Simulation of Radiation balance of the Earth

Daniel Stefanik

1. Explanation of aim

The aim of this exercise was to create model of physical phenomenon - global radiation budget of the Earth. Implementation of computer simulation helps to understand the mechanisms describing the phenomenon and it theoretical concept. Another objective is to compare results of no atmosphere and with atmosphere calculations. Next step is to find switch points of mean temperature for glaciations mechanism transitions.

2. Used tools

Simulation was created in python language using Jupyter Notebook application. Libraries and their use:

- *pylab* - creating diagram of relationship between mean temperature and solar constant,
- *display* from *IPython* - proper display of the plot,
- *fsolve* from *scipy.optimize* - library to solve set of nonlinear equations,
- *numpy* - operations on lists.

3. Results

a. Calculation of mean Earth temperature assuming no atmosphere

Simple calculations assuming there is no atmosphere. Result of calculating set of energy balance equations provided result of -18.33 Celsius degrees.

b. Calculation of mean Earth temperature taking atmosphere

Set of energy balance equations with atmosphere are nonlinear, to solve them there were used *fsolve* library. Below results:

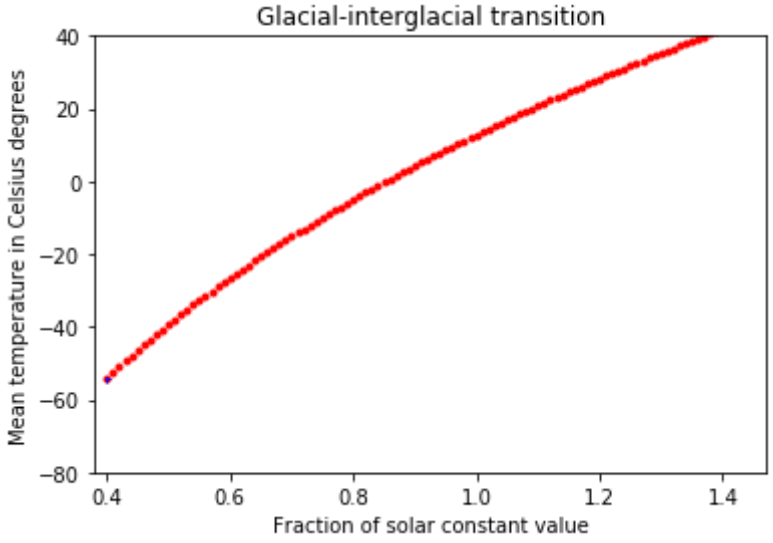
- mean temperature of the atmosphere in Celsius degrees -24.63,
- mean earth surface temperature in Celsius degrees 12.84.

c. Comparison of above calculations

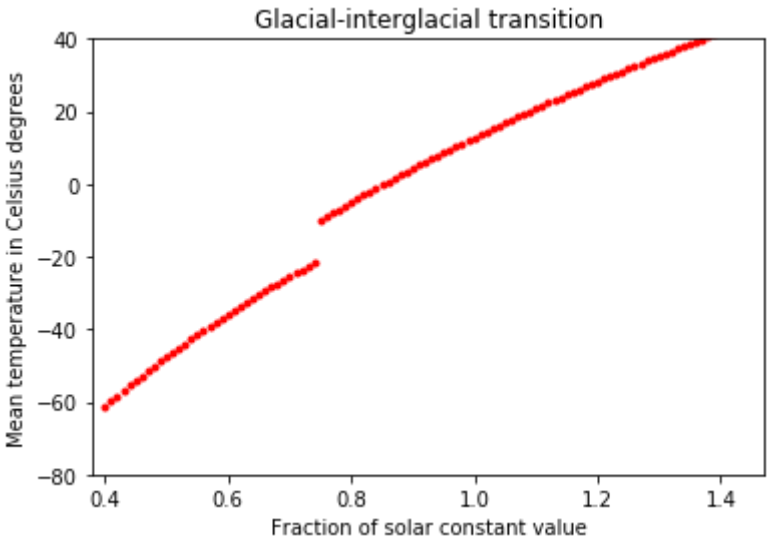
We can see that when calculating temperature without atmosphere the result is very low. This effect is caused by no influence of atmosphere in temperature of earth surface. We can observe that taking atmosphere effect into calculation increase the temperature. It is called the greenhouse effect. Atmosphere increases temperature of earth surface because it reflects some of long-wave radiation.

d. Relationship between mean temperature and solar constant and glaciations mechanism transitions

When considering variable fraction of solar constant (from 0.4 to 1.4) we can observe that the mean temperature is rising what is presented on diagram below.



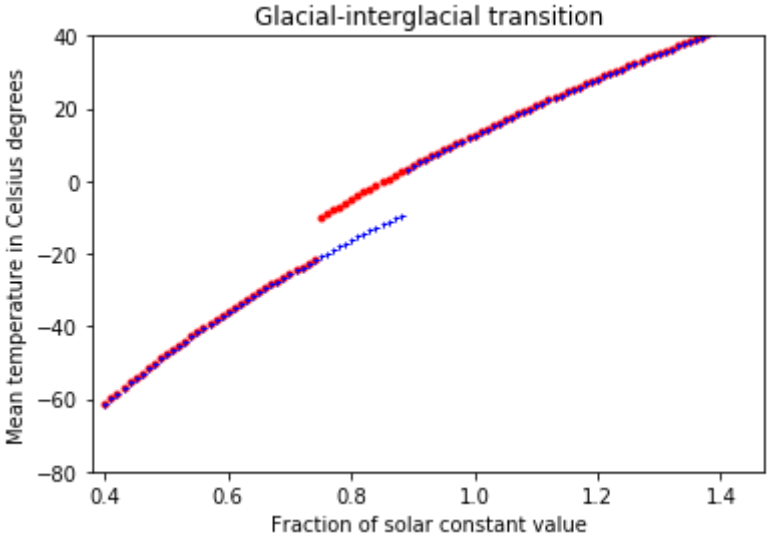
When temperature is low then glacial-interglacial transitions cause increase of surface albedo for short wave radiation. In simulation is assumed that critical temperature is -10 Celsius degrees and parameter of surface albedo for short wave radiation is changing between 0.19 (above critical temperature) and 0.65 (below critical temperature). We can observe in diagram below that there is switch point of temperature caused by the glacial-interglacial transitions.



e. Solar constant values associated with glacial-interglacial transition of the Earth system

When simulating decreasing and increasing of fraction of solar constant value we can observe two switch point of temperature:

- for fraction of solar constant equal to 0.75 (solar constant = 1024.50 *W/m*^2) the temperature changed instantly (red line) this effect is caused by increase of albedo for short wave radiation for critical temperature,
- the second change of temperature we can observe only after the fraction of solar constant is equal to 0.87 (solar constant = 1188.40 *W/m*^2).



4. Conclusions

Energy balance of the earth calculated assuming no atmosphere leads to incorrect result of very low mean temperature of the earth surface. Taking atmosphere influence into calculations which consist the energy emission reflected back to the earth give us proper result. Observation of changing solar constant value leads to conclusions that when reaching some of its critical values can instantly decrease or increase the mean temperature of earth surface what is caused by the glacial-interglacial transition of the Earth system.

5. Code listening

```
import numpy as np
from scipy.optimize import fsolve

% matplotlib inline
import time
import pylab as pl
from IPython import display

# PSl - Power of solar radiation arriving to the Earth (short wave radiation)
# Pz - Power of radiation emitted from Earth (long wave radiation)
# A - mean albedo of the Earth surface
# S - solar constant
# PowZ - area of the Earth
# sbc -Stefan-Boltzmann constant

# PSl = S * (PowZ/4) *(1 - A)
# Pz = sbc * (T**4) * PowZ
# Pz = PSl

# sbc * (T**4) * PowZ = S * (PowZ/4) *(1 - A)
# (T**4) = (S * (PowZ/4) * (1 - A))/(sbc*PowZ) = (S * (1 - A))/(4 * sbc)

# Diagram methods


def doPlot(x, y, col):
   pl.plot(x, y, col, markersize=3)
   display.clear_output(wait=True)
   display.display(pl.gcf())
   time.sleep(0.1)


def setRanges():
   pl.xlim(S_range_left * 0.95, S_range_right * 1.05)
   pl.ylim(-80, 40)
   pl.xlabel('Fraction of solar constant value')
   pl.ylabel('Mean temperature in Celsius degrees')
   pl.title('Glacial-interglacial transition')
```

```
def changeColor(i, val):
    if i >= val:
        return 'b+'
    else:
        return 'ro'


# Computation methods

def k_to_c(k):
    return k - 273.15

# No atmosphere methods


def withoutAtmosphere(S, A, sbc):
    return pow((S * (1 - A)) / (4 * sbc), 0.25)

# Taking atmosphere methods


def withAtmosphereEquations(p):
    Ts, Ta = p
    e1 = (-sw_ta) * (1 - sw_as) * S / 4 + c * (Ts - Ta) + sbc * (Ts ** 4) * (1 - lw_aa) -
sbc * (Ta ** 4)
    e2 = -(1 - sw_aa - sw_ta + sw_as * sw_ta) * S / 4 - c * (Ts - Ta) - sbc * (Ts ** 4) *
(
            1 - lw_ta - lw_aa) + 2 * sbc * (Ta ** 4)
    return e1, e2


def withAtmosphere():
    return fsolve(withAtmosphereEquations, (0.0, 0.0))


def changeAs(t):
    if (t < Tc):
        return sw_as_init_r
    else:
        return sw_as_init

# Variable initiation

# PSl = 0.0
# Pz = 0.0
A = 0.3
S_init = 1366.0  # W/m 2
S = S_init  # W/m 2
# PowZ = 0.0
sbc = 5.67 * pow(10.0, -8)  # W/m^2*K^4
```

```
# Short-wave radiation
sw_as_init = 0.19
sw_as_init_r = 0.65
sw_as = sw_as_init
sw_ta = 0.53
sw_aa = 0.30

# Long-wave radiation
lw_ta = 0.06
lw_aa = 0.31

c = 2.7  # Wm^-2 K^-1

# S in range 0.8 to 1.2 S
S_range_left = 0.4
S_range_right = 1.4
S_range_step = 0.01

Tc = -10  # C degrees

# No atmosphere
T = withoutAtmosphere(S, A, sbc)
print("No atmosphere in Celsius degrees: " + str(k_to_c(T)) + ".")

# Taking atmosphere
Ts, Ta = withAtmosphere()
print("Mean temperature of the atmosphere in Celsius degrees  " + str(k_to_c(Ta)) + ".")
print("Mean surface temperature in Celsius degrees  " + str(k_to_c(Ts)) + ".")

setRanges()

arr = list(np.arange(S_range_left, S_range_right, S_range_step))
iterator = list(arr)
iterator.reverse()
iterator.extend(arr)

sp1_val = None
sp2_val = None
sp1_frac = None
sp2_frac = None
sp1_temp = None
sp2_temp = None

# Main Loop
for i in range(len(iterator)):
   S = iterator[i] * S_init
   Ts, Ta = withAtmosphere()
   TaC = k_to_c(Ta)
   TsC = k_to_c(Ts)
   sw_as = changeAs(TsC)

   if (sw_as != sw_as_init) and (sp1_val is None):
       sp1_val = S
```

```
        sp1_frac = iterator[i]
        sp1_temp = TsC

    if sw_as != sw_as_init:
        sp2_val = S
        sp2_frac = iterator[i]
        sp2_temp = TsC

    col = changeColor(i, len(iterator) / 2)
    doPlot(iterator[i], TsC, col)
    print("S: " + str(S) + " TaC: " + str(TaC) + " TsC: " + str(TsC) + " Ta: " + str(Ta)
+ " Ts: " + str(Ts))

print("sp1_val: " + str(sp1_val) + " W/m^2 sp1_frac: " + str(sp1_frac) + " temp1: " +
str(sp1_temp))
print("sp2_val: " + str(sp2_val) + " W/m^2 sp2_frac: " + str(sp2_frac) + " temp2: " +
str(sp2_temp))
```