ESM for Lunch

PLASIM – a toy ESM for quick tests

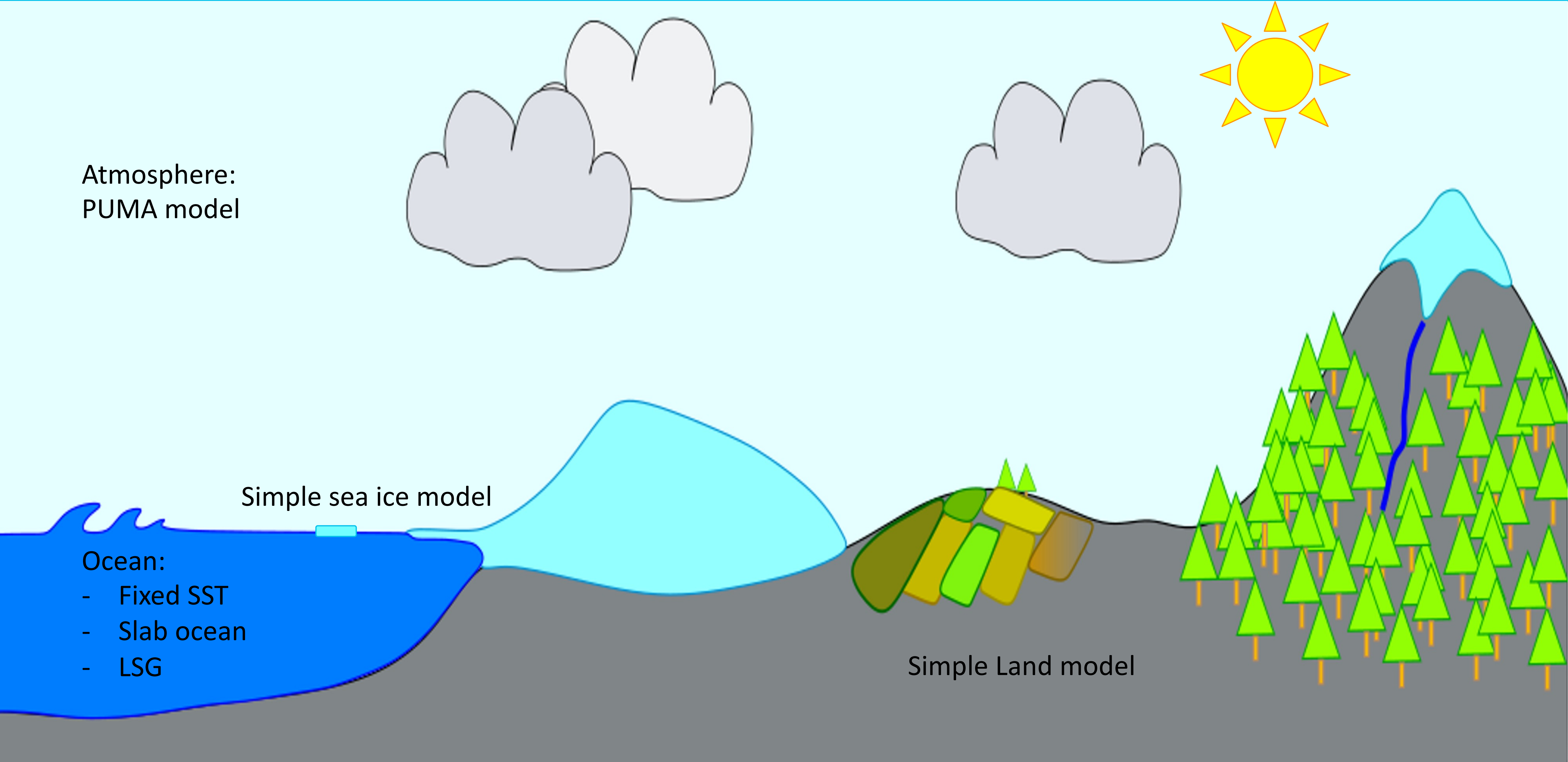
Claudia Wieners

What is PLASIM

- Intermediate complexity model

- Basically, a “Mini-ESM”, can run easily on Gemini (probably Lorenz too)

PLASIM model overview



What is PLASIM

- Intermediate complexity model

- Basically, a “Mini-ESM”, can run easily on Gemini (probably Lorenz too)

- Resolution:
  Atm: 32-64 latitudes, twice as many longitudes (i.e. ca 5 degrees – ca 2.5 degrees)
  Ocean (LSG): roughly 5 degrees always

- Used in MSc course Earth System Modelling; manual will be put on our wiki

- Runtime: some minutes per model year (depending on settings)

- Reasonable mean state + climate change (CO2 only)
- Not so reasonable for e.g. ENSO

The formal structure of PLASIM

Rough overview

Formal structure of PLASIM

PLASIM model contains (among others):
-- actual model code (FORTRAN)
-- codes to make things work, e.g.
  - compile model -> most.x
  - postprocess model output
-- namelists, provide parameters after compilation

Overall workflow of a simulation

Preparation

Adapt code [if needed]

Compile
Adjust input parameters in namelists [if needed]

Submit simulation to computation cluster queue

After simulation is done: postprocess and analyse.

Setting up a simulation…

Preparation

-- set up environment (load modules)

-- install plasim, configure

Adapt code (if needed)

-- Typically, you won't do this

-- might do so for running e.g. simulation with solar geoengineering…

Compilation

-- Compilation: translate human-readable code into computer-readable code

-- Use most.x to make basic settings

-> generates folder PLASIM/plasim/run with all material to run simulation

-- if needed, adjust input parameters through namelists

Setting up a simulation…

Job submission

-- adjust run script if needed
-- submit to compute nodes
-- monitor whether simulation has started / is finished

Postprocessing

-- output data is not readable for humans!
-- postprocessor “pumaburn” (in PLASIM folder) can convert output into netcdf format
-- netcdf data is readable by python and matlab.

Analyse ☺

Working with PLASIM (gemini)

Preparation

Settings in Gemini

```
module load mpi/openmpi-x86_64
module load sge/2011
```

(can fix that in bash script, see manual)

Installing PLASIM

```
git clone https://github.com/HartmutBorth/PLASIM.git
```

-- Then a folder PLASIM is made with the model code etc in it.

-- Go into the folder and type

```
./configure.sh
```

Configuration and Compilation

Set configurations

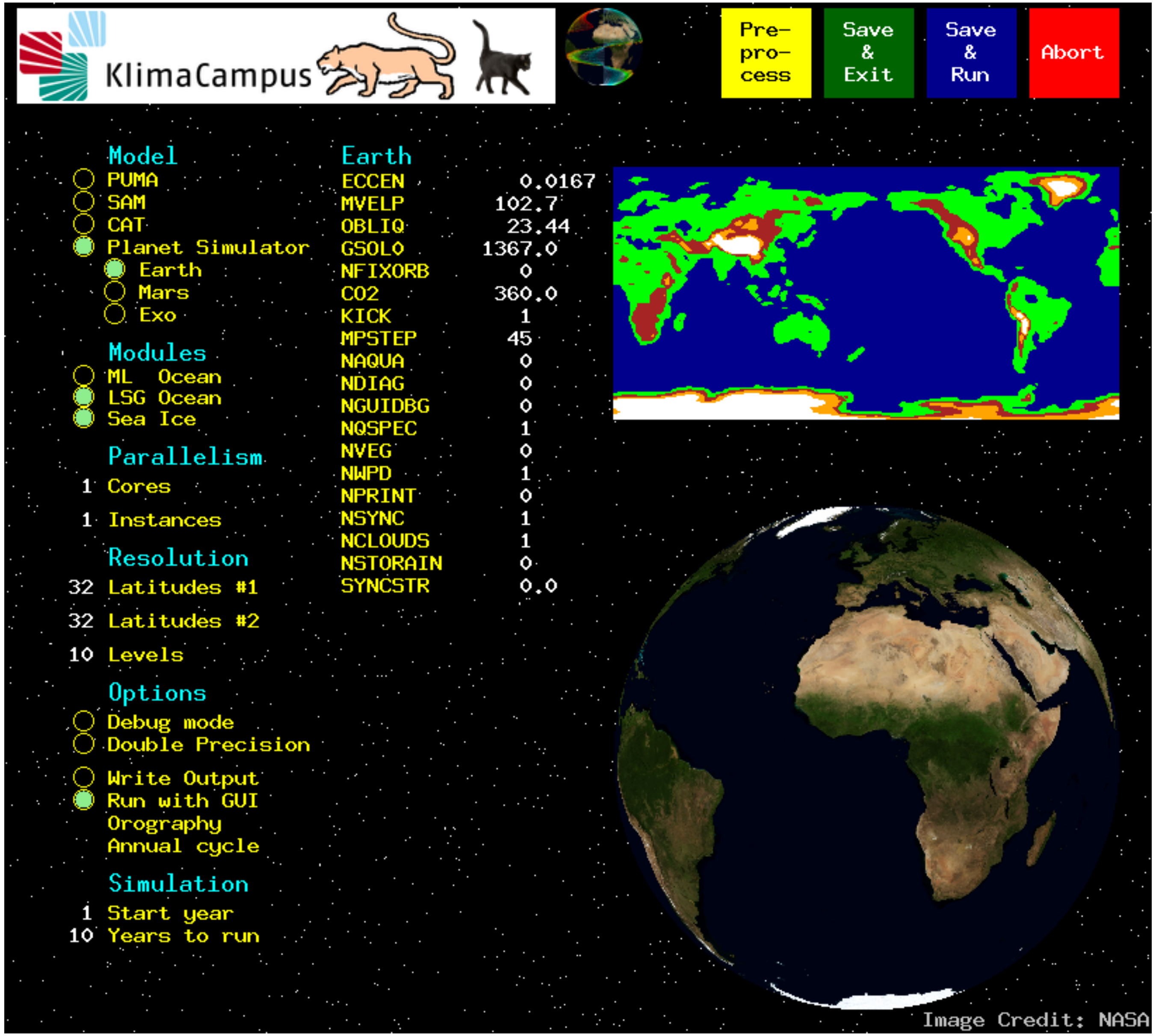
`./most.x`

opens an interface where you can make choices, for example…

- Model type -> choose PLASM
- Which ocean and sea ice model
  -> need to also set namelists and some parameters, see manual
- How many cores
- How many model years
- Whether to write output or just look at it in interface (-> write output!)
- Planetary parameters (insolation, CO_2) …

⇒ Clicking “save and exit” compiles the model + creates a folder ”plasim/run” with executable, run script, namelists and input data

Configuration and Compilation



Namelists

Namelists contain model parameters that can be read in after compilation
-> radmod_namelist contains CO_2
-> planet_namelist contains solar radiation (GSOL)

To run with ocean model LSG, open oceanmod_namelist and set
NOCEAN=1
NLSG=1

To run with slab ocean model, open oceanmod_namelist and set
NOCEAN=1
NLSG=0
NHDIFF=1 (switch on mixing)
HDIFFK = 4.E4 (low res), 4.5E4 (high res) (mixing parameter)

To run with sea ice model, open icemod_namelist and set NICE=1

Some additional LSG issues

To combine the LSG ocean model with higher resolutions in the atmosphere, must adjust ocean code before compilation:

```
cd ~/PLASIM
./cleanplasim
cd ~/PLASIM/plasim/src/
vi cpl.f90
```

Replace

`parameter(nxa=64,nya=32)`

By

`parameter(nxa=96,nya=48)` (for 48 latitudes)

Or

`parameter(nxa=128,nya=64)` (for 64 latitudes)

Submitting your job

Open most_plasim_run (jobscript)

- Need to add some machine-specific stuff on top (see manual for a gemini usecase)
  e.g., specify wall-clock time…

```
#!/bin/bash
#$ -S /bin/bash        # name of the used shell
#$ -V                  # copies your current (main node) settings to compute nodes
#$ -N plasim           # name of the job
#$ -pe mpich 8         # number of cores to run on (< 8 when mpich is used)
#$ -l h_rt=01:00:00    # reserved time on the nodes in hh:mm:ss
#$ -cwd                # execute job in current working directory
```

Submitting your job

Open most_plasim_run (jobscript)

- Need to add some machine-specific stuff on top (see manual for a gemini usecase)
  e.g., specify wall-clock time…
- Decide whether to restart:
  yes -> comment out `rm -f plasim_restart`
- Decide number of years (YEAR+1 = first year you run; YEARS = last year)

Script will
- Go one year further
- Check for restart file
- Run model for 1 year + write output and new restart file
- Repeat until all years are done

Submit using qsub most_plasim_run.

Postprocessing (atmosphere output)

Go to PLASIM/postprocessor

To configure the postprocessor, type:

```
c++ -O2 -o burn7.a burn7.cpp -I/opt/local/include -L/opt/local/lib -lm -lnetcdf_c++
```

Before postprocessing, specify in a namelist (here: example.nl)

- which variables to put out (code = …, …, …)
- At which pressure levels (hpa= 100,500, …)
- Whether to use monthly means (mean=1)
- Output type (netcdf=1)
- Whether to postprocess several years (e.g., multi = 30)

Then type something like:

```
./burn7.a /path_to_simulation/MOST.001 /path_to_output/output.nc < example.nl
```

Note: for ocean postprocessing, see manual.

Fun Usecases

CO2 scenarios

Adjust job script to call a python script every year

```
while [ $YEAR -lt $YEARS ]
do
   YEAR=`expr $YEAR + 1`
   DATANAME=`printf '%s.%03d' $EXP $YEAR`
   DIAGNAME=`printf '%s_DIAG.%03d' $EXP $YEAR`
   RESTNAME=`printf '%s_REST.%03d' $EXP $YEAR`
   python3 CO2_scenario.py $YEAR
   mpirun -np $NSLOTS most_plasim_t21_l10_p8.x
   [ -e Abort_Message ] && exit 1
   [ -e plasim_output ] && mv plasim_output $DATANAME
   [ -e plasim_diag ] && mv plasim_diag $DIAGNAME
   [ -e plasim_status ] && cp plasim_status plasim_restart
   [ -e plasim_status ] && mv plasim_status $RESTNAME
done
```

CO2 scenarios

Adjust job script to call a python script every year

```
while [ $YEAR -lt $YEARS ]
do
   YEAR=`expr $YEAR + 1`
   DATANAME=`printf '%s.%03d' $EXP $YEAR`
   DIAGNAME=`printf '%s_DIAG.%03d' $EXP $YEAR`
   RESTNAME=`printf '%s_REST.%03d' $EXP $YEAR`
   python3 CO2_scenario.py $YEAR
   mpirun -np $NSLOTS most_plasim_t21_l10_p8.x
   [ -e Abort_Message ] && exit 1
   [ -e plasim_output ] && mv plasim_output $DATANAME
   [ -e plasim_diag ] && mv plasim_diag $DIAGNAME
   [ -e plasim_status ] && cp plasim_status plasim_restart
   [ -e plasim_status ] && mv plasim_status $RESTNAME
done
```

The python script changes the CO2 value in the namelist (radmod_namelist)

Solar Geoengineering

No aerosol dynamics -> dim the sun!

Simple case: just change solar radiation uniformly (planet_namelist) … like for CO2.
Latitude-dependent case: Change incoming sunlight in radiation model!

```
js = 1
je = NLON
do jlat=1,NLPP !a < b .AND. a < c
   if (deglat(jlat) > 30. AND. deglat(jlat) < 60) then
      shade(js:je)=0.1
   else
      shade(js:je)=0.0
   endif
   unshade(js:je)= 1 - shade(js:je)
   js = js + NLON
   je = je + NLON
end do
```

Solar Geoengineering

No aerosol dynamics -> dim the sun!

Simple case: just change solar radiation uniformly (planet_namelist) … like for CO2.
Latitude-dependent case: Change incoming sunlight in radiation model!

```
js = 1
je = NLON
do jlat=1,NLPP !a < b .AND. a < c
   if (deglat(jlat) > 30. AND. deglat(jlat) < 60) then
      shade(js:je)=0.1
   else
      shade(js:je)=0.0
   endif
   unshade(js:je)= 1 - shade(js:je)
   js = js + NLON
   je = je + NLON
end do
```

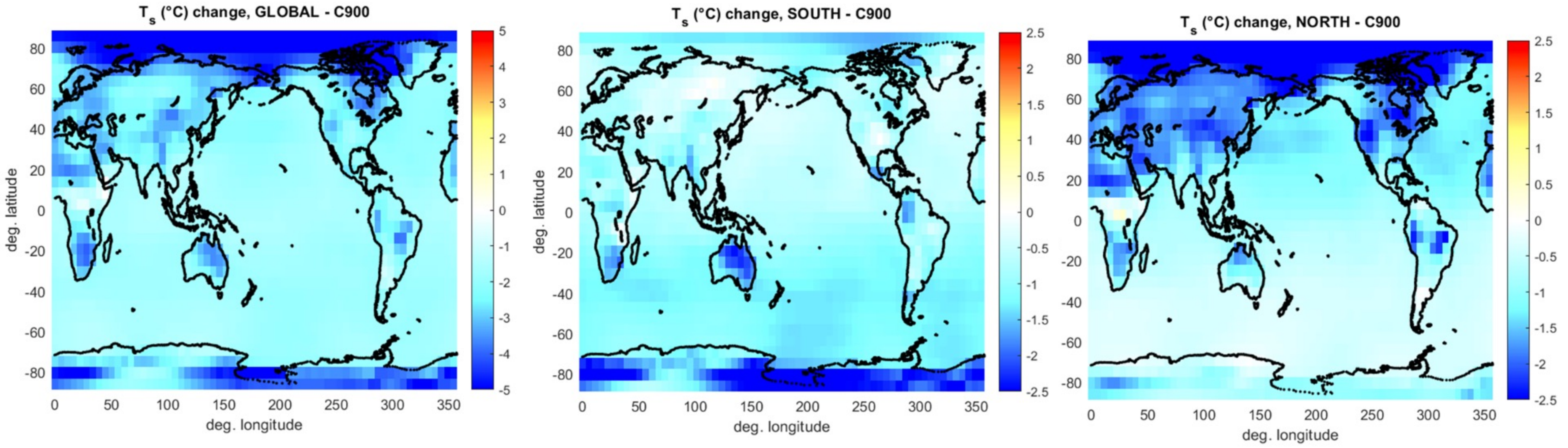
Then recompile…

Solar Geoengineering

It works!

Temperature changes under

- Global uniform SG
- Southern or Northern hemisphere only



BACKUP SLIDES

The following are some pieces of additional information on PLASIM components, especially radiation and cloud model. I’ll leave them in just in case someone is interested, but you don’t need this stuff to run the model.

Atmosphere and Ocean Dynamics (dy-core)

Dynamical Cores (dy-cores)…

-- oldest components of climate (better: weather) models

-- numerical solution of (simplified) Navier-Stokes equations

density -> pressure -> current/wind -> convergence/divergence
-> tracer advection (-> density)

PLASIM:
-- atmospheric dy-core with rough resolution (500 – 250 km)
-- ocean (LSG): dy-core with also ≈500km resolution.

A few more words on dy-cores will be said in later lecture when we talk about grids.

Radiation in the Atmosphere

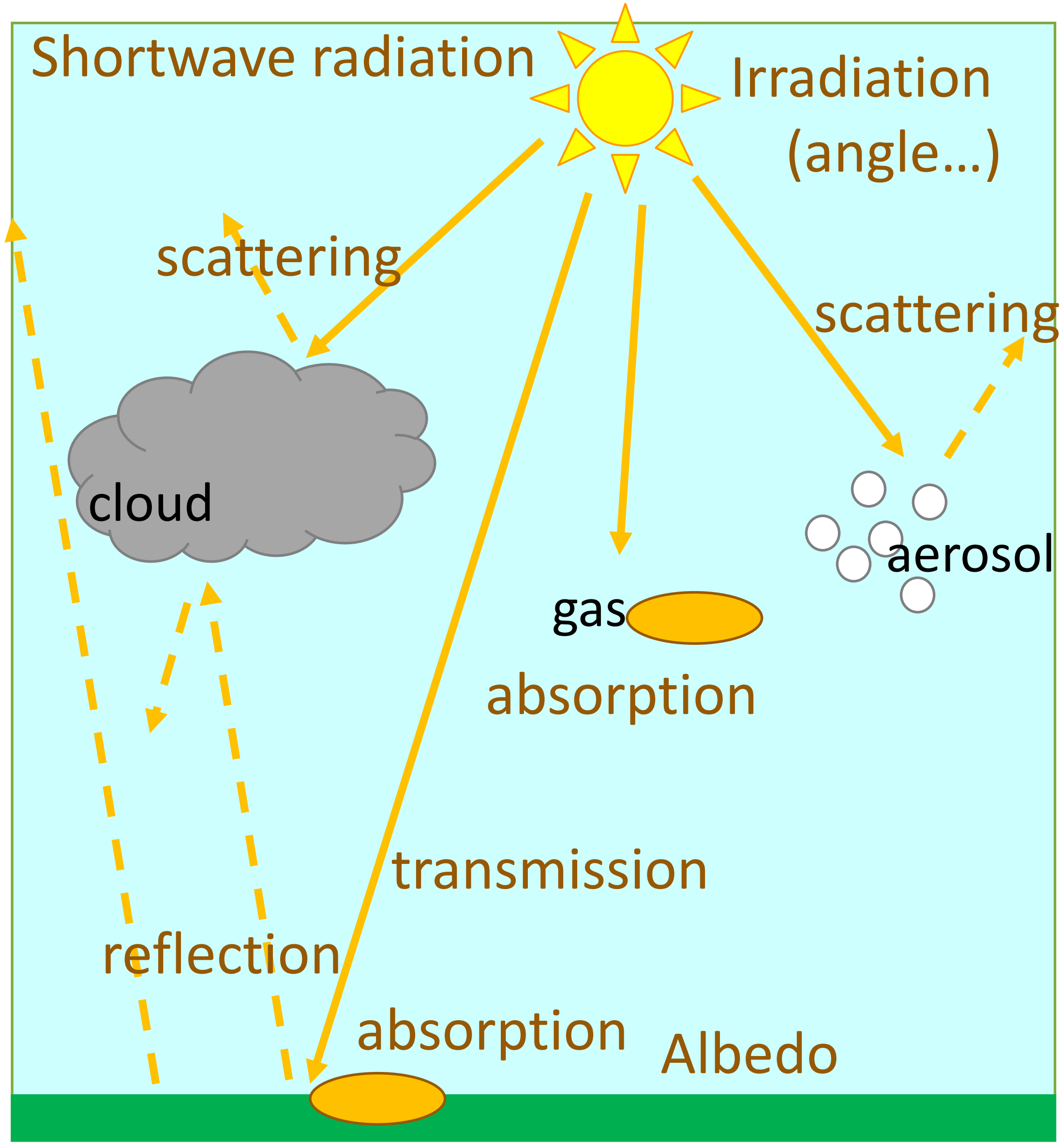
Shortwave radiation (UV, visible, near-infrared)
- comes from sun
- partly reflected back into space (-> Albedo)
- partly absorbed and converted to other forms of energy

Longwave radiation (IR)
- emitted by Earth because of its nonzero temperature

Radiation is (almost) the only way by which Earth exchanges energy with space.
In a stable climate, incoming and outgoing radiation are balanced (time-mean).

For maths details on implementation in PLASIM, read the PLASIM documentation.
Also nice: McGuffie and Henderson-Sellers, Climate Modelling Primer, edition 4, chapter 4.3

Radiation (atmosphere): shortwave



Shortwave radiation processes

-- transmission (to ground / space)

-- scattering /reflection at

-- absorption by

- clouds
- aerosol [not PLASIM]
- gas
- ground

These processes happen

-- at different heights (-> layers)

-- at different wavelengths

Radiation: shortwave [PLASIM]

$$F^{\downarrow SW} = \mu_0 \, E_0 \cdot \mathcal{T}_R \cdot \underbrace{\mathcal{T}_O \cdot \mathcal{T}_W}_{} \cdot \underbrace{\mathcal{T}_D \cdot \mathcal{T}_C}_{} \cdot \mathcal{R}_S$$

Downward SW radiation flux intensity (W/m^2) at ground

Cosine of Incidence angle

Solar intensity

Rayleigh scattering at gasses

Absorption
- Ozone (O)
- Water vapour (W)

scattering & absorption
- Dust (D)
- Clouds (C)

Surface reflection

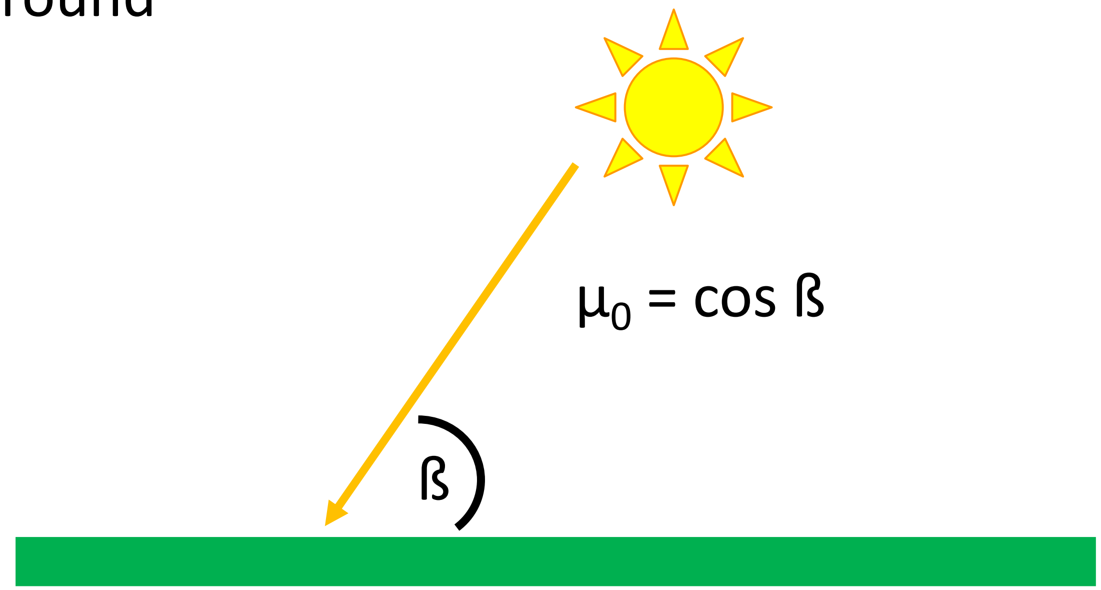
Radiation: shortwave [PLASIM]

$$F^{\downarrow SW} = \mu_0 \, E_0 \cdot \mathcal{T}_R \cdot \mathcal{T}_O \cdot \mathcal{T}_W \cdot \mathcal{T}_D \cdot \mathcal{T}_C \cdot \mathcal{R}_S$$

Downward SW radiation flux intensity (W/m^2) at ground

Cosine of Incidence angle

Solar intensity



Depend on
-- Earth's orbital parameters
-- season
-- latitude

Standard setting: daily average
Alternative: switch on diurnal cycle.

Radiation: shortwave [PLASIM]

$$F^{\downarrow SW} = \mu_0 \, E_0 \cdot \mathcal{T}_R \cdot \underbrace{\mathcal{T}_O \cdot \mathcal{T}_W}_{} \cdot \underbrace{\mathcal{T}_D \cdot \mathcal{T}_C}_{} \cdot \mathcal{R}_S$$

Downward SW radiation flux intensity (W/m^2) at ground

Cosine of Incidence angle

Solar intensity

Rayleigh scattering at gasses

Absorption
- Ozone (O)
- Water vapour (W)

scattering & absorption
- Dust (D)
- Clouds (C)

Shortwave Radiation: attenuation general

General ideas attenuation (loss by scattering or absorption)

For any wavelength λ and vertical layer we get terms like:

$$\mathbf{F_{bottom}(\lambda) = F_{top}(\lambda) * \exp(-k_\lambda u)}$$

-- F is the amount of transmitted downward radiation for a certain wavelength
-- u relates to the amount of attenuating (absorbing /scattering) material crossed
(depends also on incidence angle)
-- k is the “cross section” (strength of attenuation).

Shortwave Radiation: attenuation general

For any wavelength λ and vertical layer we get terms like:

$F_{bottom}(\lambda) = F_{top}(\lambda) * \exp(- k_{\lambda} u)$

Why are these processes very challenging to model?

Shortwave Radiation: attenuation general

For any wavelength λ and vertical layer we get terms like:

$F_{bottom}(\lambda) = F_{top}(\lambda) * \exp(- k_{\lambda} u)$

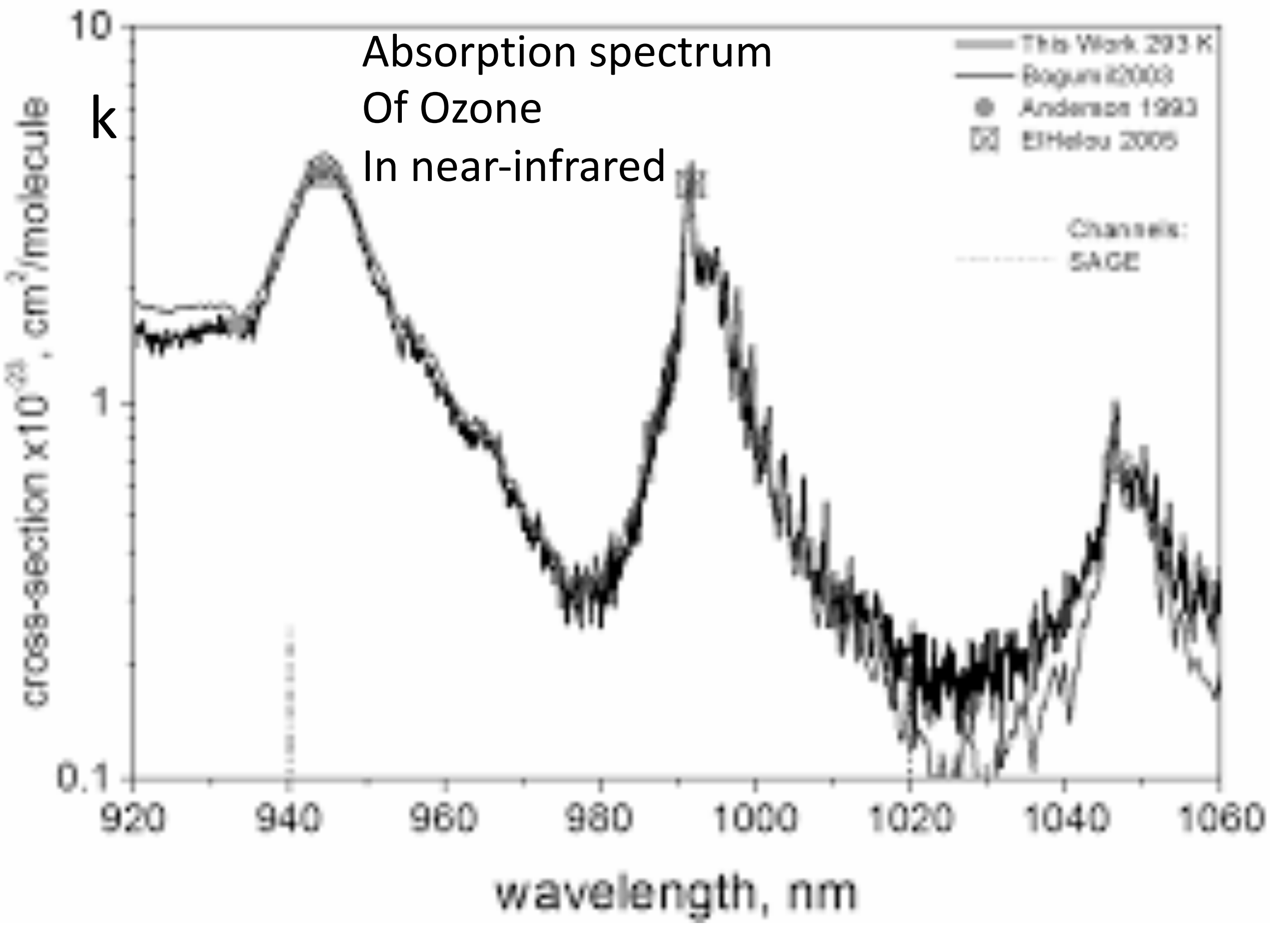
Why are these processes very challenging to model?

Because they depend on wavelength in a very unsmooth way…

Shortwave Radiation: attenuation general

Scattering and absorption depend on wavelength in a very unsmooth way…

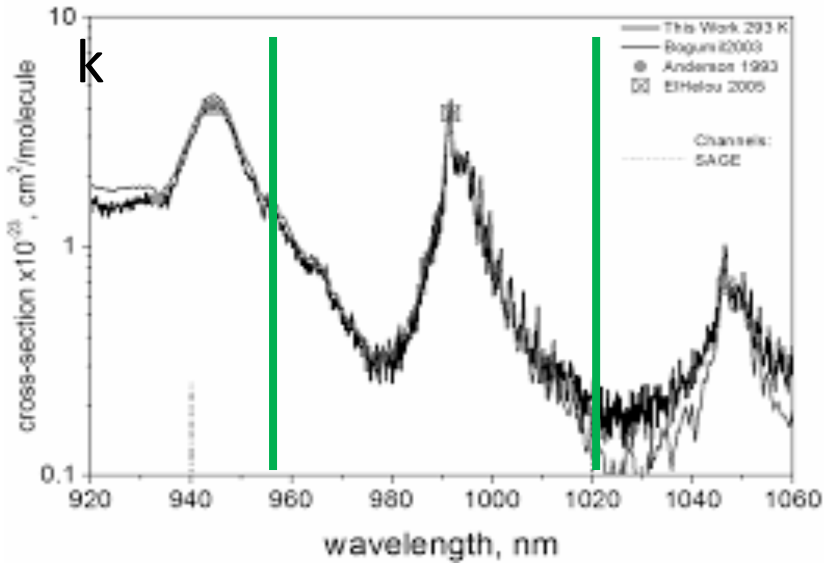
Hard to treat all wavelength separately!

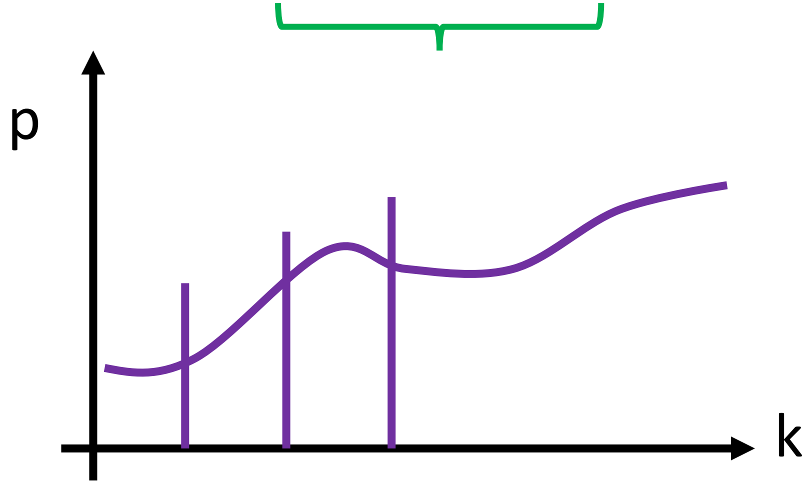


How can we simplify this?

Shortwave Radiation: attenuation general

Scattering and absorption depend on wavelength in a very unsmooth way…





Hard to treat all wavelength separately!

How can we simplify this?

-- split spectrum into wavelength bands

-- for each band j make ”k-distribution”: probability distribution of k-values

-- $\mathbf{F_{bottom,j} = F_{top,j} * \Sigma_i\, a_i \; exp\,(-\, k_i\, u)}$

a_i being a weight factor linked to the probability of k-values

Radiation: shortwave [PLASIM]

$$F^{\downarrow SW} = \mu_0 \, E_0 \cdot \mathcal{T}_R \cdot \underbrace{\mathcal{T}_O \cdot \mathcal{T}_W}_{\text{Absorption}} \cdot \underbrace{\mathcal{T}_D \cdot \mathcal{T}_C}_{\text{scattering \& absorption}} \cdot \mathcal{R}_S$$

Downward SW radiation flux intensity (W/m^2) at ground

μ_0: Cosine of Incidence angle

E_0: Solar intensity

$\mathcal{T}_R$: Rayleigh scattering at gasses

Absorption
- Ozone (O)
- Water vapour (W)

scattering & absorption
- ~~Dust (D)~~
- Clouds (C)

Neglected in Plasim

Radiation: shortwave [PLASIM]

$$F^{\downarrow SW} = \mu_0 \, E_0 \cdot \mathcal{T}_R \cdot \mathcal{T}_O \cdot \mathcal{T}_W \cdot \mathcal{T}_D \cdot \mathcal{T}_C \cdot \mathcal{R}_S$$

PLASIM splits shortwave in just 2 bands:
UV + visible: Rayleigh scattering, ozone absorption, cloud scattering
near-infrared: Water vapour absorption, cloud scattering + absorption

Radiation: shortwave [PLASIM]

$$F^{\downarrow SW} = \mu_0 \, E_0 \cdot \mathcal{T}_R \cdot \mathcal{T}_O \cdot \mathcal{T}_W \cdot \mathcal{T}_D \cdot \mathcal{T}_C \cdot \mathcal{R}_S$$

PLASIM splits shortwave in just 2 bands:
UV + visible: Rayleigh scattering, ozone absorption, cloud scattering
near-infrared: Water vapour absorption, cloud scattering + absorption

In each vertical layer, attenuation depends on
-- amount of attenuating matter passed [sometimes 0]
-- incidence angle
using simplified bulk formula.

Radiation: shortwave [PLASIM]

$$F^{\downarrow SW} = \mu_0 \, E_0 \cdot \mathcal{T}_R \cdot \mathcal{T}_O \cdot \mathcal{T}_W \cdot \mathcal{T}_D \cdot \mathcal{T}_C \cdot \mathcal{R}_S$$

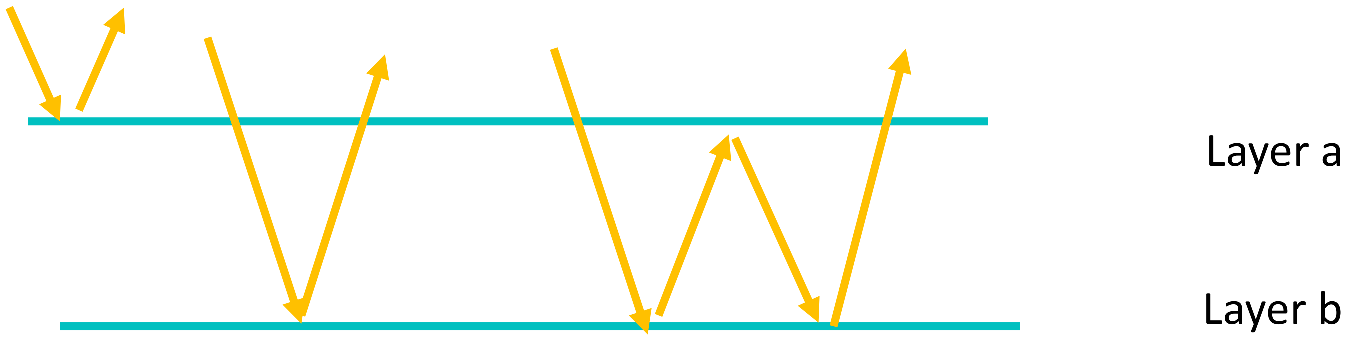
PLASIM splits shortwave in just 2 bands:
UV + visible: Rayleigh scattering, ozone absorption, cloud scattering
near-infrared: Water vapour absorption, cloud scattering + absorption

In each vertical layer, attenuation depends on
-- amount of attenuating matter passed [sometimes 0]
-- incidence angle
using simplified bulk formula.

Formulae are given both for the
downwards (direct) light beam
and the upwards (reflected, diffuse) light.

Radiation: shortwave [PLASIM]

The layers' effects are now integrated vertically,
where multiple reflections among layers are allowed.



Radiation: shortwave [PLASIM]

The layers' effects are now integrated vertically,
where multiple reflections among layers are allowed.

…. Gradually add all layers to find total combined effect.

Finally, check for each layer how much radiation passes *net*
through upper and lower boundary.

Any possible imbalances are converted into heat, warming the layer.

Atmosphere-surface interaction

Surface interaction [Plasim]

Surface (land, ocean, …) influences atmosphere by ….

-- mechanical drag

$$F_u = \rho\, C_m\, |\vec{v}|\, u \qquad\qquad F_v = \rho\, C_m\, |\vec{v}|\, v$$

Increases with:

- air density
- drag coefficient Cm (higher for rougher surfaces)
- square of wind speed in lowest level

Surface interaction [Plasim]

Surface (land, ocean, …) influences atmosphere by ….

-- sensible heat flux $$F_T = c_p \, \rho \, C_h \, |\vec{v}| \, (\gamma T - T_S)$$

Increases with:

- air heat capacity (for const. pressure)
- air density
- heat conductivity at surface Ch
- wind speed (more wind -> more vertical air mixing -> more heat transported away
- temperature difference between surface (Ts) and potential temp. lowest level, γT.

Surface interaction [Plasim]

Surface (land, ocean, …) influences atmosphere by ….

-- latent heat flux
$$L\,F_q \;=\; L\,\rho\,C_h\,C_w\,|\vec{v}|\,(\delta q - q_S)$$
(latent heat H2O * moisture flux)

Increases with:
- Latent heat
- air density
- heat conductivity at surface Ch
- wetness factor Cw (depends on local soil moisture)
- wind speed (more wind -> more vertical air mixing -> more heat transported away
- difference between actual moisture content and saturation moisture content

Surface interaction [Plasim]

To evaluate these fluxes, we need

Quickly changing surface quantities:
-- surface temperature
-- surface wetness

Slowly changing surface quantities:
-- e.g. drag coefficient
can be prescribed, but could also change e.g. with changing vegetation type

Surface/Land [Plasim]: Soil temperature

Surface temperature:

-- 5-layer model of upper 6.4m of the soil
-- sensible heat flux among soil layers based on temperature difference
-- in addition, surface heat flux (sensible, latent, and radiation)
  and heat loss from snow melt, if appropriate

No permafrost dynamics…

Surface/Land [Plasim]: Soil moisture

Soil moisture

-- 1-layer “bucket model”
-- water added by precipitation
-- water vanishes by evapotranspiration (-> moisture flux)
-- water reservoir determines “wetness factor”
-- water reservoir full -> runoff to neighbouring cells, if these lie lower
(eventually, runoff goes into sea)

Surface/Land [Plasim]: Land model

-- Forest cover determines roughness and surface albedo (even if snow)

-- Forest cover is present if sufficient biomass is present

-- Biomass loss is proportional to biomass (exponential decay)
-- Biomass production is the minimum of
    - light-limited
    - water-limited
(CO2 fertilisation plus inhibition from cold play a role, too.)

Ocean in Plasim

-- Sea Surface Temperature (and air temperature) influence sensible and latent heat flux

-- SST in turn is influenced by
- ocean surface heat flux (latent, sensible, radiation)
- oceanic heat transport

There are 3 ways to run PLASIM:
-- fixed SST (and sea ice) from observations
-- with a mixed-layer (slab) ocean
-- with a simple ocean model “Large Scale Geostrophic” or LSG model

Ocean in Plasim

The Mixed-layer model:

-- The whole ocean is just one well-mixed shallow layer
-- Currents not explicitly computed

Horizontal can be heat transport mimicked in 2 ways:
-- horizontal diffusion (which we will do)
-- "flux correction" from below, to "restore" mixed-layer temperature to correct values

Sea Ice in Plasim

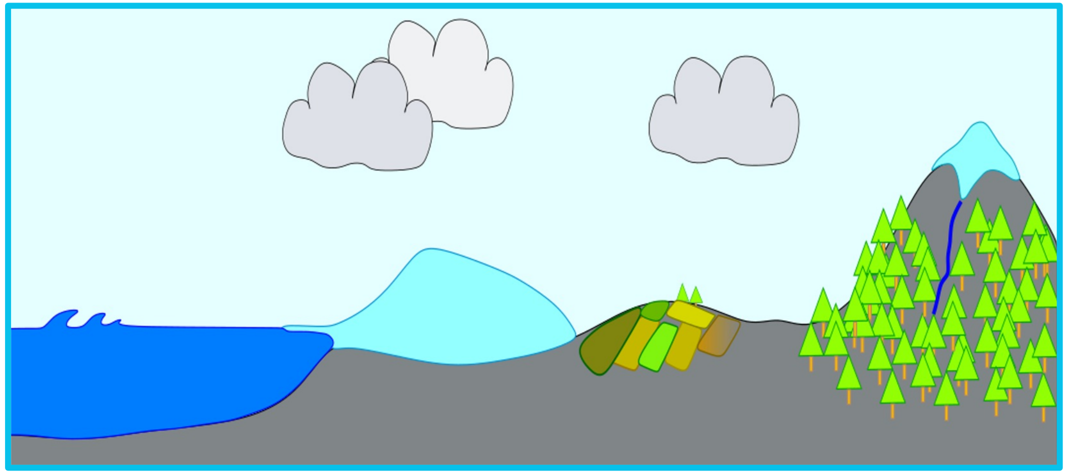
-- Only local formation and loss, no transport

-- Model first computes ice depths h
- Ice freezes from below, if ocean surface temperature below freezing (T_f = -1.9ºC)
- In addition, snow cover can be converted to ice if sufficiently massive (
- Ice melts from above if ice surface temperature exceeds T_f

-- a grid cell is considered ice-covered if h>10cm, else not.

Details: see model description

SUMMARY



-- ESMs have multiple interacting components

-- the choice of active components depends on the modellers' question (and computational constraints)

PLASIM is a small ESM

-- complete (but simple) atmosphere: dy-core, radiation, clouds & rain
-- can be run with a complete (but simple) ocean model LSG;
or minimalistic ocean (mixed layer / prescribed SST)
-- simple sea ice model
-- can be run with (simple) dynamic vegetation