Citcoms Changes in Source codes

1 Multi-tracer Code

To append anything related to tracer see Tracer_setup.c and Ggrd_handling.c.

NOTE: This changes are for isochemical mantle convection.

If used with thermochemical convection then append **Compostion_related.c** to set different buoyancy_ratio for each tracer flavor.

1.1 Changes in Tracer_setup.c

Change from static to dynamic memory allocation in ggrd initialization of trace.z_interface

Previous Verisons:

```
   #ifdef USE_GGRD
case 1:
case 99:  /* will override restart */
/* from grid in top n materials, this will override he checkpoint input */
input_string("ictracer_grd_file",E->trace.ggrd_file,"",m); /* file from which to read */
input_int("ictracer_grd_layers",(E->trace.ggrd_layers),"2",m);

   /*>0 :  which top layers to use, layer <= ictracer_grd_layers
    <0 :  only use one layer layer == -ictracer_grd_layers */

   break;
#endif
```

Newer Verisons:

```
   #ifdef USE_GGRD
case 1:
case 99:  /* will override restart */
/* from grid in top n materials, this will override he checkpoint input */
input_string("ictracer_grd_file",E->trace.ggrd_file,"",m); /* file from which to read */
input_int("ictracer_grd_layers",(E->trace.ggrd_layers),"2",m);

   /*>0 :  which top layers to use, layer <= ictracer_grd_layers
    <0 :  only use one layer layer == -ictracer_grd_layers */

   /*** Modified by Debanjan Pal to allocate dynamic memory to  trace.z_interface ***/

   E->trace.z_interface = (double*) malloc((E->trace.nflavors-1) *sizeof(double));

   for(i=0; i<E->trace.nflavors-1; i++)  {
input_double_vector("z_interface", E->trace.nflavors-1, E->trace.z_interface, m);
}

   /********************* Modified upto here ***********************************************************/
break;
#endif
```

1.2 Changes in Ggrd_handling.c

Previous Verisons:

```
   if(!E->control.ggrd_comp_smooth) {
/* limit to 0 or 1 */
if(indbl < .5)
indbl = 0.0;
if(indbl > .5  indbl < 1.5)
indbl=1.0;
E->trace.extraq[j][0][kk]= indbl; }

   else {
/* below */
E->trace.extraq[j][0][kk] = 0.0;
} } }
```

Newer Verisons:

While inserting tracer.grd file containng three tracer flavors, flavors having value less than 0.5 will be assigned 0, between 0.5 and 1.5 will be 1, and greater than 1.5 will be 2.

In this I assume the flavor for ambient mantle=0, weak boundaries=1, craton=2

*****These modifications only hold if you are introducing tracers that lie in single layer.**

```
/***Modified by Debanjan Pal to include more than 2 tracers, apprenend according to ggrd file***/

   if(!E->control.ggrd_comp_smooth) {
/* limit to 0 or 1 */
if(indbl < .5)
indbl = 0.0;
if(indbl > .5  indbl < 1.5)
indbl=1.0;
else
indbl=2.0;
E->trace.extraq[j][0][kk]= indbl; }


   /********************** Modified upto here ***************************************************/
else {
/* below */
E->trace.extraq[j][0][kk] = 0.0;
} } }
```

****Suppose you want to introduce weak plate boundaries in top 100Km and cratons upto top 300Km**

To use this version, you have to define ictracer_grd_layers the layer upto which craton is to be defined.

```
/***Modified by Debanjan Pal to include more than 2 tracers, apprenend according to ggrd file***/

   if(!E->control.ggrd_comp_smooth) {
/* limit to 0 or 1 */
if(indbl < .5)
indbl = 0.0;
if(indbl > .5  indbl < 1.5)
if ( rad > 0.98125) /*Checks for depth condition and defines weak boundaries in top 100Km*/
indbl=1.0;
else
indbl=2.0;
E->trace.extraq[j][0][kk]= indbl; }
/************************** Modified upto here ***************************************************/

    else {
/* below */
E->trace.extraq[j][0][kk] = 0.0;
} } }
```

****Suppose you want to introduce cratons in top 300 Km and thermochemical layer at the base of CMB upto 2500 Km. Still you need to give ictracer_grd_layers upto the layer in which you are defining cratons.**

Here I assume craton to have tracer flavor 1 and LLSVPs as 2 and 0 be the ambient mantle

/***Modified by Debanjan Pal to include more than 2 tracers, apprenend according to ggrd file***/

```
   if(!E->control.ggrd_comp_smooth) {
/* limit to 0 or 1 */
if(indbl < .5)
indbl = 0.0;
if(indbl > .5  indbl < 1.5)
if ( rad > 0.95290) /*Checks for depth condition and defines weak cratons in top 300Km*/
indbl=1.0;
else
indbl=0.0;

   else
if ( rad < 0.6) /*Checks for depth condition and defines LLSVPs in top 300Km*/
indbl=2.0;

   E->trace.extraq[j][0][kk]= indbl; }
/************************** Modified upto here ********************************************/

    else {
/* below */
E->trace.extraq[j][0][kk] = 0.0;
} } }
```

1.3 Generalizing the multi-tracer code

**Note This setup is done to generalize the tracer of any flavor that lie in the same layer. For different layer one has to make changes in the source code Ggrd_handling.c.

Previous Verisons:

```
   if(!E->control.ggrd_comp_smooth) {
/* limit to 0 or 1 */
if(indbl < .5)
indbl = 0.0;
if(indbl > .5  indbl < 1.5)
indbl=1.0;
E->trace.extraq[j][0][kk]= indbl; }

   else {
/* below */
E->trace.extraq[j][0][kk] = 0.0;
} } }
```

Newer Version:

/***Modified by Debanjan Pal to generalize multi-tracers in a single layer***/

```
    E->trace.extraq[j][0][kk]= round(indbl);}

   else {
/* below */
E->trace.extraq[j][0][kk] = 0.0;
} } }
```

1.4 Changes in Compostion_realated.c

It you are using thermochemical pile in addition to cratons defined with another tracer flavors, remember to add this changes to Composition_related.c. These changes assign different density to different tracer flavors. There are 2 changes in differnts parts of the code

1) Previous Verisons:

```
   /* default values ....  */
for (i=0; i<E->composition.ncomp; i++)  {
E->composition.buoyancy_ratio[i] = 1.0;
input_double_vector("buoyancy_ratio", E->composition.ncomp, E->composition.buoyancy_ratio,m);
}
```

Newer Verisons:

```
   /* default values ....  */

   /** Modified by Debanjan Pal to include different chemical density to diff tracer flavor.***/

   for (i=0; i<E->trace.nflavors; i++)  {
E->composition.buoyancy_ratio[i] = 1.0;
input_double_vector("buoyancy_ratio", E->trace.nflavors,, E->composition.buoyancy_ratio,m);
}
/************************** Modified upto here ********************************************/
```

2) Previous Verisons:

```
   for(k=0; k<E->composition.ncomp; k++)   {
fprintf(E->trace.fpt,"Buoyancy Ratio:  %f", E->composition.buoyancy_ratio[k]);
}
```

Newer Verisons:

```
   /** Modified by Debanjan Pal to include different chemical density to diff tracer flavor.***/

   for(k=0; k<E->trace.nflavors; k++)   {
fprintf(E->trace.fpt,"Buoyancy Ratio:  %f", E->composition.buoyancy_ratio[k]);
}

   /************************** Modified upto here ********************************************/
```

1.5 Input file for using the multi-tracer setup

In this example, I assume three tracer flavors, cratons of flavor 1, LLSVPs as flavor 2 and ambient mantle as 0. I assign composition viscosity prefactor of 100 to cratons, and buoyancy ratio of 0.15 to the LLSVPs. In the input tracer file crat.grd contains the geographical locations of cratons at 140 Ma whose third column is the tracer flavor, in this case the 3rd column (tracer flavor) is 1 for cratons and 0 for ambient.

```
 tracer=on

CDEPV=on

cdepv_ff=1,100,1 ## Takes viscosity in the sequential order of tracer flavor 0, 1, 2

tracer_flavors=3 ## No of tracer flavors

ic_method_for_flavors=1

z_interface=0.7000 ##depth 2550Km

ictracer_grd_layers=2

ictracer_grd_file="crat.grd"

tracer_flavors=3

chemical_buoyancy=on

buoy_type=1

buoyancy_ratio=0,0.15 ## Takes B values in the sequential order of tracer flavor 1, 2

reset_initial_composition=off
```

2 Defining Regional weak layer

To append anything related to viscosity see Viscosity_Structures.c.

This modification is done to implement regional weak layer, for example, introducing regional weak layer below 660km in the regions of intense downwelling.

Note: Works in layer/s below the lithopshere.

Used this temperature condition check to implement this:

$$T - T_{avg} < -0.1$$

where, T is the temperture at mesh point, T_{avg} (which is generally 0.5) is the average temperature of the layer.

Example if the mesh point temperature is 0.4 or below, then only the regional weak layer be will generated.

This change is for rheol=1, but the structure will be similar for other rheologies also.

Previous Version

```
   EEta[m][ (i-1)*vpts + jj ] = tempa*
exp( E->viscosity.E[l] * (E->viscosity.T[l] - temp));

   }
}

   break;
```

Newer Version

```
   /*** Modified by Debanjan Pal to introduce regional weak layer below 660 when there is slab ***/

   /* layer number n-1 */
if(l==3)
if((temp - 0.5) < -0.1)

   EEta[m][ (i-1)*vpts + jj ] = tempa*0.1*
exp( E->viscosity.E[l] * (E->viscosity.T[l] - temp));

   else
EEta[m][ (i-1)*vpts + jj ] = tempa*
exp( E->viscosity.E[l] * (E->viscosity.T[l] - temp));

   else
EEta[m][ (i-1)*vpts + jj ] = tempa*
exp( E->viscosity.E[l] * (E->viscosity.T[l] - temp));

   /************************* Modified upto here ************************************************/

   }
}

   break;
```

Note:
rheo=1 formulation:

$$\eta = \eta_0 \times exp((0.5 - T))$$

`tempa` is η_0 in the equation rheo = 1 which is multiplied with 0.1 indicating 10 times viscosity reduction. `temp` is the mesh point temperature and `l` is the layer no and layer counting starts from 0 in the source code. For example if you want to implement viscosity reduction in layer 4 of **num_mat**, then `l==3` in the source code.

3 Time-dependent moving weak zone

In the recent CitcomS version this functionality is disabled. To turn it on I appended the code Ggrd_ handling.c

```
if(E->control.ggrd_mat_is_code)

use_nearneighbor = TRUE;

/*** Modified by Debanjan Pal to to turn on time-dependent moving weak layer ***/

E->control.ggrd.time_hist.nvtimes=140;
```

I have given 140 Ma as all my models starts from 140 Ma onwards.

****

In the parameter file you have to add these inputs.

```
ggrd_mat_control=1

grd.time_hist.nvtimes=75

ggrd_time_hist_file="/free_slip_models/Subduction_ridge_slabs_postion/time.dat"

ggrd_mat_file="/free_slip_models/Subduction_ridge_slabs_postion/"
```

grd.time_hist.nvtime denotes the staring time of computation (75 Ma)

ggrd_time_hist_file denotes the time history file (I have included a sample file "time.dat" for example)

grd_mat_file denotes the ggrd file locations

You have too keep separate grd files for every Ma from 75 Ma onwards. Name the files as weak.grd and keep weak.grd in a folder. Rename the folder to the age.

For example in the folder Subduction_ridge_slabs_postion I have folders labelled from 0 to 75, and in each of these folders there is a grd file named weak.grd

**weak.grd has 3 columns, column 1 and 2 denotes longitude and latitude respectively, while the column 3 denotes the viscosity pre-factor.

**time.dat input example.

```
75 74
74 73
73 72
.. ..
.. ..
.. ..
.. ..
3 2
2 1
1 0
```

**Note Here the numbers in time.dat denotes the ages, the first line here suggest that the grd file weak.grd in folder 75 will be implemented between 75-74 and so on.

**Caution If you are running any simulation say 75 -55 Ma, still you require weak boundaries in 0 Ma, otherwise it will give error. You need weak.grd starting from 75 till 0 Ma, though you simulate only between 75-55 Ma.

4 Only letting the tracers to advect on the surface and do not let them to subduct (Still require more refinement).

I have done this modifications, to use tracers as weak zones in trenches. The problem with the original code is that with downwelling slabs the tracers also get subducted and after a certain time the weak zone do not persists. Two changes are required in Tracer_setup.c

Under the function static void predict_tracers in Tracer_Setup.c append the following lines.

1) Previous Verisons:

```
    x_pred=x0+velocity_vector[1]*dt;
y_pred=y0+velocity_vector[2]*dt;
z_pred=z0+velocity_vector[3]*dt;
```

Newer Verisons:

```
   x_pred=x0+velocity_vector[1]*dt;
y_pred=y0+velocity_vector[2]*dt;
z_pred=z0;
```

Under the function static void correct_tracers in Tracer_Setup.c append the following lines.

1) Previous Verisons:

```
   x_cor=x0 + dt * 0.5*(Vx0+Vx_pred);
y_cor=y0 + dt * 0.5*(Vy0+Vy_pred);
z_cor=z0 + dt * 0.5*(Vz0+Vz_pred);
```

Newer Verisons:

```
   x_cor=x0 + dt * 0.5*(Vx0+Vx_pred);
y_cor=y0 + dt * 0.5*(Vy0+Vy_pred);
z_cor=z0;
```

The idea is when the tracers are advected with the time evolving velocity field using Runge-Kutta method, I make the displacement along the z-axis to be 0.