The Radiative Transfer Model for the Leman Lake

FreeFEM++ for Climatology

Olivier Pironneau & Frédéric Hecht[1]

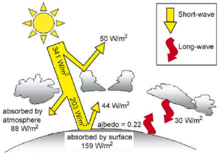
[1]LJLL, Sorbonne Université

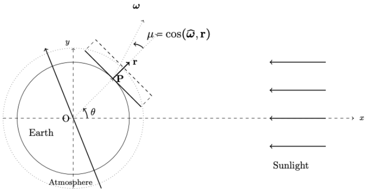
FreeFEM days, Paris, December 2021

A Climate Model Written with FreeFEM++ ?

Recall that a climate model is

- A Navier-Stokes (or multi-layers shallow water) module for the Oceans
- A varying density Navier-Stokes module for the atmosphere
- An ice melting model for the artic and Antarctic
- A Radiative Transfer module for the effect of sunlight





- $\mathbf{I}_\nu(\mathbf{x},\mu)$ light intensity of frequency ν in direction $\boldsymbol{\omega}$ at point $\mathbf{x}$. $\mu = \cos(\widehat{\boldsymbol{\omega},\mathbf{r}})$.
- $\mathbf{T}(\mathbf{x})$ the temperature.
- κ_ν is the absorption coefficient, else use the Stefan-Boltzmann formula
- a_ν is the scattering albedo.

$$\int_0^\infty \frac{\nu^3}{e^{\frac{\nu}{T}}-1} = \frac{\pi^4}{15}T^4$$

Fundamental equations

For all $\tau \in (0, Z)$, $\mu \in (-1, 1)$, $\nu \in (0, +\infty)$

$$\mu\partial_\tau I + \kappa_\nu I - \frac{\kappa_\nu a_\nu}{2}\int_{-1}^{1} I(\tau,\mu')\mathrm{d}\mu' = \kappa_\nu(1-a_\nu)B_\nu(T(\tau)),$$

$$B_\nu(T) = \frac{\nu^3}{\mathrm{e}^{\frac{\nu}{T}} - 1},$$

$$\partial_t T + \mathbf{u}\nabla T - \kappa_T\Delta T + \int_0^\infty \kappa_\nu(1-a_\nu)B_\nu(T(\tau))\mathrm{d}\nu = \int_0^\infty \frac{\kappa_\nu}{2}\int_{-1}^{1} I\mathrm{d}\mu\mathrm{d}\nu,$$

$$T \textit{ or } \frac{\partial T}{\partial n} \text{ given on } \partial\Omega, \qquad I(0,\mu)|_{\mu>0} = Q_\nu\mu, \quad I(Z,\mu)|_{\mu<0} = 0,$$

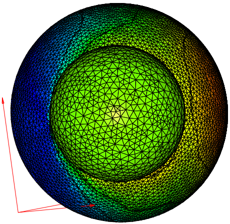
Direct finite element discretization with upwinding does not work because of singularities at $\mu = 0$.

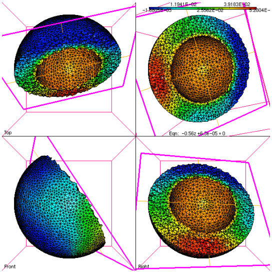
Example for Earth atmosphere

Proposition *If κ is constant (then the temperature of a planet receiving light from infinity is governed by*

$$\partial_t T \quad + \mathbf{u}\nabla T - \kappa_T \Delta T + \kappa\sigma T^4$$
$$= \frac{1}{2} Q^0 \sigma T^4_{sun} E_3(\kappa z) + \frac{1}{2}\kappa \int_0^Z E_1(\kappa|z-t|)\sigma T^4(t)\mathrm{d}t$$

where z is altitude and E_n is the exponential integral $\quad E_n(x) = \int_0^1 \mu^{n-2} \mathrm{e}^{-\frac{x}{\mu}} \mathrm{d}\mu.$





Proof: Iterative Scheme

$$\mu\partial_\tau I_\nu^{n+1} + \kappa_\nu I_\nu^{n+1} = \frac{\kappa_\nu a_\nu}{2}\int_{-1}^{1} I_\nu^n(\tau,\mu')\mathrm{d}\mu' + (1-a_\nu)B_\nu(T^n),$$

$$\partial_t T^{n+1} + \mathbf{u}\nabla T^{n+1} - \kappa_T\Delta T^{n+1} + \int_0^\infty \kappa_\nu B_\nu(T^{n+1})\mathrm{d}\nu = \int_0^\infty \kappa_\nu J_\nu^n d\nu$$

$$\text{where } J^n = \tfrac{1}{2}\int_{-1}^{1} I_\nu^{n+1}\mathrm{d}\mu\mathrm{d}\nu,$$

Monotone convergence can be shown: $\|T^{n+1} - T^*\|_0 \le c\|T^n - T^*\|_0, \quad c<1$
Discretisation in time leading to a symmetric system

$$\partial_t T + \mathbf{u}\nabla T \approx \frac{1}{\delta t}(T^{m+1} - T^m(\mathbf{x} - \mathbf{u}\delta t))$$

Tip: Minimize the energy to solve the nonlinear temperature equation

$$T^{m+1} = \arg\min\left\{\int_\Omega \left[\tfrac{1}{2}(T)^2 + \frac{\kappa_T}{2}|\nabla T|^2 + \mathcal{B}(T)\right]dx - \int_\Omega \mathcal{J}^n T dx\right\}$$

where $\mathcal{B} = \int_0^\infty \kappa_\nu \int^T B_\nu(T')dT'd\nu$.

One last step: An Integral formulation

Consider $\mu\partial_\tau I_\nu + \kappa_\nu I_\nu = F \quad I_\nu(Z,-\mu)=0, \quad I_\nu(0,\mu)=\mu Q_\nu, \quad 0<\mu<1$

There is a closed form solution, by the method of characteristics:

$$I = \mathbf{1}_{\mu>0}\left[Q_\nu \mu \mathrm{e}^{-\kappa_\nu \frac{\tau}{\mu}} + \int_0^\tau \frac{\mathrm{e}^{\kappa_\nu \frac{t-\tau}{\mu}}}{\mu} F(t)dt\right] - \mathbf{1}_{\mu<0}\int_\tau^Z \frac{\mathrm{e}^{\kappa_\nu \frac{t-\tau}{\mu}}}{\mu} F(t)dt,$$

An integration in μ gives

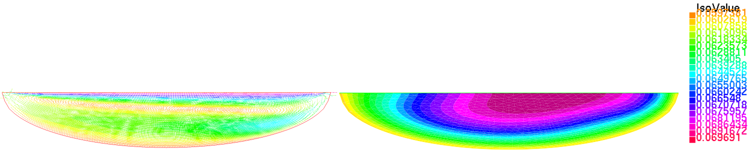
$$J_\nu(\tau) = \tfrac{1}{2}Q_\nu E_3(\kappa_\nu\tau) + \tfrac{1}{2}\kappa_\nu\int_0^Z E_1(\kappa_\nu|\tau-t|)B_\nu(T(t))\mathrm{d}t$$

So the problem is reduced to one equation only

$$\begin{aligned}\partial_t T^{n+1} \quad & +\mathbf{u}\nabla T^{n+1} - \kappa_T \Delta T^{n+1} + \int_0^\infty \kappa_\nu B_\nu(T^{n+1})\mathrm{d}\nu \\ & = \int_0^\infty \left[\tfrac{1}{2}Q_\nu E_3(\kappa_\nu\tau) + \tfrac{1}{2}\kappa_\nu\int_0^Z E_1(\kappa_\nu|\tau-t|)B_\nu(T^n(t))\mathrm{d}t\right] d\nu\end{aligned}$$

Temperature of a Lake Exposed to Sunlight

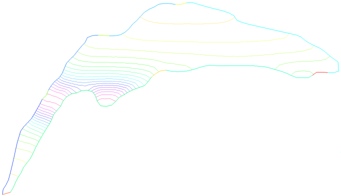
The 2D lake has an eddy due to the wind at the surface (no evaporation).
Sunlight on its surface penetrates in the water with κ_ν linear in ν.

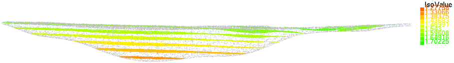


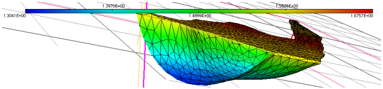
Key points in the freeFem program

```
func real expintE1(real t){
    real epst=1e-10, gamma =0.577215664901533, abst=abs(t);
    int K=9+(abst-1)*4; // precision for E1
    if(abst<epst) return 0.;
    if(abst>18) {cout << "Wrong  E1("<<t<<">18)"<<endl;return 0;}
    real ak=abst, somme=-gamma - log(abst)+abst;
    for(int k=2;k<K;k++){ ak *= -abst*(k-1)/sqr(k);somme += ak; }
    return somme;
}
```

```
func real crit(real[int] & Z){
    T[] = Z;
    return
      int2d(th)(T*T/dt/2 + kappaT*(dx(T)*dx(T) + dy(T)*dy(T))/2 + intintBkappa(T))
        - int2d(th)(convect([ux,uy],-dt,Told)*T/dt  + Jkappamean*T) ;
}
func real[int] dcrit(real[int] &Z) {
    T[] = Z;
    varf AA(T1,T1h)=int2d(th)(T*T1h/dt + kappaT*(dx(T)*dx(T1h) + dy(T)*dy(T1h)) )
        - int2d(th)(convect([ux,uy],-dt,Told)*T1h/dt + Jkappamean*T1h - T1h*intBkappa(T))
                                                        + on(a1, T1=0);
    Z= AA(0,Vh);
    return Z;
}
Told=0.06;
real[int] Z=Told[];
for(int tstep=0;tstep<15;tstep++) {
    Jkappamean = intJkappa();
    BFGS(crit,dcrit, Z, eps=1.e-6, nbiter=15, nbiterline=20);
    Told[]=Z; T=Told;
}
```







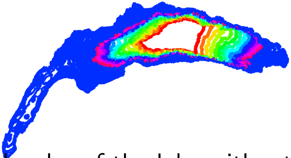
Done with the file 3d-Leman.edp" in the freeFem examples. To do better:

- GEO_LAC_LEMAN.shp Lake border and adjacent rivers.
- ISOBATHE_10_LEMAN.shp Lake bathymetric lines.
- GEO_LAC_LEMAN.shx, ISOBATHE_10_LEMAN.shx needed by shpdump.

Can we use these to be more realistic

Build a Mesh for the Leman Lake from Geographic Files

Construct a surface mesh and use the bathymetry for the volume mesh



Installation files (`gnuplot` must be installed)

- `CleanIso3d.dylib`: plugin
- texttttshapelib from `shapelib.maptools.org/`
- `border-lac-leman.edp` Construct the polygonal border of the lake without the small details `ThL-0.mesh`. Line #61 is extracted and stored in `border-lac-leman.txt`.
- `build-3d-leman.edp`. Reads the bathymetric depth lines and store them in binary format in `of.dt`
- `read-of.edp` Make sure the bathymetric lines don't overlap and generate `deep.txt`.
- `read-deep.edp` and generates `ThT.msh`.

The Different Phases

Difficulties

- The resulting triangulation has too many nodes
- The current in the lake is very small
- What is the temperature on the bottom and on the walls
- Modelling of the wind effects
- Boussinesq instabilities