

Lime Cheat sheet

Version 1.3

Christian Brinch, 2013

```
void
input(inputPars *par, image *img)
{
    par->parameter = value;
    img[i].parameter = value;
}
```

Required parameters:

(double) par->radius
Outer model radius in meters.

(double) par->minScale
Smallest scales sampled by grid.

(integer) par->pIntensity
Number of grid points.

(integer) par->sinkPoints
Number of surface grid points.

Optional parameters:

(integer) par->sampling
0 for spherical sampling, 1 for Cartesian sampling. Default is 0.

(double) par->tcmb
Temperature of the microwave background . The default value is 2.725 K.

(string) par->moldatfile[i]
Path to the i'th molecular data file.

(string) par->dust
Path to dust opacity table. The moldatfile and dust parameters are optional in the sense that at least one of them (or both) should be set.

(string) par->outputfile
Path to level population ascii output.

(string) par->binoutputfile
Path to level population binary output used to restart LIME.

(string) par->restart
File used to restart LIME with previously calculated populations stored in binary format.

(string) par->gridfile
Path to VTK grid output file.

(string) par->pregrid
Path to a file containing a predefined grid.

(integer) par->lte_only
Perform an LTE calculation only.

(integer) par->blend
Set this parameter to take line blending into account. The default is unset.

(integer) par->antialias
This parameter determines the level of antialiasing in the output images. A higher number makes better images but the code takes longer to run.

(integer) par->polarization
For continuum polarization calculations, this parameter should be set to 1. Default is 0.

Image parameters:

The img structure is an array so that img[i].value denotes the setting for the i'th image. Multiple images can be defined, i.e., img[0], img[1], etc.

Required image parameters:

(integer) img[i].pxls
Number of pixels per dimension.
(double) img[i].imgres
Image resolution in arcsec per pixel.

(double) img[i].theta
Inclination from 0 (face-on) to $\pi/2$ (edge-on)

(double) img[i].distance
Object distance in meters.

(integer) img[i].unit
0 for Kelvin, 1 for Jansky per pixel, 2 for SI units, and 3 for Solar luminosities per pixel. 4 gives a tau image cube.

(string) img[i].filename
Path to the output fits image file.

Optional image parameters:

(double) img[i].phi
Object rotation from 0 to 2π . Default is 0.

(double) img[i].source_vel
Source velocity offset. Default is 0.

(integer) img[i].nchan
Number of frequency channels.

(double) img[i].velres
Velocity resolution in meters per second.

(integer) `img[i].trans`
Zero-indexed J quantum number of the line centered on the image, i.e., 0 for $J = 1-0$.

(double) `img[i].freq`
Center frequency of the output image.

(double) `img[i].bandwidth`
Width of the frequency axis in Hz.

The `nchan`, `velres`, `trans`, `freq`, and `bandwidth` parameters are optional according to the following rules:

For continuum images: LIME decides to make continuum images if the `nchan` parameters is not set. If `nchan` is not set, only `freq` can be set. If `velres`, `trans`, `bandwidth`, or `moldatfile` are set, LIME will produce an error. Remember to set the dust parameter

For line images: Use *either* `nchan`, `velres`, and `trans` *or* `nchan`, `freq`, and `bandwidth`. Any other combination will produce an error. Remember to set `moldatfile` and optionally `dust`.

```
void
density(double x, double y,
double z, double *density){
    density[0] = f(x,y,z);
    density[1] = f(x,y,z);
    ...
    density[n] = f(x,y,z);
}
```

`density[i]` is the number density of the i 'th collision partner. LIME will produce an error if the number of collision partner density profiles (n)

does not match the number of collision partners in the data file specified by `par->moldatfile`.

```
void
abundance(double x, double y,
double z, double *abundance){
    abundance[0] = f(x,y,z);
    abundance[1] = f(x,y,z);
    ...
    abundance[n] = f(x,y,z);
}
```

`abundance[i]` gives the fractional abundance of the i 'th molecular species given by the i 'th `moldatfile`. If the number of abundance profiles (n) does not match the number of molecular datafiles, LIME will produce an error.

```
void
temperature(double x, double y,
double z, double *temperature){
    temperature[0] = f(x,y,z);
    temperature[1] = f(x,y,z);
}
```

`temperature[0]` is the kinetic gas temperature. `temperature[1]` is the dust temperature. `temperature[1]` is optional. If left out, LIME will assume that $T_{\text{dust}} = T_{\text{gas}}$.

```
void
doppler(double x, double y,
double z, double *doppler){
    *doppler = f(x,y,z);
}
```

This function gives the value of the Doppler b -parameter. Notice that this parameter is a single value and not an array (`*doppler` is correct; `doppler[0]` is wrong).

```
void
velocity(double x, double y,
double z, double *vel){
    vel[0] = f(x,y,z);
    vel[1] = f(x,y,z);
    vel[2] = f(x,y,z);
}
```

`vel` holds the x , y , and z components of the velocity vectors.

```
void
magfield(double x, double y,
double z, double *B){
    B[0] = f(x,y,z);
    B[1] = f(x,y,z);
    B[2] = f(x,y,z);
}
```

`B` holds the x , y , and z components of the magnetic field vectors. This function is by default located in `src/magfieldfit.c` for backwards compatability.

Important notice:

All values in the LIME input file should be given in SI units. That means velocities in m s^{-1} , temperatures in K, and densities in m^{-3} .

A number of "hidden" parameters can be found in the file `lime.h`. These parameters should generally not be adjusted. Changes made to `lime.h` are global and do not relate to a specific model input file.