III. 数値計算による偏微分方程式の解

地球惑星内部物理学演習 A の第 VIII 章では熱伝導方程式を解析的に解く方法を考えた。解析解といってもフーリエ級数あるいは積分不可能な関数で表されるため，最終的な解を求めるには数値計算が不可欠である。ここでは，得られた解析解から値を数値的に計算する方法を考える。

7. フーリエ級数の応用： プレート冷却モデル

両端に温度が与えられる有限の物体の温度はフーリエ級数で表される。そのような場合の応用として，プレートの温度を推定する問題を取り上げる。

7.1 プレートの熱的進化と熱伝導

プレートは中央海嶺で生まれ，海溝でマントルの中へ沈み込む。熱いマントル物質がマントル深部から上昇して来て湧き出す場所が中央海嶺である。地表にぶつかり，水平に運動すると，表面から熱伝導により徐々に冷やされる。これによって，冷えた部分が熱くなり，冷たい岩石は変形しにくいので剛体の板のように振る舞う。これがプレートの熱的進化である。

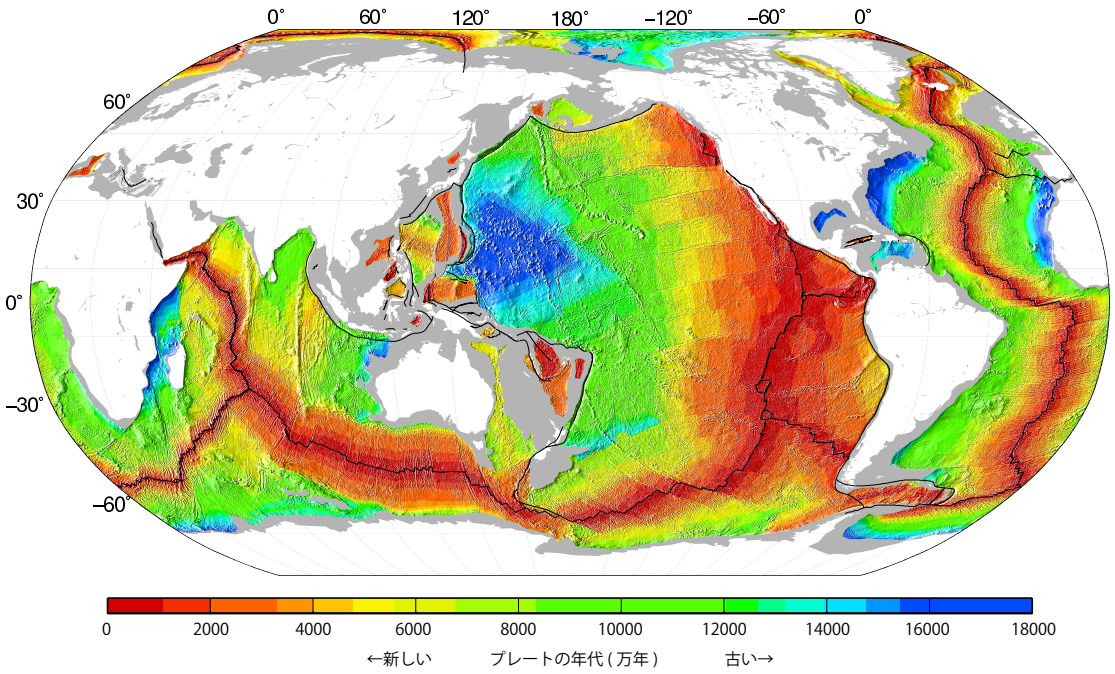

図 7.1 海洋底の年代

ここで，プレートが海嶺で生まれ，徐々に冷やされて厚くなっていく様子をシミュレートしてみよう。マントルの温度の深さ分布が時間変化するのを計算する。

プレートはマントル対流における低温の熱境界層であるから，マントル対流の一部部であると言える。したがって，プレートの温度は対流におけるエネルギー保存の式，すなわち移流・拡散方程式

$$\rho C_p\left(\frac{\partial T}{\partial t}+\boldsymbol{u}\cdot\nabla T\right)=k\nabla^2 T+\rho H \tag{7.1}$$

に従はずである。ところで，プレートに運動と伴に移動する観測者から見ると，移流を考えなくてよい。また，水平方向の温度勾配は小さいので，熱輸送に対する効果は無視して良い。そのため，プレート部分の熱輸送は1次元の熱伝導のみで表される。すなわち，温度の変化は熱伝導方程式

$$\rho C_p\frac{\partial T}{\partial t}=k\frac{\partial^2 T}{\partial z^2}+\rho H \tag{7.2}$$

に支配される。ここで，T は求める温度(°C)，z は深さ[m]，t は時間[s]である。ρ は密度[kg m^{-3}]，C_p は定圧比熱[J kg^{-1}]，k は熱伝導率(W m^{-1} K^{-1})，H は単位質量の岩石中にある放射性元素が出す熱発生率[W kg^{-1}]である。さらに，放射性元素の発熱は，プレートの年齢の1億年程度では無視できる。熱拡散率 κ (m^2 s^{-1})

$$\kappa=\frac{k}{\rho C_p} \tag{7.3}$$

を用いると，拡散方程式と同じ形，

$$\frac{\partial T}{\partial t}=\kappa\frac{\partial^2 T}{\partial z^2} \tag{7.4}$$

となる。中央海嶺に上がってきた温度 T_M を持つ熱い物質が，温度 T_0 の冷たい上部から冷やされていくという条件下[*2]で，この式を解くことになる。

7.2 フーリエ級数解を用いたプレート温度の計算

両端の温度を与えるような境界条件を持つモデルを考えるとフーリエ級数による解を応用することができる。すなわち境界条件は，

$$z=0, z=L \tag{7.5}$$

において，

$$T=T_0 \tag{7.6}$$

$$T=T_M \tag{7.7}$$

である。初期温度は，

$$T(z,0) = T_M \tag{7.8}$$

このとき，フーリエ級数で表した解は，

$$T = T_0 + (T_M - T_0)\left[\frac{z}{L} + \sum_{n=1}^{\infty}\frac{2}{n\pi}\sin\left[\frac{n\pi z}{L}\right]\exp\left[-\frac{n^2\pi^2\kappa t}{L^2}\right]\right] \tag{7.9}$$

となる。

ここで，L を小さく(~ 100 [km]程度)とると，プレートが厚さ L 以上に成長しないモデルをつくることができる。このようなモデルを**プレート冷却モデル**(plate cooling model, PCM)とよぶ。プレート冷却モデルは大洋底の水深が 80 [Ma]以上で一定となることを説明するために考えられたモデルである。L を十分に大きくとる(> 400 [km])と，海洋プレートの年代程度では，冷えてプレートが成長し続けるモデルにこの式を適用することができる。このようなモデルは**半無限体冷却モデル**(half-space cooling model, HSCM)とよばれる。なお，観測量である地殻熱流量の予測値は，

$$\frac{\partial T(z_i,t)}{\partial z} = \frac{T_M - T_0}{L}\left\{1 + 2\sum_{n=1}^{\infty}\exp\left[-\frac{n^2\pi^2\kappa t}{L^2}\right]\cos\left[\frac{n\pi z_i}{L}\right]\right\} \tag{7.10}$$

より，$z = 0$ を代入し，熱伝導率 k をかけると，

$$q = k\frac{T_M - T_0}{L}\left\{1 + 2\sum_{n=1}^{\infty}\exp\left[-\frac{n^2\pi^2\kappa t}{L^2}\right]\right\} \tag{7.11}$$

と与えられる。

(7.9)のフーリエ級数は鋸波のフーリエ級数である。これは，初期条件 $t = 0$ において，温度は

$$T(z,0) = T_M = T_0 + (T_M - T_0)\left(\frac{z}{L} - \frac{L-z}{L}\right) = T(z,\infty) + T'(z,0) \tag{7.12}$$

であり，定常解 $T(z, \infty)$を差し引いた部分が鋸波となっていることによる。また，指数関数の項は時間とともに鋸型の波形が減衰することを示している。この指数の中に n^2 が付いていることは減衰定数が長さの２乗に反比例することを表している。

(7.9)を数値的に計算するためには z を離散化する。すなわち，

$$T\left(z_i,t\right)=T_0+\left(T_M-T_0\right)\frac{z_i}{L}$$

$$+\left(T_M-T_0\right)\sum_{n=1}^{N}\frac{2}{n\pi}\exp\left[-\frac{n^2\pi^2\kappa t}{L^2}\right]\sin\left[\frac{n\pi z_i}{L}\right] \tag{7.13}$$

のように表して計算する。

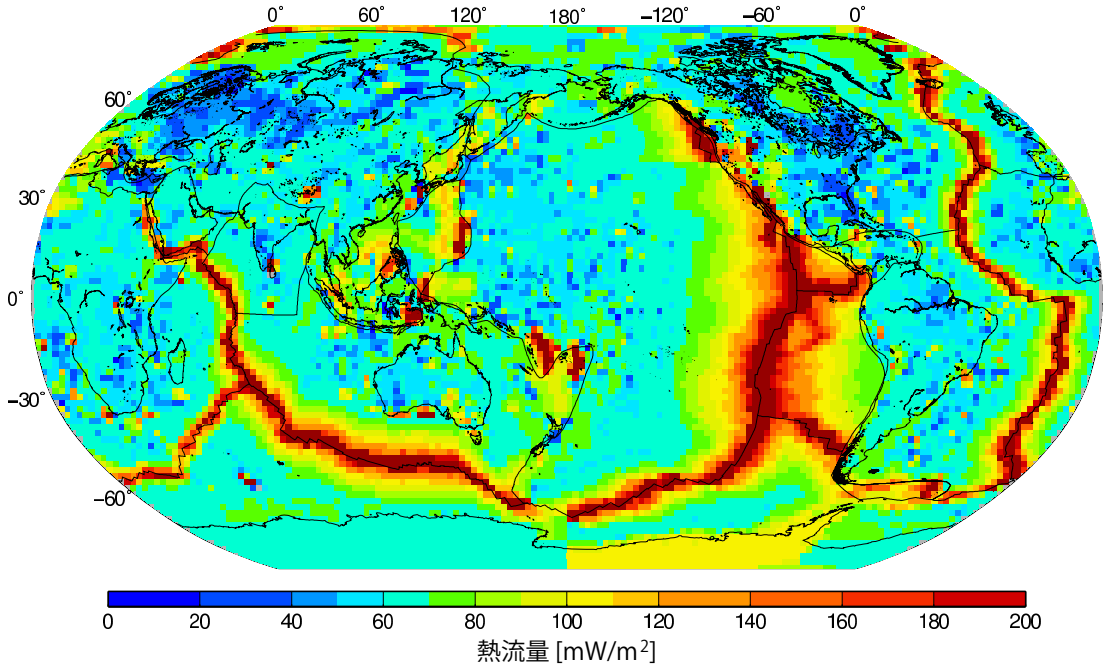

図 7.2 地殻熱流量の分布(データは Davies, 2013 による)

問題 III-1

(1) プログラムをコンパイル・実行してプレートの温度を求めよ。

(2) 地殻熱流量を(7.11)に従って計算するプログラムを作成せよ。

(3) 地殻熱流量の時間変化を(7.11)に従って計算するプログラムを作成せよ。

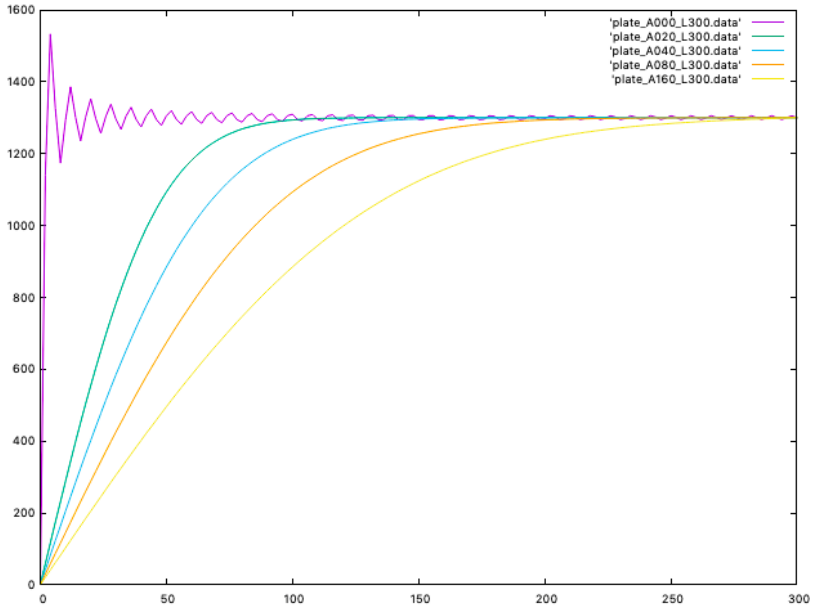

図 7.3 フーリエ級数によるプレート冷却モデルの解

8．数値積分の応用： プレートの半無限体冷却モデル

8.1 数値積分のプログラムを利用してプレートの温度を計算するプログラム

地球惑星内部物理学演習 A の資料 8 によると

$$T(z,t) = T_0 + (T_M - T_0)\,\mathrm{erf}\left[\frac{z}{2\sqrt{\kappa t}}\right] \tag{8.1}$$

と求められる。ここで，erf は誤差関数である。誤差関数は解析的に求めることができないので，Chap. 3 で説明した数値積分により求める。

時間が経つと深部まで冷却されて，温度の低い部分，すなわちプレートは厚くなる。このモデルは，マントルはプレートよりも十分厚いと考えて，無限遠に境界が取られている。このため，時間が経つとプレートは際限なく厚くなる。したがって，このモデルは**半無限体冷却モデル**(half-space cooling model)である[*3]。プレートテクトニクスが誕生するよりもずっと昔, William Thomson (Lord Kelvin)は半無限体冷却モデルを地球の冷却に適用し，地球の年代を推定した。このモデルは誤りであったが，物理的手法から地球の年代を求めようとする初めての試みであった。

(8.1)において，温度が一定となるのは

$$\zeta = \frac{z}{2\sqrt{\kappa t}} \tag{8.2}$$

である深さである。したがって，等温線は時間の平方根に比例して深くなる。すなわちプレートの厚さが時間の平方根に比例ことを意味する。を深さで微分して熱伝導率をかけると地殻熱流量が求められる。すなわち，

$$q = k\frac{T_M - T_0}{\sqrt{\pi\kappa t}} = k\frac{T_M - T_0}{\delta} \tag{8.3}$$

である。ここで，δはプレートの代表的厚さであり，

$$\delta = \sqrt{\pi\kappa t} \tag{8.4}$$

と表すことができる。

*2　解く式は時間に1階，空間に2階の微分方程式なので，解くために時間に対して1つ，空間に対して2つの条件が必要である。ここでは，時間 t=0 と空間 z=0 と∞に条件を与える。時間 t=0 の時に与える条件を初期条件と呼ぶ。また，空間の端や境界に与える条件を境界条件とよぶ。初期条件として熱いマントル

$$T(z, t=0) = T_M \tag{8.5}$$

と仮定する。境界条件は

$$T(z=0, t) = T_0 \tag{8.6}$$

$$T(z=\infty, t) = T_M \tag{8.7}$$

とする。この条件下で解を解析的に表すことが出来ない。

*3　プレートは際限なく厚くなると考えるより，一定の厚さに近づくと考える方が，年代による海底の水深の変化を説明出来るとするモデルもある。このモデルはプレート冷却モデルと呼ばれる。その原因はよく分かっていないが，プレート底面の重力不安定による剥離や熱伝導の温度依存性の効果などが考えられる。

8.2　数値積分のプログラムを利用してプレートの温度を計算するプログラム

問題 III-2 にプレートの温度を計算するプログラムを示す。このプログラムではプレートの年代を入力すると温度の深さ分布を計算するようになっている。

このプログラムは, 誤差関数計算の部分とその他の部分という２つにプログラムからなっている。誤差関数の計算する部分は, その他の部分から呼び出される。このような, 別のプログラムから呼び出されるプログラムをサブルーチンプログラム(あるいは単にサブルーチン)と呼ぶ。プログラムの一番目に動く部分をメインプログラムあるいはメインルーチンと呼ぶ。２つのプログラムは１つのファイルにしても良いが, サブルーチンごとに別のファイルにすることも出来る。ここでは, ２つのファイル(plate_main.f および plate_erf.f)とした。サブルーチンを使う時には変数値の受け渡し方や配列の取り方に決まりがある。特に, 変数の受け渡しの所は間違えると実行時エラーになり, エラーの原因が分かりにくいので, 注意が必要である。

コンパイルするときには, ファイルの名前を羅列すればよい。つまり

f95 *plate_main.f plate_erf.f*

と入力する。１つの実行形式ファイル a.out が作られる。

問題 III-2

１．メインとサブルーチンの２つのプログラムを入力して, コンパイルせよ。

２．プログラムを実行し, １千万年, ２千万年, ４千万年, ８千万年, １億６千万年のプレートの温度分布を計算せよ。結果をグラフに表し, それぞれの時間においてプレートの厚さ(1200°C 程度の温度になる深さ)を求めよ。プレートの厚さは時間に対し, どのような規則で厚くなるのか？(必要なら計算する年代を増やして) その規則性を発見せよ。

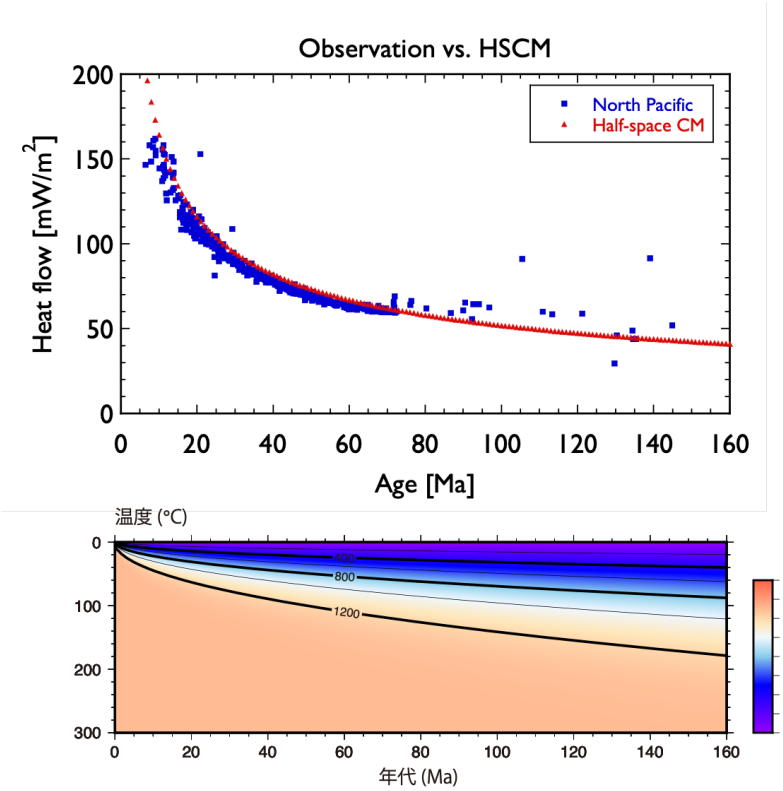

図 8.1 地殻熱流量(上:観測値と計算値)と温度(下: 計算値)。

熱流量のデータは Davies (2013)のグリッドデータから RMS の小さい場所の値のみプロット。

問題 III-1　プログラム

プログラムのほか，パラメータを入力するためのデータファイルが必要である．それぞれのファイルを同じディレクトリに置いて，コンパイル，実行すること．

メインプログラム plate_main2_w.f90

```
! ************************************************************************
! *                                                                      *
! *     1-D Half-space cooling model for plate thermal evolution         *
! *                                                                      *
! ************************************************************************

    program plcm

      implicit none

      real(8):: rho,  Cp,   ak,   D
      real(8):: age,  T0,   TM
      real(8):: zk,   z,    zmax, dz,   dzk,  t
      real(8):: hpl,  si_ex
      integer:: mz,   j,    ndeg
      character(40):: apara
      character(30):: fname

      real(8),allocatable:: Tmp(:)

!  *****  read parameters  *****

!  Open file

      open ( 10, file = 'pl_para2_w.dat' )
```

```
!  read file

      read (10,*) apara
      read (10,*) rho
      write (6,*) apara,' = ',rho

      read (10,*) apara
      read (10,*) Cp
      write (6,*) apara,' = ',Cp

      read (10,*) apara
      read (10,*) D
      write (6,*) apara,' = ',D

      read (10,*) apara
      read (10,*) age
      write (6,*) apara,' = ',age

      read (10,*) apara
      read (10,*) T0
      write (6,*) apara,' = ',T0

      read (10,*) apara
      read (10,*) TM
      write (6,*) apara,' = ',TM

      read (10,*) apara
      read (10,*) mz
      write (6,*) apara,' = ',mz

      read (10,*) apara
      read (10,*) zmax
      write (6,*) apara,' = ',zmax
```

```
      read (10,*) apara
      read (10,*) fname
      write (6,*) apara,' = ',fname

!  convert unit km -> m, Ma -> sec

      t    = age  * 1.0d6 * 365.2422d0 * 24.0d0 * 3600.0d0
      hpl  = zmax * 1.0d3

!  depth interval

      dz   = hpl / dble( mz )
      dzk  = dz  / 1.0d3

!  thermal conductivity

      ak = rho * Cp * D

!  allocate dimension variables

      allocate ( Tmp( 0:mz ) )

!  maximum degree of Fourier series

        ndeg = mz / 2
!
!  ******  calculate temperature  ******
!

!  at the surface

      Tmp(0)  = T0
```

```
      Tmp(mz) = TM

!  at z = dz to zmax-dz ( j=1,mz-1 )

      do j = 1,mz-1
        z = dz * dfloat( j )

!  Fouries series of saw-tooth function and decay with time

        call     fourexp( si_ex, z, t,  D, hpl, ndeg )

!  composite the steady-state and varying temperatures

        Tmp(j) = T0 + ( TM-T0 ) * ( z/hpl + si_ex )

!        write (6,*) z(j),Tmp(j),si_ex

      end do
!
!   *****  Output results to a file
!

!  Open file

      open ( 20, file = fname )

!  write file

      do j = 0,mz
        zk = dzk * dfloat( j )
        write (20,*) zk, Tmp( j )
      end do
```

```
!  close files

      close (10)
      close (20)

      stop

    end program plcm
```

サブルーチンプログラム plate_four_w.f90

```
!
!  *******  Fourier series of saw-tooth function for PLCM  *******
!
   subroutine fourexp( si_ex, z, t,  D, hpl, ndeg )

      implicit none

      real(8):: si_ex
      real(8):: z,   t
      real(8):: D,   hpl
      real(8):: s1,  sall
      real(8):: pi
      integer:: ndeg
      integer:: i,   n
      real(8),allocatable:: f(:)

!  ****  allocate  ****

      allocate ( f( ndeg ) )

!  ****  pi  ****
```

```
      pi = 4.0d0 * atan( 1.0d0 )

!  ****  calculate sin( n*pi*z/H )*exp( -(n*pi/H)**2*D*t )  ****

      do i = 1,ndeg
        f(i) = 2.0d0 / ( dfloat(i) * pi )                          &
        &      * sin( dfloat(i) * pi *z / hpl )                    &
        &      * exp( - ( dfloat( i ) * pi / hpl )**2 * D * t )
      end do

!  ****  summation  ****

      s1 = 0.0d0
      do i = 1,ndeg
        s1 = s1 + f(i)
      end do

      si_ex = s1

      return

    end subroutine fourexp
```

パラメータデータファイル pl_para2_w.dat

```
'density of the mantle (kg/m**3)'
3300.0d0
'specific heat (J/kg)'
1200.0d0
'thermal diffusivity (m^2/s)'
1.0d-6
'age of plate (Ma)'
040.0d0
```

```
'surface temperature (deg. C)'
0.0d0
'mantle temperature (deg. C)'
1300.0d0
'number of depth interval'
150
'plate thickess (km)'
300.0d0
'output file name'
'plate_A040_L300.data'
```

1 次元熱伝導方程式の数値解を求めるプログラムである．入力されたインターバルで深さを変えながら温度を計算する．プログラムは 2 つの部分に分けられており，フーリエ級数を含む級数を計算する部分をサブルーチン，その他をメインルーチンにしている．それぞれを 1 つのファイルにしている．

主プログラム(メインルーチン)

`program plcm`： program 文．プログラム名を宣言する．

`implicit none`：宣言文．変数をすべて明示的に宣言することを宣言している．

`real(8):: rho,…`：宣言文．`rho` 以下が 8 バイト(倍精度)実数変数であることを宣言．

`integer:: mz,…`：宣言文．`mz` 以下が 4 バイト整数であることを宣言．

`character(40):: apara`：宣言文．`apara` が 40 バイト文字変数(ASCII 文字 40 字)であることを宣言．

`real(8),allocatable:: Tmp(:)`：宣言文．`Tmp` が倍精度実数であるとともに，動的割り付け配列変数であることを宣言．

`open ( 10, file = 'pl_para2_w.dat' )`：補助出入力(open)文．物理パラメータが書かれたファイル `pl_para2_w.dat` を開いて装置番号 10 番に関連づける．

`read (10,*) apara`：入力文．ファイルの 1 行目を文字列として `apara` に読み込む．文字列はクォーテーションマークで囲わねばならない．この read 文が終わると，改行して次の行へ移動する．

`read (10,*) rho`：入力文．ファイルの 2 行目の数値を倍精度実数変数 `rho` に読み込む．

`write (6,*) apara,' = ',rho`：出力文．`apara` の値と“=”と `rho` の値を装置番号 6 番の装置(標準出力装置：画面)に 1 行に出力する．

t = age * 1.0d6 * 365.2422d0 * 24.0d0 * 3600.0d0：代入文．Ma (百万年)を秒に直す

dz = hpl / mz：代入文．計算点の間隔数 mz から深さの間隔 dz を計算する．計算点のインデックス j は 0 から始まるので，計算点は全部 mz+1 では個ある．

allocate (Tmp(0:mz))：割り付け文．Tmp の配列の範囲が 0 から mz であるとして，メモリの領域を Tmp 用に (mz+1)×8 バイト分割り付ける．

ndeg = mz / 2：代入文．離散フーリエ級数の次数の最大値を mz/2 に設定する．

Tmp(0) = T0：代入文．表面の点の温度を代入している．

Tmp(mz) = TM：代入文．マントルあるいはプレート底の点の温度を代入している．

do j = 1,mz-1 〜 end do：do ループ．計算点のインデックスには j を使用しているが，ループのインデックスとしてもこれを使用している．深さを変化させながら温度を計算する．

z = dz * dfloat(j)：代入文．インデックス j に対応した深さを計算している．dz は深さの間隔である．

call fourexp(si_ex, z, t, D, hpl, ndeg) ：call 文．級数部分を計算するサブルーチンを呼び出す．( )の中はサブルーチンとの間でローカル変数の値を受け渡すための引数が入る．ここでは si_ex, z, t, D, hpl, ndeg の 6 つである．si_ex 以外は前に定義されいる．つまり，この値をサブルーチンに値を引き継ぐ．si_ex は前に定義されていない変数で，サブルーチン fourexp で計算した結果が返って来る．

Tmp(j) = T0 + (TM-T0) * (z/hpl + si_ex)：代入文．定常解と級数の結果を足し合わせて温度を計算する．

open (20, file = fname)：open 文．出力先ファイルを開く．“read (10,*) fname”で読み込んだファイル名のファイルを新規作成する，あるいは開く．

do j = 0,mz 〜 end do：do ループ．結果をファイルに書くための do ループである．

write (20,*) zk, Tmp(j)：出力文．zk と Tmp(j)の値をファイルを装置番号 20 のファイルに 1 行として書き込む．

close (10)：補助出入力(close)文．．ファイルに EOF(end of file)を書き，ファイルを閉じる．

副プログラム(サブルーチン)

subroutine fourexp(si_ex, z, t, D, hpl, ndeg) ：subroutine 文．サブルーチンの始まりを表す．( )の中は引数が入る．引数は call 文と数や型が合っていなければならない．それぞれの変数はプログラム単位ごとに定義されている．コールされたときには，引数の値は，値をメインルーチンからもらっている．リターンのときは逆に，メインルーチンへ変数の値を返している．コールされたときにはメインルーチン側で z, t, D, hpl, ndeg の値は決まっているが，si_ex は不定である．サブルーチンで計算した結果が si_ex に代入される．

real(8):: z, t：宣言文．変数はプログラム単位ごとに決まる(ローカル変数)なので，プログラム単位ごとに宣言する必要がある．

pi = 4.0d0 * atan(1.0d0)：代入文．円周率を計算する

f(i) = 2.0d0 / (dfloat(i) * pi)…&：代入文．級数項を計算する．最後の&は次の行へ継続することを表す．

& * sin(dfloat(i) *…&：継続行．先頭の&は省略できるが継続行であることをわかりやすくするために書いている．この行も最後に&があり，継の行へ継続する．

return：return 文．サブルーチンから呼び出したプログラムへ復帰する．引数の所に入っている変数の値がすべて呼び出したプログラムへ引き渡される (つまり，サブルーチンで変えてしまった値はコールした側でも変化する)．これによって，メインルーチンで si_ex が確定した値となる．

end subroutine fourexp：end subroutine 文．副プログラムの終わりを表す．

パラメータファイル

奇数行はその下の値の意味を表すコメント行，偶数行がデータとなっている．

上から，マントルの密度(kg m^{-3})，定圧比熱(J kg^{-1})，熱拡散率(m^2 s^{-1})，プレートの年代(Ma)，地表温度(°C)，マントルの温度(°C)，計算点の間隔数(個)，計算する最大の深さ(km)，出力ファイル名，である．

問題 III-2　プログラム

プログラムのほか，パラメータを入力するためのデータファイルが必要である。それぞれのファイルを同じディレクトリに置いて，コンパイル，実行すること。

メインプログラムファイル　plate_main.f90

```
! ************************************************************************
! *                                                                      *
! *     1-D Half-space cooling model for plate thermal evolution         *
! *                                                                      *
! ************************************************************************

   program hcm1_main

      implicit none
      real(8):: rho,Cp,ak,D
      real(8):: age,T0,TM
      real(8):: zk,z,zmax,dz,dzk,t
      real(8):: eta1,eta2,erfy,derf
      real(8), allocatable:: Tmp(:)
      integer:: nz,j,ntra
      character(40):: apara,outfile

!  *****  read parameters  *****

!  Open parameter file

      open ( 10, file = 'pl_para.dat' )

!  read file

      read (10,*) apara
```

```
read (10,*) outfile
write (6,*) apara,' = ',outfile

read (10,*) apara
read (10,*) rho
write (6,*) apara,' = ',rho

read (10,*) apara
read (10,*) Cp
write (6,*) apara,' = ',Cp

read (10,*) apara
read (10,*) ak
write (6,*) apara,' = ',ak

read (10,*) apara
read (10,*) age
write (6,*) apara,' = ',age

read (10,*) apara
read (10,*) T0
write (6,*) apara,' = ',T0

read (10,*) apara
read (10,*) TM
write (6,*) apara,' = ',TM

read (10,*) apara
read (10,*) dzk
write (6,*) apara,' = ',dzk

read (10,*) apara
read (10,*) zmax
```

```
      write (6,*) apara,' = ',zmax

      read (10,*) apara
      read (10,*) ntra
      write (6,*) apara,' = ',ntra

!  convert unit km -> m, Ma -> sec

      dz   = dzk  * 1.0d3
      t    = age  * 1.0d6 * 365.2422d0 * 24.0d0 * 3600.0d0

!  thermal diffusivity

      D = ak / ( rho*Cp )

!  *****  nz: number of z points  ****

      nz = int( zmax / dzk + 0.5d0 )

!  allocate dimension variables

      allocate ( Tmp(0:nz) )

!  ******  calculate error function  ******

!  at the surface

      Tmp(0) = T0
      eta1 = 0.0d0
      erfy  = 0.0d0
!  at z = dz to zmax

      do j = 1,nz
```

```
        z = dz * dfloat( j )

!  eta = z / 2*(D*t)**(1/2) ****

        eta2 = 0.5d0 * z / sqrt( D*t )

!  error function

        call     derrfunc( eta1,  eta2,  derf,  ntra )
        erfy = erfy + derf

!  Temperature

        Tmp(j) = T0 + ( TM-T0 ) * erfy

!        write (6,*) z,Tmp(j),erfy

        eta1 = eta2
      end do

!   *****  Output results to a file

!  Open a new file to save result

      open ( 20, file = outfile,status='new' )

!  write to file

      do j = 0,nz
        zk = dzk * dfloat( j )
        write (20,*) zk,Tmp(j)
      end do
```

```
!  close files

      close (10)
      close (20)

      stop
   end program hcm1_main
```

サブルーチンプログラムファイル　plate_derf.f90

```
!
!  *******  Error function subroutine  *******
!   Numerical integration by trapezoid method
!
   subroutine derrfunc( eta1,  eta2,  derf,  n )

      implicit none

      real(8), allocatable::  f(:)
      real(8)::  eta1,eta2
      real(8)::  x,x1,x2,dx
      real(8)::  s1,sall
      real(8)::  derf
      real(8)::  pi,rootpi
      integer:: i,n

!  ****  integration section  ****

      x1 = eta1
      x2 = eta2

      dx = ( x2-x1 ) / dfloat( n )
```

```
!  ****  Value of function to be integrated  ****

      allocate ( f(0:n) )

      do i = 0,n
        x = x1 + dx * dfloat(i)
        f(i) = exp( - x**2 )
      end do

!  ****  integration  ****

      s1 = 0.0d0
      do i = 1,n-1
        s1 = s1 + 2.0d0*f(i)
      end do

      sall = dx * 0.5d0  * ( f(0) + s1 + f(n) )

!  ****  error function  ****

      pi = 4.0d0 * atan( 1.0d0 )
      rootpi = sqrt( pi )
      derf = 2.0d0 / rootpi * sall

      return
   end subroutine derrfunc
```

パラメータファイル　pl_para.dat

```
'output file name'
'plate_T_100Ma.data'
'density of the mantle (kg/m**3)'
3300.0d0
```

```
'specific heat (J/kg)'
1200.0d0
'thermal conductivity (W/m)'
4.0d0
'age of plate (Ma)'
20.0d0
'surface temperature (deg. C)'
0.0d0
'mantle temperature (deg. C)'
1300.0d0
'depth interval (km)'
2.0d0
'maximum depth (km)'
300.0d0
'number of sub-section for integration'
5
```

1次元熱伝導方程式の数値解を求めるプログラムである。入力されたインターバルで深さを変えながら温度を計算する。

プログラムは2つの部分に分けられており，誤差関数の増分を計算する部分をサブルーチンにしている。それぞれを1つのファイルにしている。サブルーチンは，メインプログラムから，必要なときに呼び出される(call 文)。サブルーチンを呼び出すときには，処理に必要な変数の値をサブルーチンに引き渡す。処理が終わったら，今度は結果の変数の値をサブルーチンに引き渡してもらう。変数の受け渡しは，引数によって行う(common 文を使う方法もある)。サブルーチンは subroutine 文で始まる。メインプログラムの program 文と異なり，プログラムがサブルーチンであることと，引数を記述する。

主プログラム(メインルーチン)

`open ( 10, file = 'pl_para.dat' )`：open 文。物理パラメータが書かれたファイルを開く。

`t    = age  * 1.0d6 * 365.2422d0 * 24.0d0 * 3600.0d0`：代入文。Ma (百万年)を秒に直す

`nz = int( zmax / dzk + 0.5d0 )`：代入文。温度を計算する最大の深さと深さの間隔から計算点の数を計算している。最後に 0.5 を足しているのは，次のような理由である。数を2進数に直

すときに必ずしも割り切れず，例えば3.0(0.3×10^{1}として変数に入っている)が2.99998になってしまうことがある。これを整数に変数の型を変換すると，3になってほしいところ，2になってしまう。このため，1以下の数(ここでは0.5)を足して3.49998にしておけば3が得られるのである。

`do j = 0,nz 〜 end do`：doループ。ループのインデックスとして，`j`を使用している。深さを変化させながら温度を計算する。

`z = dz * dfloat( j )`：代入文。インデックス`j`に対応した深さを計算している。`dz`は深さの間隔である。

`eta2 = z / sqrt( D*t )` ：代入文。$\eta_2=z/\sqrt{(\varkappa t)}$ を計算している。

`call derrfunc( eta1, eta2, derf, ntra )` ：call文。誤差関数の増分$\mathrm{erf}(\eta_2)$–$\mathrm{erf}(\eta_1)$を計算するサブルーチンを呼び出す。( )の中はサブルーチンとの間でローカル変数の値を受け渡すための引数が入る。ここでは`eta1`，`eta2`，`derf`，`ntra`の4つである。`eta1`，`eta2`，`ntra`は前に定義されいる。つまり，この値をサブルーチンに値を引き継ぐ。`derf`は前に定義されていない変数で，サブルーチン`derrfunc`で計算した結果が返って来る。

`erf = erf + derf`：代入文。サブルーチンから返ってきた増分`derf`を`erf`に加算する

`Tmp(j) = T0 + ( TM-T0 ) * erf`：代入文。`erf`の値を使って温度を計算する

副プログラム(サブルーチン)

`subroutine derrfunc( eta1, eta2, derf, n )` ：subroutine文。サブルーチンの始まりを表す。( )の中は引数が入る。引数はcall文と数や型が合っていなければならない。ここでは，コールしたメインルーチンから`eta1`，`eta2`，`n`をもらって，計算結果`derf`をメインルーチンに返している。

`return`：return文。サブルーチンから呼び出したプログラムへ復帰する。引数の所に入っている変数の値が呼び出したプログラムへ引き継がれる。

`end subroutine derrfunc`：end subroutine文。副プログラムの終わりを表す。

パラメータファイル

上から，マントルの密度(kg m^{-3})，定圧比熱(J kg^{-1})，熱伝導率(W m^{-1})，プレートの年代(Ma)，地表温度(°C)，マントルの温度(°C)，計算する深さの間隔(km)，計算する最大の深さ(km)，誤差関数の増分を計算する際の積分区間の分割数，である。

Fortran サブルーチンに関するいくつかの注意

副プログラム(サブルーチン)の呼び出し

```
call subroutine_name ( var1, var2, var3,…  )
```

ここで，var1, var2, var3,…は引数である。呼び出す側は，数値またはパラメータであっても，変数や配列変数であっても良い。変数の名前が違っていたり，配列変数の大きさが違っていると，メモリ上のアドレスにずれが生じて実行時エラー(すべて segmentation fault error になる)や誤った処理の原因になる。バグを見つけるのが大変で，頭の痛いところである。対処するには，１つ１つの変数に正しい値が入っているのかチェックしていくのが地道だか近道である。関数との違いはサブルーチン名そのものが値を持たないことである。このため，call 文を使ってサブルーチンを呼び出す

副プログラム(サブルーチン)の始め

```
subroutine subroutine_name ( var1, var2, var3,…)
```

引数は呼び出される側は基本的に変数である。

副プログラム(サブルーチン)から呼び出したプログラムに復帰

```
return
```

だいたいサブルーチン最後にある(if 文の中に書かれる場合などを除き)。return で，呼び出したプログラムに戻って処理が継続する。return がないと処理が呼び出したプログラムに復帰しない。復帰するとき，すべての引数が呼び出したプログラム戻される。サブルーチン内で値を変えていると，呼び出したプログラムでも値が変わってしまうので注意が必要である。

副プログラム(サブルーチン)の最後

```
end subroutine subroutine_name
```

subroutine 文から end 文までが１つのサブルーチンのプログラム単位である。１つのファイルの中にメインプログラムや複数のサブルーチンがあっても構わない。

図の描き方

(1) gnuplot を使用する。

```
gnuplot
gnuplot> plot "t000" with line
gnuplot> replot "t002" with line
gnuplot> quit
```

グラフは画面キャプチャなどを用いて画像として保存する。

(2) Generic Mapping Tools (GMT)を使用する

```
psxy -R0/200/0/1500 –JX10/8 –X2.0 –Y2.0 -W1.0 -Ba50f10/a200f20 t001 > temp.ps
```

Postscript file の表示

```
evince temp.ps
```