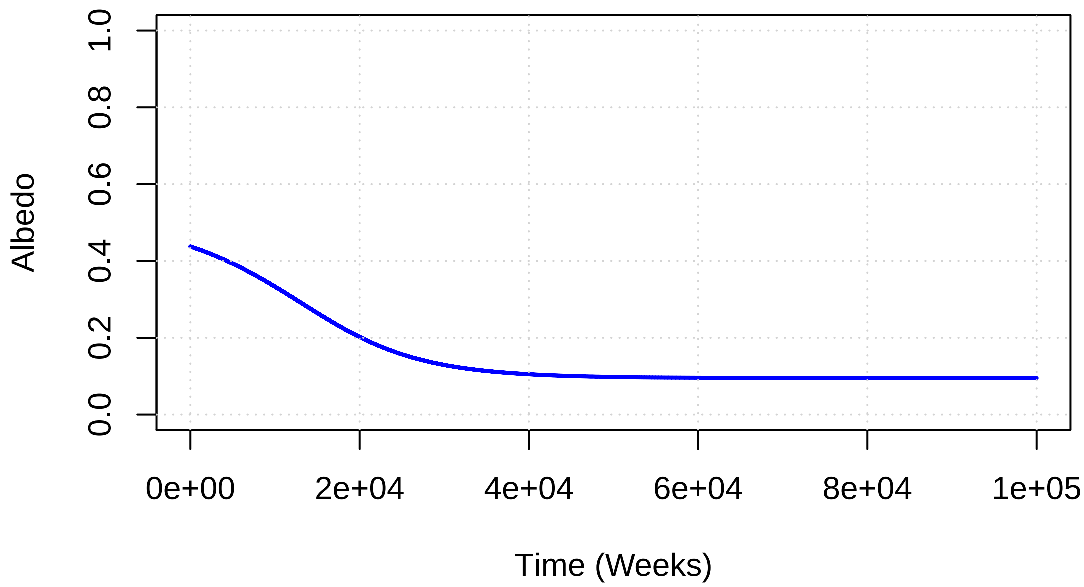
```
# Define a function to calculate the rates of change of the different
# temperatures in terms of the parameters and current temperatures.
change_rates <- function(T_surf, T_atm1, T_atm2, C_surf, C_atm1, C_atm2,
                         R_sun, alpha, sigma, epsilon1, epsilon2){

   A_surf <- (1-alpha)*R_sun + epsilon1*sigma*T_atm1^4
                             + epsilon2*(1-epsilon1)*sigma*T_atm2^4
   R_surf <- sigma*T_surf^4

   A_atm1 <- epsilon1*sigma*T_surf^4 + epsilon2*epsilon1*sigma*T_atm2^4
   R_atm1 <- 2*epsilon1*sigma*T_atm1^4

   A_atm2 <- epsilon2*(1-epsilon1)*sigma*T_surf^4 + epsilon2*epsilon1*sigma*T_atm1^4
   R_atm2 <- 2*epsilon2*sigma*T_atm2^4

   T_dot_surf <- (A_surf - R_surf) / C_surf
   T_dot_atm1 <- (A_atm1 - R_atm1) / C_atm1
   T_dot_atm2 <- (A_atm2 - R_atm2) / C_atm2

   return( c(T_dot_surf, T_dot_atm1, T_dot_atm2) )
}

# Set up the relevant parameters.
R_sun <- 342 #W/m^2 or J/s/m^2
sigma <- 5.67 * 10^(-8) #W/m^2/K^4 or J/s/m^2/K^4
epsilon1 <- 0.8 #fraction
epsilon2 <- 0.5 #fraction
C_surf <- 1.176 * 10^(10) #J/K
C_atm <- 1.4 * 10^7 #J/K
```

```r
# Assume the heat capacities of the two layers of atmosphere are equal.
C_atm1 <- 0.5 * C_atm #J/K
C_atm2 <- 0.5 * C_atm #J/K

# Set up the initial temperatures and albedo.
T_surf_0 <- 5 #C
T_atm1_0 <- -20 #C
T_atm2_0 <- -20 #C

# Calculate initial albedo.
alpha_0 <- 1/(1+exp(T_surf_0/20))

# Set up a sequence of times to calculate the temperatures.
# Since the temperature of the Earth changes slowly, increment
# the times by weeks instead of seconds.
num_steps <- 100000
times_weeks <- seq(num_steps)

# Since the units for the rates of change are J/s and K/s, we
# need to express convert weeks to seconds for the calculations.
times <- times_weeks * 60*60*24*7

# Create dummy lists to store the temperatures and albedo over time.
Ts_surf <- matrix(0,1,num_steps)
Ts_atm1 <- matrix(0,1,num_steps)
Ts_atm2 <- matrix(0,1,num_steps)
albedos <- matrix(0,1,num_steps)

# Insert the initial temperatures into the storage lists.
Ts_surf[1] <- T_surf_0
Ts_atm1[1] <- T_atm1_0
Ts_atm2[1] <- T_atm2_0
albedos[1] <- alpha_0

# Iteratively fill in the temperatures in the storage lists.
for(t in 2:num_steps){

  # Calculate the time step.
  delta <- times[t] - times[t-1]

  # Convert the previous temperatures from Celsius to Kelvin.
  T_surf_K <- Ts_surf[t-1] + 273
  T_atm1_K <- Ts_atm1[t-1] + 273
  T_atm2_K <- Ts_atm2[t-1] + 273

  # Get previous albedo.
  alpha <- albedos[t-1]

  # Input the previous temperatures into the change_rates function.
  rates <- change_rates(T_surf_K, T_atm1_K, T_atm2_K, C_surf,
                        C_atm1, C_atm2, R_sun, alpha,
                        sigma, epsilon1, epsilon2)
```

```r
  # Calculate the current temperatures using the previous temperatures,
  # time step, and rates of change.
  Ts_surf[t] <- Ts_surf[t-1] + delta * rates[1]
  Ts_atm1[t] <- Ts_atm1[t-1] + delta * rates[2]
  Ts_atm2[t] <- Ts_atm2[t-1] + delta * rates[3]

  # Calculate current albedo.
  albedos[t] <- 1/(1+exp(Ts_surf[t-1]/20))
}

# Plot the results
plot(times_weeks,Ts_surf,type="l",ylim=c(-100,100),col="red",lwd=2,
     xlab="Time (Weeks)",ylab="Temperature (Celsius)")
lines(times_weeks,Ts_atm1,type="l",col="blue",lwd=2)
lines(times_weeks,Ts_atm2,type="l",col="green",lwd=2)
grid()
```




```r
# Plot the albedos
plot(times_weeks,albedos,type="l",ylim=c(0,1),col="blue",lwd=2,
     xlab="Time (Weeks)",ylab="Albedo")
grid()
```




```r
# Print the final temperatures
message(sprintf("Temperature of Surface (in deg Cel): %3.1f",Ts_surf[num_steps]))
```

```
## Temperature of Surface (in deg Cel): 45.1
```

```r
message(sprintf("Temperature of Lower Atmosphere (in deg Cel): %3.1f",Ts_atm1[num_steps]))
```

```
## Temperature of Lower Atmosphere (in deg Cel): 5.0
```

```r
message(sprintf("Temperature of Upper Atmosphere (in deg Cel): %3.1f",Ts_atm2[num_steps]))
```

```
## Temperature of Upper Atmosphere (in deg Cel): -31.3
```

```r
message(sprintf("Albedo: %3.2f",albedos[num_steps]))
```

```
## Albedo: 0.10
```