GPS Code Collection

Generated by Doxygen 1.8.11

ii CONTENTS

Contents

1	Main	n Page	2
2	Data	a Structure Index	2
	2.1	Data Structures	2
3	File	Index	2
	3.1	File List	2
4	Data	Structure Documentation	3
	4.1	atm_t Struct Reference	3
		4.1.1 Detailed Description	3
		4.1.2 Field Documentation	4
	4.2	ctl_t Struct Reference	5
		4.2.1 Detailed Description	6
		4.2.2 Field Documentation	6
	4.3	gps_t Struct Reference	9
		4.3.1 Detailed Description	10
		4.3.2 Field Documentation	10
	4.4	los_t Struct Reference	12
		4.4.1 Detailed Description	12
		4.4.2 Field Documentation	12
	4.5	met_t Struct Reference	14
		4.5.1 Detailed Description	14
		4.5.2 Field Documentation	14
	4.6	obs_t Struct Reference	15
		4.6.1 Detailed Description	16
		4.6.2 Field Documentation	16
	4.7	tbl_t Struct Reference	18
		4.7.1 Detailed Description	18
		4.7.2 Field Documentation	19

5	File	Documentation	20
	5.1	events.c File Reference	20
		5.1.1 Function Documentation	20
	5.2	events.c	22
	5.3	jurassic.c File Reference	23
		5.3.1 Detailed Description	25
		5.3.2 Function Documentation	26
	5.4	jurassic.c	93
	5.5	jurassic.h File Reference	150
		5.5.1 Detailed Description	153
		5.5.2 Function Documentation	153
	5.6	jurassic.h	220
	5.7	libgps.c File Reference	227
		5.7.1 Function Documentation	228
	5.8	libgps.c	244
	5.9	libgps.h File Reference	255
		5.9.1 Function Documentation	256
	5.10	libgps.h	271
	5.11	map.c File Reference	273
		5.11.1 Function Documentation	273
	5.12	map.c	274
	5.13	perturbation.c File Reference	275
		5.13.1 Function Documentation	275
	5.14	perturbation.c	277
	5.15	prof.c File Reference	278
		5.15.1 Function Documentation	278
	5.16	prof.c	279
	5.17	response.c File Reference	280
		5.17.1 Function Documentation	280
	5.18	response.c	282
	5.19	variance.c File Reference	284
		5.19.1 Function Documentation	284
	5.20	variance.c	287

Index 291

1 Main Page

The JUelich RApid Spectral SImulation Code (JURASSIC) is a fast radiative transfer model for the mid-infrared spectral region. This reference manual provides information on the algorithms and data structures used in the code. Further information can be found at: http://www.fz-juelich.de/ias/jsc/jurassic

2 Data Structure Index

2.1 Data Structures

Here are the data structures with brief descriptions:

atm_t	
Atmospheric data	3
·	
ctl_t	
Forward model control parameters	5
gps_t	
GPS-RO profile data	9
los_t	
Line-of-sight data	12
met_t	
Meteorological data	14
obs_t	
Observation geometry and radiance data	15
tbl_t	
Emissivity look-up tables	18

3 File Index

3.1 File List

Here is a list of all files with brief descriptions:

events.c	20
jurassic.c JURASSIC library definitions	23
jurassic.h JURASSIC library declarations	150
libgps.c	227

libgps.h	255
map.c	273
perturbation.c	275
prof.c	278
response.c	280
variance.c	284

4 Data Structure Documentation

4.1 atm_t Struct Reference

Atmospheric data.

```
#include <jurassic.h>
```

Data Fields

• int np

Number of data points.

• double time [NP]

Time (seconds since 2000-01-01T00:00Z).

• double z [NP]

Altitude [km].

• double lon [NP]

Longitude [deg].

• double lat [NP]

Latitude [deg].

• double p [NP]

Pressure [hPa].

• double t [NP]

Temperature [K].

• double q [NG][NP]

Volume mixing ratio.

• double k [NW][NP]

Extinction [1/km].

4.1.1 Detailed Description

Atmospheric data.

Definition at line 219 of file jurassic.h.

```
4.1.2 Field Documentation
4.1.2.1 int atm_t::np
Number of data points.
Definition at line 222 of file jurassic.h.
4.1.2.2 double atm_t::time[NP]
Time (seconds since 2000-01-01T00:00Z).
Definition at line 225 of file jurassic.h.
4.1.2.3 double atm_t::z[NP]
Altitude [km].
Definition at line 228 of file jurassic.h.
4.1.2.4 double atm_t::lon[NP]
Longitude [deg].
Definition at line 231 of file jurassic.h.
4.1.2.5 double atm_t::lat[NP]
Latitude [deg].
Definition at line 234 of file jurassic.h.
4.1.2.6 double atm_t::p[NP]
Pressure [hPa].
Definition at line 237 of file jurassic.h.
4.1.2.7 double atm_t::t[NP]
Temperature [K].
Definition at line 240 of file jurassic.h.
4.1.2.8 double atm_t::q[NG][NP]
Volume mixing ratio.
Definition at line 243 of file jurassic.h.
```

4.1.2.9 double atm_t::k[NW][NP]

Extinction [1/km].

Definition at line 246 of file jurassic.h.

The documentation for this struct was generated from the following file:

· jurassic.h

4.2 ctl_t Struct Reference

Forward model control parameters.

```
#include <jurassic.h>
```

Data Fields

• int ng

Number of emitters.

char emitter [NG][LEN]

Name of each emitter.

int nd

Number of radiance channels.

• int nw

Number of spectral windows.

• double nu [ND]

Centroid wavenumber of each channel [cm $^{\wedge}$ -1].

• int window [ND]

Window index of each channel.

char tblbase [LEN]

Basename for table files and filter function files.

double hydz

Reference height for hydrostatic pressure profile (-999 to skip) [km].

• int ctm_co2

Compute CO2 continuum (0=no, 1=yes).

• int ctm_h2o

Compute H2O continuum (0=no, 1=yes).

• int ctm_n2

Compute N2 continuum (0=no, 1=yes).

int ctm_o2

Compute O2 continuum (0=no, 1=yes).

• int reirac

Take into account refractivity (0=no, 1=yes).

double rayds

Maximum step length for raytracing [km].

double raydz

Vertical step length for raytracing [km].

• char fov [LEN]

Field-of-view data file.

· double retp_zmin Minimum altitude for pressure retrieval [km]. double retp_zmax Maximum altitude for pressure retrieval [km]. double rett_zmin Minimum altitude for temperature retrieval [km]. · double rett_zmax Maximum altitude for temperature retrieval [km]. double retq_zmin [NG] Minimum altitude for volume mixing ratio retrieval [km]. double retq_zmax [NG] Maximum altitude for volume mixing ratio retrieval [km]. • double retk_zmin [NW] Minimum altitude for extinction retrieval [km]. double retk_zmax [NW] Maximum altitude for extinction retrieval [km]. int write_bbt Use brightness temperature instead of radiance (0=no, 1=yes). · int write matrix Write matrix file (0=no, 1=yes). 4.2.1 Detailed Description Forward model control parameters. Definition at line 251 of file jurassic.h. 4.2.2 Field Documentation 4.2.2.1 int ctl_t::ng Number of emitters. Definition at line 254 of file jurassic.h. 4.2.2.2 char ctl_t::emitter[NG][LEN] Name of each emitter. Definition at line 257 of file jurassic.h. 4.2.2.3 int ctl_t::nd Number of radiance channels. Definition at line 260 of file jurassic.h.

```
4.2.2.4 int ctl_t::nw
Number of spectral windows.
Definition at line 263 of file jurassic.h.
4.2.2.5 double ctl_t::nu[ND]
Centroid wavenumber of each channel [cm^-1].
Definition at line 266 of file jurassic.h.
4.2.2.6 int ctl_t::window[ND]
Window index of each channel.
Definition at line 269 of file jurassic.h.
4.2.2.7 char ctl_t::tblbase[LEN]
Basename for table files and filter function files.
Definition at line 272 of file jurassic.h.
4.2.2.8 double ctl_t::hydz
Reference height for hydrostatic pressure profile (-999 to skip) [km].
Definition at line 275 of file jurassic.h.
4.2.2.9 int ctl_t::ctm_co2
Compute CO2 continuum (0=no, 1=yes).
Definition at line 278 of file jurassic.h.
4.2.2.10 int ctl_t::ctm_h2o
Compute H2O continuum (0=no, 1=yes).
Definition at line 281 of file jurassic.h.
4.2.2.11 int ctl_t::ctm_n2
Compute N2 continuum (0=no, 1=yes).
Definition at line 284 of file jurassic.h.
4.2.2.12 int ctl_t::ctm_o2
Compute O2 continuum (0=no, 1=yes).
Definition at line 287 of file jurassic.h.
```

```
4.2.2.13 int ctl_t::refrac
Take into account refractivity (0=no, 1=yes).
Definition at line 290 of file jurassic.h.
4.2.2.14 double ctl_t::rayds
Maximum step length for raytracing [km].
Definition at line 293 of file jurassic.h.
4.2.2.15 double ctl_t::raydz
Vertical step length for raytracing [km].
Definition at line 296 of file jurassic.h.
4.2.2.16 char ctl_t::fov[LEN]
Field-of-view data file.
Definition at line 299 of file jurassic.h.
4.2.2.17 double ctl_t::retp_zmin
Minimum altitude for pressure retrieval [km].
Definition at line 302 of file jurassic.h.
4.2.2.18 double ctl_t::retp_zmax
Maximum altitude for pressure retrieval [km].
Definition at line 305 of file jurassic.h.
4.2.2.19 double ctl_t::rett_zmin
Minimum altitude for temperature retrieval [km].
Definition at line 308 of file jurassic.h.
4.2.2.20 double ctl_t::rett_zmax
Maximum altitude for temperature retrieval [km].
Definition at line 311 of file jurassic.h.
4.2.2.21 double ctl_t::retq_zmin[NG]
Minimum altitude for volume mixing ratio retrieval [km].
```

Definition at line 314 of file jurassic.h.

```
4.2.2.22 double ctl_t::retq_zmax[NG]
Maximum altitude for volume mixing ratio retrieval [km].
Definition at line 317 of file jurassic.h.
4.2.2.23 double ctl_t::retk_zmin[NW]
Minimum altitude for extinction retrieval [km].
Definition at line 320 of file jurassic.h.
4.2.2.24 double ctl_t::retk_zmax[NW]
Maximum altitude for extinction retrieval [km].
Definition at line 323 of file jurassic.h.
4.2.2.25 int ctl_t::write_bbt
Use brightness temperature instead of radiance (0=no, 1=yes).
Definition at line 326 of file jurassic.h.
4.2.2.26 int ctl_t::write_matrix
Write matrix file (0=no, 1=yes).
Definition at line 329 of file jurassic.h.
The documentation for this struct was generated from the following file:
    · jurassic.h
4.3 gps_t Struct Reference
GPS-RO profile data.
#include <libgps.h>
```

Data Fields

• int nds

Number of profiles.

• int nz [NDS]

Number of altitudes per profile.

· double time [NDS]

Time (seconds since 2000-01-01T00:00Z).

• double z [NDS][NZ]

Altitude [km].

• double lon [NDS][NZ]

Longitude [deg].

double lat [NDS][NZ]

Latitude [deg].

double p [NDS][NZ]

Pressure [hPa].

· double t [NDS][NZ]

Temperature [K].

double wv [NDS][NZ]

Water vapor volume mixing ratio [ppm].

double pt [NDS][NZ]

Temperature perturbation [K].

double th [NDS]

Tropopause height [km].

4.3.1 Detailed Description

GPS-RO profile data.

Definition at line 40 of file libgps.h.

4.3.2 Field Documentation

4.3.2.1 int gps_t::nds

Number of profiles.

Definition at line 43 of file libgps.h.

4.3.2.2 int gps_t::nz[NDS]

Number of altitudes per profile.

Definition at line 46 of file libgps.h.

4.3.2.3 double gps_t::time[NDS]

Time (seconds since 2000-01-01T00:00Z).

Definition at line 49 of file libgps.h.

```
4.3.2.4 double gps_t::z[NDS][NZ]
Altitude [km].
Definition at line 52 of file libgps.h.
4.3.2.5 double gps_t::lon[NDS][NZ]
Longitude [deg].
Definition at line 55 of file libgps.h.
4.3.2.6 double gps_t::lat[NDS][NZ]
Latitude [deg].
Definition at line 58 of file libgps.h.
4.3.2.7 double gps_t::p[NDS][NZ]
Pressure [hPa].
Definition at line 61 of file libgps.h.
4.3.2.8 double gps_t::t[NDS][NZ]
Temperature [K].
Definition at line 64 of file libgps.h.
4.3.2.9 double gps_t::wv[NDS][NZ]
Water vapor volume mixing ratio [ppm].
Definition at line 67 of file libgps.h.
4.3.2.10 double gps_t::pt[NDS][NZ]
Temperature perturbation [K].
Definition at line 70 of file libgps.h.
4.3.2.11 double gps_t::th[NDS]
Tropopause height [km].
Definition at line 73 of file libgps.h.
The documentation for this struct was generated from the following file:
```

Generated by Doxygen

• libgps.h

4.4 los_t Struct Reference

```
Line-of-sight data.
```

```
#include <jurassic.h>
```

Data Fields

• int np

Number of LOS points.

• double z [NLOS]

Altitude [km].

· double lon [NLOS]

Longitude [deg].

· double lat [NLOS]

Latitude [deg].

• double p [NLOS]

Pressure [hPa].

• double t [NLOS]

Temperature [K].

• double q [NG][NLOS]

Volume mixing ratio.

double k [NW][NLOS]

Extinction [1/km].

· double tsurf

Surface temperature [K].

• double ds [NLOS]

Segment length [km].

• double u [NG][NLOS]

Column density [molecules/cm²].

4.4.1 Detailed Description

Line-of-sight data.

Definition at line 334 of file jurassic.h.

4.4.2 Field Documentation

4.4.2.1 int los_t::np

Number of LOS points.

Definition at line 337 of file jurassic.h.

4.4.2.2 double los_t::z[NLOS]

Altitude [km].

Definition at line 340 of file jurassic.h.

```
4.4.2.3 double los_t::lon[NLOS]
Longitude [deg].
Definition at line 343 of file jurassic.h.
4.4.2.4 double los_t::lat[NLOS]
Latitude [deg].
Definition at line 346 of file jurassic.h.
4.4.2.5 double los_t::p[NLOS]
Pressure [hPa].
Definition at line 349 of file jurassic.h.
4.4.2.6 double los_t::t[NLOS]
Temperature [K].
Definition at line 352 of file jurassic.h.
4.4.2.7 double los_t::q[NG][NLOS]
Volume mixing ratio.
Definition at line 355 of file jurassic.h.
4.4.2.8 double los_t::k[NW][NLOS]
Extinction [1/km].
Definition at line 358 of file jurassic.h.
4.4.2.9 double los_t::tsurf
Surface temperature [K].
Definition at line 361 of file jurassic.h.
4.4.2.10 double los_t::ds[NLOS]
Segment length [km].
Definition at line 364 of file jurassic.h.
```

```
4.4.2.11 double los_t::u[NG][NLOS]
Column density [molecules/cm<sup>2</sup>].
Definition at line 367 of file jurassic.h.
The documentation for this struct was generated from the following file:
    · jurassic.h
4.5 met_t Struct Reference
Meteorological data.
#include <libgps.h>
Data Fields
    • double time
          Time [s].
    • int nx
          Number of longitudes.
    • int ny
          Number of latitudes.
    • int np
          Number of pressure levels.
    · double lon [EX]
          Longitude [deg].

    double lat [EY]

          Latitude [deg].

 double p [EP]

          Pressure [hPa].
    • float t [EX][EY][EP]
          Temperature [K].
4.5.1 Detailed Description
Meteorological data.
Definition at line 78 of file libgps.h.
4.5.2 Field Documentation
4.5.2.1 double met_t::time
Time [s].
```

Definition at line 81 of file libgps.h.

```
4.5.2.2 int met_t::nx
Number of longitudes.
Definition at line 84 of file libgps.h.
4.5.2.3 int met_t::ny
Number of latitudes.
Definition at line 87 of file libgps.h.
4.5.2.4 int met_t::np
Number of pressure levels.
Definition at line 90 of file libgps.h.
4.5.2.5 double met_t::lon[EX]
Longitude [deg].
Definition at line 93 of file libgps.h.
4.5.2.6 double met_t::lat[EY]
Latitude [deg].
Definition at line 96 of file libgps.h.
4.5.2.7 double met_t::p[EP]
Pressure [hPa].
Definition at line 99 of file libgps.h.
4.5.2.8 float met_t::t[EX][EY][EP]
Temperature [K].
Definition at line 102 of file libgps.h.
The documentation for this struct was generated from the following file:
    • libgps.h
4.6 obs_t Struct Reference
Observation geometry and radiance data.
#include <jurassic.h>
```

Data Fields

• int nr

Number of ray paths.

• double time [NR]

Time (seconds since 2000-01-01T00:00Z).

• double obsz [NR]

Observer altitude [km].

• double obsion [NR]

Observer longitude [deg].

double obslat [NR]

Observer latitude [deg].

double vpz [NR]

View point altitude [km].

double vplon [NR]

View point longitude [deg].

double vplat [NR]

View point latitude [deg].

double tpz [NR]

Tangent point altitude [km].

• double tplon [NR]

Tangent point longitude [deg].

double tplat [NR]

Tangent point latitude [deg].

• double tau [ND][NR]

Transmittance of ray path.

double rad [ND][NR]

Radiance [W/(m^2 sr cm $^-$ -1)].

4.6.1 Detailed Description

Observation geometry and radiance data.

Definition at line 372 of file jurassic.h.

4.6.2 Field Documentation

4.6.2.1 int obs_t::nr

Number of ray paths.

Definition at line 375 of file jurassic.h.

4.6.2.2 double obs_t::time[NR]

Time (seconds since 2000-01-01T00:00Z).

Definition at line 378 of file jurassic.h.

4.6.2.3 double obs_t::obsz[NR] Observer altitude [km]. Definition at line 381 of file jurassic.h. 4.6.2.4 double obs_t::obslon[NR] Observer longitude [deg]. Definition at line 384 of file jurassic.h. 4.6.2.5 double obs_t::obslat[NR] Observer latitude [deg]. Definition at line 387 of file jurassic.h. 4.6.2.6 double obs_t::vpz[NR] View point altitude [km]. Definition at line 390 of file jurassic.h. 4.6.2.7 double obs_t::vplon[NR] View point longitude [deg]. Definition at line 393 of file jurassic.h. 4.6.2.8 double obs_t::vplat[NR] View point latitude [deg]. Definition at line 396 of file jurassic.h. 4.6.2.9 double obs_t::tpz[NR] Tangent point altitude [km]. Definition at line 399 of file jurassic.h. 4.6.2.10 double obs_t::tplon[NR] Tangent point longitude [deg]. Definition at line 402 of file jurassic.h. 4.6.2.11 double obs_t::tplat[NR] Tangent point latitude [deg]. Definition at line 405 of file jurassic.h.

```
4.6.2.12 double obs_t::tau[ND][NR]
Transmittance of ray path.
Definition at line 408 of file jurassic.h.
4.6.2.13 double obs_t::rad[ND][NR]
Radiance [W/(m^2 sr cm^--1)].
Definition at line 411 of file jurassic.h.
The documentation for this struct was generated from the following file:
    · jurassic.h
4.7 tbl_t Struct Reference
Emissivity look-up tables.
#include <jurassic.h>
Data Fields
    • int np [NG][ND]
          Number of pressure levels.
    int nt [NG][ND][TBLNP]
          Number of temperatures.
    • int nu [NG][ND][TBLNP][TBLNT]
          Number of column densities.

    double p [NG][ND][TBLNP]

          Pressure [hPa].
    • double t [NG][ND][TBLNP][TBLNT]
          Temperature [K].
    • float u [NG][ND][TBLNP][TBLNT][TBLNU]
          Column density [molecules/cm<sup>2</sup>].
    • float eps [NG][ND][TBLNP][TBLNT][TBLNU]
          Emissivity.
    • double st [TBLNS]
          Source function temperature [K].
    • double sr [ND][TBLNS]
          Source function radiance [W/(m^2 sr cm^--1)].
4.7.1 Detailed Description
Emissivity look-up tables.
```

Definition at line 416 of file jurassic.h.

4.7.2 Field Documentation 4.7.2.1 int tbl_t::np[NG][ND] Number of pressure levels. Definition at line 419 of file jurassic.h. 4.7.2.2 int tbl_t::nt[NG][ND][TBLNP] Number of temperatures. Definition at line 422 of file jurassic.h. 4.7.2.3 int tbl_t::nu[NG][ND][TBLNP][TBLNT] Number of column densities. Definition at line 425 of file jurassic.h. 4.7.2.4 double tbl_t::p[NG][ND][TBLNP] Pressure [hPa]. Definition at line 428 of file jurassic.h. 4.7.2.5 double tbl_t::t[NG][ND][TBLNP][TBLNT] Temperature [K]. Definition at line 431 of file jurassic.h. 4.7.2.6 float tbl_t::u[NG][ND][TBLNP][TBLNT][TBLNU] Column density [molecules/cm²]. Definition at line 434 of file jurassic.h. 4.7.2.7 float tbl_t::eps[NG][ND][TBLNP][TBLNT][TBLNU] Emissivity. Definition at line 437 of file jurassic.h. 4.7.2.8 double tbl_t::st[TBLNS] Source function temperature [K]. Definition at line 440 of file jurassic.h.

4.7.2.9 double tbl_t::sr[ND][TBLNS]

Source function radiance [W/(m² sr cm⁻¹)].

Definition at line 443 of file jurassic.h.

The documentation for this struct was generated from the following file:

· jurassic.h

5 File Documentation

5.1 events.c File Reference

Functions

```
• int main (int argc, char *argv[])
```

5.1.1 Function Documentation

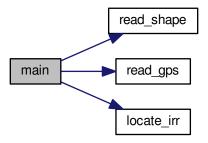
5.1.1.1 int main (int argc, char * argv[])

Definition at line 3 of file events.c.

```
00005
00006
00007
        gps_t *gps;
80000
00009
        FILE *in, *out;
00010
00011
        static double ptmin, ptmax, se[NZ], sz[NZ], w, wmax, wsum2 = 1.0, var;
00012
00013
        static int iarg, ids, idx, iz, sn;
00014
00015
        /* Allocate... */
00016
        ALLOC(gps, gps_t, 1);
00017
00018
        /* Check arguments... */
00019
        if (argc < 5)
00020
        ERRMSG("Give parameters: <ctl> <events.tab> <sens.tab> "
00021
                  "<gps1.nc> [<gps2.nc> ...]");
00022
        /* Create file... */
printf("Write event data: %s\n", argv[2]);
00023
00024
        if (!(out = fopen(argv[2], "w")))
00025
00026
          ERRMSG("Cannot create file!");
00027
00028
        /* Write header... */
        00029
00030
                 "# $2 = longitude [deg]\n"
00031
                 "# $3 = latitude [deg]\n"
00032
00033
                 "# $4 = minimum perturbation [K] \n"
                 "# $5 = maximum perturbation [K]\n"
"# $6 = temperature variance [K^2]\n\n");
00034
00035
00036
        /* Read vertical sensitivity function... */
if (argv[3][0] != '-') {
00037
00038
00039
         read_shape(argv[3], sz, se, &sn);
00040
          if (sn > NZ)
00041
00042
            ERRMSG("Too many data points!");
00043
00044
        /* Loop over data files... */
00045
        for (iarg = 4; iarg < argc; iarg++) {</pre>
```

```
00046
00047
           /* Read gps data... */
00048
           if (!(in = fopen(argv[iarg], "r")))
00049
            continue;
00050
           else (
00051
            fclose(in);
00052
            read_gps(argv[iarg], gps);
00053
00054
00055
           /\!\star Loop over profiles... \!\star/
00056
           for (ids = 0; ids < gps->nds; ids++) {
00057
00058
             /* Check tropopause height... */
00059
             if (!gsl_finite(gps->th[ids]))
00060
00061
             /* Multiply with vertical sensitivity function... */ if (argv[3][0] != '-') {
00062
00063
               wmax = wsum2 = 0;
00064
00065
               for (iz = 0; iz < gps->nz[ids]; iz++) {
00066
                 if (gps->z[ids][iz] < sz[0] || gps->z[ids][iz] > sz[sn - 1])
                 w = 0;
else {
00067
00068
00069
                   idx = locate_irr(sz, sn, gps->z[ids][iz]);
00070
00071
                    LIN(sz[idx], se[idx], sz[idx + 1], se[idx + 1],
00072
                          gps->z[ids][iz]);
00073
00074
                 if (gsl_finite(gps->t[ids][iz]) && gps->pt[ids][iz]) {
00075
                   gps->pt[ids][iz] *= w;
wmax = GSL_MAX(w, wmax);
00076
00077
                    wsum2 += gsl_pow_2(w);
00078
00079
              for (iz = 0; iz < gps->nz[ids]; iz++)
   gps->pt[ids][iz] /= wmax;
wsum2 /= gsl_pow_2(wmax);
08000
00081
00082
00083
00084
00085
             /\star Get minimum and maximum perturbation... \star/
             ptmin = ptmax = var = 0;
for (iz = 0; iz < gps->nz[ids]; iz++)
   if (gsl_finite(gps->pt[ids][iz])) {
00086
00087
00088
                 ptmin = GSL_MIN(ptmin, gps->pt[ids][iz]);
00089
00090
                 ptmax = GSL_MAX(ptmax, gps->pt[ids][iz]);
00091
                  var += gsl_pow_2(gps->pt[ids][iz]) / wsum2;
00092
00093
             00094
00095
00096
00097
00098
00099
        }
00100
         /* Close file... */
00101
        fclose(out);
00103
00104
         /* Free... */
00105
        free(gps);
00106
00107
        return EXIT_SUCCESS;
00108 }
```

Here is the call graph for this function:



5.2 events.c

```
00001 #include "libgps.h"
00002
00003 int main(
00004
       int argc,
00005
       char *argv[]) {
00006
00007
       gps_t *gps;
00008
00009
       FILE *in, *out;
00010
00011
        static double ptmin, ptmax, se[NZ], sz[NZ], w, wmax, wsum2 = 1.0, var;
00012
00013
        static int iarg, ids, idx, iz, sn;
00014
00015
        /* Allocate... */
00016
        ALLOC(gps, gps_t, 1);
00017
00018
        /* Check arguments... ∗/
00019
        if (argc < 5)
          00020
00021
00022
00023
        /* Create file... */
       printf("Write event data: %s\n", argv[2]);
if (!(out = fopen(argv[2], "w")))
00024
00025
         ERRMSG("Cannot create file!");
00026
00027
00028
        /* Write header... */
00029
        fprintf(out,
00030
                "# $1 = time [s] \n"
                "# $2 = longitude [deg]\n"
"# $3 = latitude [deg]\n"
00031
00032
00033
                "# $4 = minimum perturbation [K]\n"
                "# $5 = maximum perturbation [K]\n"
00034
00035
                "# $6 = temperature variance [K^2]\n\n");
00036
00037
        /\star Read vertical sensitivity function... \star/
        if (argv[3][0] != '-') {
00038
         read_shape(argv[3], sz, se, &sn);
if (sn > NZ)
00039
00040
00041
            ERRMSG("Too many data points!");
00042
00043
00044
        /* Loop over data files... */
00045
        for (iarg = 4; iarg < argc; iarg++) {</pre>
00046
00047
          /* Read gps data... */
00048
          if (!(in = fopen(argv[iarg], "r")))
00049
            continue;
          else {
00050
00051
           fclose(in);
            read_gps(argv[iarg], gps);
00052
00053
00054
```

```
/* Loop over profiles... */
00056
          for (ids = 0; ids < gps->nds; ids++) {
00057
00058
            /\star Check tropopause height... \star/
00059
            if (!gsl_finite(gps->th[ids]))
00060
              continue:
00061
00062
            /\star Multiply with vertical sensitivity function... \star/
00063
            if (argv[3][0] != '-') {
00064
              wmax = wsum2 = 0;
              for (iz = 0; iz < gps->nz[ids]; iz++) {
00065
                if (gps->z[ids][iz] < sz[0] || gps->z[ids][iz] > sz[sn - 1])
00066
                w = 0;
else {
00067
00068
00069
                  idx = locate_irr(sz, sn, gps->z[ids][iz]);
00070
                    LIN(sz[idx], se[idx], sz[idx + 1], se[idx + 1],
00071
00072
                         gps->z[ids][iz]);
00073
00074
                if (gsl_finite(gps->t[ids][iz]) && gps->pt[ids][iz]) {
                  gps->pt[ids][iz] *= w;
wmax = GSL_MAX(w, wmax);
00075
00076
00077
                  wsum2 += gsl_pow_2(w);
00078
                }
00079
08000
              for (iz = 0; iz < gps->nz[ids]; iz++)
00081
                gps->pt[ids][iz] /= wmax;
00082
              wsum2 /= gsl_pow_2(wmax);
00083
00084
00085
            /\star Get minimum and maximum perturbation... \star/
00086
            ptmin = ptmax = var = 0;
00087
            for (iz = 0; iz < gps->nz[ids]; iz++)
88000
              if (gsl_finite(gps->pt[ids][iz]))
               ptmin = GSL_MIN(ptmin, gps->pt[ids][iz]);
ptmax = GSL_MAX(ptmax, gps->pt[ids][iz]);
00089
00090
00091
                var += gsl_pow_2(gps->pt[ids][iz]) / wsum2;
00092
00093
            00094
00095
00096
00097
00098
          }
00099
00100
00101
        /* Close file... */
00102
        fclose(out);
00103
00104
        /* Free... */
00105
        free(gps);
00106
00107
        return EXIT_SUCCESS;
00108 }
```

5.3 jurassic.c File Reference

JURASSIC library definitions.

Functions

• size_t atm2x (ctl_t *ctl, atm_t *atm, gsl_vector *x, int *iqa, int *ipa)

Compose state vector or parameter vector.

void atm2x_help (atm_t *atm, double zmin, double zmax, double *value, int val_iqa, gsl_vector *x, int *iqa, int *ipa, size_t *n)

Add elements to state vector.

• double brightness (double rad, double nu)

Compute brightness temperature.

void cart2geo (double *x, double *z, double *lon, double *lat)

Convert Cartesian coordinates to geolocation.

void climatology (ctl_t *ctl, atm_t *atm)

Interpolate climatological data. • double ctmco2 (double nu, double p, double t, double u) Compute carbon dioxide continuum (optical depth). double ctmh2o (double nu, double p, double t, double q, double u) Compute water vapor continuum (optical depth). double ctmn2 (double nu, double p, double t) Compute nitrogen continuum (absorption coefficient). double ctmo2 (double nu, double p, double t) Compute oxygen continuum (absorption coefficient). void copy_atm (ctl_t *ctl, atm_t *atm_dest, atm_t *atm_src, int init) Copy and initialize atmospheric data. void copy_obs (ctl_t *ctl, obs_t *obs_dest, obs_t *obs_src, int init) Copy and initialize observation data. • int find_emitter (ctl_t *ctl, const char *emitter) Find index of an emitter. void formod (ctl t *ctl, atm t *atm, obs t *obs) Determine ray paths and compute radiative transfer. void formod_continua (ctl_t *ctl, los_t *los, int ip, double *beta) Compute absorption coefficient of continua. void formod fov (ctl t *ctl, obs t *obs) Apply field of view convolution. void formod_pencil (ctl_t *ctl, atm_t *atm, obs_t *obs, int ir) Compute radiative transfer for a pencil beam. void formod_srcfunc (ctl_t *ctl, tbl_t *tbl, double t, double *src) Compute Planck source function. void geo2cart (double z, double lon, double lat, double *x) Convert geolocation to Cartesian coordinates. void hydrostatic (ctl_t *ctl, atm_t *atm) Set hydrostatic equilibrium. void idx2name (ctl t *ctl, int idx, char *quantity) Determine name of state vector quantity for given index. void init tbl (ctl t *ctl, tbl t *tbl) Initialize look-up tables. void intpol_atm (ctl_t *ctl, atm_t *atm, double z, double *p, double *t, double *q, double *k) Interpolate atmospheric data. void intpol_tbl (ctl_t *ctl, tbl_t *tbl, los_t *los, int ip, double tau_path[NG][ND], double tau_seg[ND]) Get transmittance from look-up tables. • double intpol_tbl_eps (tbl_t *tbl, int ig, int id, int ip, int it, double u) Interpolate emissivity from look-up tables. double intpol_tbl_u (tbl_t *tbl, int ig, int id, int ip, int it, double eps) Interpolate column density from look-up tables. • void jsec2time (double jsec, int *year, int *mon, int *day, int *hour, int *min, int *sec, double *remain) Convert seconds to date. void kernel (ctl t *ctl, atm t *atm, obs t *obs, gsl matrix *k) Compute Jacobians.

Find array index in float array.

int locate_irr (double *xx, int n, double x)
 Find array index for irregular grid.
 int locate_reg (double *xx, int n, double x)
 Find array index for regular grid.
 int locate_tbl (float *xx, int n, double x)

```
    size_t obs2y (ctl_t *ctl, obs_t *obs, gsl_vector *y, int *ida, int *ira)

           Compose measurement vector.

    double planck (double t, double nu)

          Compute Planck function.

    void raytrace (ctl_t *ctl, atm_t *atm, obs_t *obs, los_t *los, int ir)

          Do ray-tracing to determine LOS.
    • void read_atm (const char *dirname, const char *filename, ctl_t *ctl, atm_t *atm)
          Read atmospheric data.
    void read_ctl (int argc, char *argv[], ctl_t *ctl)
          Read forward model control parameters.

    void read_matrix (const char *dirname, const char *filename, gsl_matrix *matrix)

          Read matrix.

    void read obs (const char *dirname, const char *filename, ctl t *ctl, obs t *obs)

          Read observation data.

    void read_shape (const char *filename, double *x, double *y, int *n)

          Read shape function.
    • double refractivity (double p, double t)
           Compute refractivity (return value is n - 1).
    • double scan_ctl (int argc, char *argv[], const char *varname, int arridx, const char *defvalue, char *value)
          Search control parameter file for variable entry.

    void tangent point (los t *los, double *tpz, double *tplon, double *tplat)

           Find tangent point of a given LOS.
    • void time2jsec (int year, int mon, int day, int hour, int min, int sec, double remain, double *jsec)
          Convert date to seconds.

    void timer (const char *name, const char *file, const char *func, int line, int mode)

          Measure wall-clock time.

    void write_atm (const char *dirname, const char *filename, ctl_t *ctl, atm_t *atm)

           Write atmospheric data.
    • void write_matrix (const char *dirname, const char *filename, ctl_t *ctl, gsl_matrix *matrix, atm_t *atm,
       obs_t *obs, const char *rowspace, const char *colspace, const char *sort)
    • void write_obs (const char *dirname, const char *filename, ctl_t *ctl, obs_t *obs)
           Write observation data.

    void x2atm (ctl t *ctl, gsl vector *x, atm t *atm)

          Decompose parameter vector or state vector.

    void x2atm_help (atm_t *atm, double zmin, double zmax, double *value, gsl_vector *x, size_t *n)

           Extract elements from state vector.

    void y2obs (ctl_t *ctl, gsl_vector *y, obs_t *obs)

          Decompose measurement vector.
5.3.1 Detailed Description
JURASSIC library definitions.
Definition in file jurassic.c.
```

5.3.2 Function Documentation

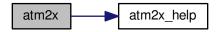
```
5.3.2.1 size_t atm2x ( ctl_t * ctl, atm_t * atm, gsl_vector * x, int * iqa, int * ipa )
```

Compose state vector or parameter vector.

Definition at line 29 of file jurassic.c.

```
00034
00035
00036
        int ig, iw;
00037
00038
        size_t n = 0;
00039
        /* Add pressure... */
00040
00041
        atm2x_help(atm, ctl->retp_zmin, ctl->retp_zmax,
00042
                   atm->p, IDXP, x, iqa, ipa, &n);
00043
       /* Add temperature... */
atm2x_help(atm, ctl->rett_zmin, ctl->rett_zmax,
00044
00045
00046
                   atm->t, IDXT, x, iqa, ipa, &n);
00047
00048
        /* Add volume mixing ratios...
00049
        for (ig = 0; ig < ctl->ng; ig++)
00050
         atm2x_help(atm, ctl->retq_zmin[ig], ctl->retq_zmax[ig],
00051
                     atm->q[ig], IDXQ(ig), x, iqa, ipa, &n);
00052
00053
        /* Add extinction... */
        for (iw = 0; iw < ctl->nw; iw++)
00055
         atm2x_help(atm, ctl->retk_zmin[iw], ctl->retk_zmax[iw],
00056
                     atm->k[iw], IDXK(iw), x, iqa, ipa, &n);
00057
00058
        return n;
00059 }
```

Here is the call graph for this function:



5.3.2.2 void atm2x_help (atm_t * atm, double zmin, double zmax, double * value, int val_iqa, gsl_vector * x, int * iqa, int * ipa, size_t * n)

Add elements to state vector.

Definition at line 63 of file jurassic.c.

```
00072
                        {
00073
00074
         int ip;
00075
00076
          /* Add elements to state vector... */
         for (ip = 0; ip < atm->np; ip++)
  if (atm->z[ip] >= zmin && atm->z[ip] <= zmax) {
    if (x != NULL)</pre>
00077
00078
00079
08000
                gsl_vector_set(x, *n, value[ip]);
00081
              if (iqa != NULL)
00082
                iqa[*n] = val_iqa;
              if (ipa != NULL)
00083
00084
                ipa[*n] = ip;
00085
              (*n)++;
00086
            }
00087 }
```

5.3.2.3 double brightness (double rad, double nu)

Compute brightness temperature.

Definition at line 91 of file jurassic.c.

```
00093 {
00094
00095 return C2 * nu / gsl_loglp(C1 * POW3(nu) / rad);
00096 }
```

5.3.2.4 void cart2geo (double * x, double * z, double * lon, double * lat)

Convert Cartesian coordinates to geolocation.

Definition at line 101 of file jurassic.c.

5.3.2.5 void climatology (ctl_t * ctl, atm_t * atm_mean)

Interpolate climatological data.

Definition at line 117 of file jurassic.c.

```
00119
00120
           static double z[121] = {
00121
             0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55,
00122
00123
00124
00125
              56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73,
00126
              74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91,
              92, 93, 94, 95, 96, 97, 98, 99, 100, 101, 102, 103, 104, 105, 106, 107,
00127
00128
             108, 109, 110, 111, 112, 113, 114, 115, 116, 117, 118, 119, 120
00129
00130
          static double pre[121] = {
  1017, 901.083, 796.45, 702.227, 617.614, 541.644, 473.437, 412.288,
  357.603, 308.96, 265.994, 228.348, 195.619, 167.351, 143.039, 122.198,
  104.369, 89.141, 76.1528, 65.0804, 55.641, 47.591, 40.7233, 34.8637,
00131
00132
00133
00134
              29.8633, 25.5956, 21.9534, 18.8445, 16.1909, 13.9258, 11.9913,
00135
              10.34, 8.92988, 7.72454, 6.6924, 5.80701, 5.04654, 4.39238, 3.82902,
00136
              3.34337, 2.92413, 2.56128, 2.2464, 1.97258, 1.73384, 1.52519, 1.34242, 1.18197, 1.04086, 0.916546, 0.806832, 0.709875, 0.624101, 0.548176,
00137
00138
              0.480974, 0.421507, 0.368904, 0.322408, 0.281386, 0.245249, 0.213465, 0.185549, 0.161072, 0.139644, 0.120913, 0.104568, 0.0903249, 0.0779269,
00139
00140
             0.0671493, 0.0577962, 0.0496902, 0.0426736, 0.0366093, 0.0313743, 0.0268598, 0.0229699, 0.0196206, 0.0167399, 0.0142646, 0.0121397,
00141
00142
              0.0103181, 0.00875775, 0.00742226, 0.00628076, 0.00530519, 0.00447183,
00143
              0.00376124, 0.00315632, 0.00264248, 0.00220738, 0.00184003, 0.00153095, 0.00127204, 0.00105608, 0.000876652, 0.00072798, 0.00060492,
00144
00145
             0.000503201, 0.000419226, 0.000349896, 0.000292659, 0.000245421, 0.000206394, 0.000174125, 0.000147441, 0.000125333, 0.000106985,
00146
00147
              9.173e-05, 7.90172e-05, 6.84172e-05, 5.95574e-05, 5.21183e-05,
00148
              4.58348e-05, 4.05127e-05, 3.59987e-05, 3.21583e-05, 2.88718e-05,
00149
00150
              2.60322e-05, 2.35687e-05, 2.14263e-05, 1.95489e-05
00151
00152
00153
           static double tem[121] = {
00154
             285.14, 279.34, 273.91, 268.3, 263.24, 256.55, 250.2, 242.82, 236.17,
00155
              229.87, 225.04, 221.19, 218.85, 217.19, 216.2, 215.68, 215.42, 215.55,
```

```
215.92, 216.4, 216.93, 217.45, 218, 218.68, 219.39, 220.25, 221.3,
                     222.41, 223.88, 225.42, 227.2, 229.52, 231.89, 234.51, 236.85, 239.42, 241.94, 244.57, 247.36, 250.32, 253.34, 255.82, 258.27, 260.39,
00157
00158
                     262.03, 263.45, 264.2, 264.78, 264.67, 264.38, 263.24, 262.03, 260.02,
00159
00160
                     258.09, 255.63, 253.28, 250.43, 247.81, 245.26, 242.77, 240.38, 237.94, 235.79, 233.53, 231.5, 229.53, 227.6, 225.62, 223.77, 222.06,
00161
                     220.33, 218.69, 217.18, 215.64, 214.13, 212.52, 210.86, 209.25,
00162
                     207.49, 205.81, 204.11, 202.22, 200.32, 198.39, 195.92, 193.46,
00163
                    190.94, 188.31, 185.82, 183.57, 181.43, 179.74, 178.64, 178.1, 178.25, 178.7, 179.41, 180.67, 182.31, 184.18, 186.6, 189.53, 192.66, 196.54, 201.13, 205.93, 211.73, 217.86, 225, 233.53, 242.57, 252.14, 261.48, 272.97, 285.26, 299.12, 312.2, 324.17, 338.34, 352.56, 365.28
00164
00165
00166
00167
00168
00169
00170
                static double c2h2[121] = {
                   1.352e-09, 2.83e-10, 1.269e-10, 6.926e-11, 4.346e-11, 2.909e-11,
00171
                    2.014e-11, 1.363e-11, 8.71e-12, 5.237e-12, 2.718e-12, 1.375e-12, 5.786e-13, 2.16e-13, 7.317e-14, 2.551e-14, 1.055e-14, 4.758e-15,
00172
00173
                     2.056e-15, 7.703e-16, 2.82e-16, 1.035e-16, 4.382e-17, 1.946e-17,
                     9.638e-18, 5.2e-18, 2.811e-18, 1.494e-18, 7.925e-19, 4.213e-19,
00175
00176
                     1.998e-19, 8.78e-20, 3.877e-20, 1.728e-20, 7.743e-21, 3.536e-21
00177
                     1.623e-21, 7.508e-22, 3.508e-22, 1.65e-22, 7.837e-23, 3.733e-23,
00178
                     1.808e-23, 8.77e-24, 4.285e-24, 2.095e-24, 1.032e-24, 5.082e-25,
                    00179
00180
                    00182
00183
00184
00185
                static double c2h6[121] = {
00186
                   2.667e-09, 2.02e-09, 1.658e-09, 1.404e-09, 1.234e-09, 1.109e-09,
00187
                     1.012e-09, 9.262e-10, 8.472e-10, 7.71e-10, 6.932e-10, 6.216e-10,
                     5.503e-10, 4.87e-10, 4.342e-10, 3.861e-10, 3.347e-10, 2.772e-10,
00188
00189
                     2.209e-10, 1.672e-10, 1.197e-10, 8.536e-11, 5.783e-11, 3.846e-11,
00190
                     2.495e-11, 1.592e-11, 1.017e-11, 6.327e-12, 3.895e-12, 2.403e-12,
                     1.416e-12, 8.101e-13, 4.649e-13, 2.686e-13, 1.557e-13, 9.14e-14,
00191
                     5.386e-14, 3.19e-14, 1.903e-14, 1.14e-14, 6.875e-15, 4.154e-15, 2.538e-15, 1.553e-15, 9.548e-16, 5.872e-16, 3.63e-16, 2.244e-16,
00192
00194
                     1.388e-16, 8.587e-17, 5.308e-17, 3.279e-17, 2.017e-17, 1.238e-17,
00195
                     7.542e-18, 4.585e-18, 2.776e-18, 1.671e-18, 9.985e-19, 5.937e-19,
00196
                     3.518e-19, 2.07e-19, 1.215e-19, 7.06e-20, 4.097e-20, 2.37e-20,
                     1.363e-20, 7.802e-21, 4.441e-21, 2.523e-21, 1.424e-21, 8.015e-22,
00197
                     4.497e-22, 2.505e-22, 1.391e-22, 7.691e-23, 4.238e-23, 2.331e-23,
00198
                     1.274e-23, 6.929e-24, 3.752e-24, 2.02e-24, 1.083e-24, 5.774e-25,
00199
                     3.041e-25, 1.593e-25, 8.308e-26, 4.299e-26, 2.195e-26, 1.112e-26,
00200
00201
                     00202
                   0, 0, 0, 0, 0, 0, 0, 0
00203
                };
00204
00205
                static double cc14[121] = {
                   1.075e-10, 1.075e-10, 1.075e-10, 1.075e-10, 1.075e-10, 1.075e-10,
                     1.075e-10, 1.075e-10, 1.075e-10, 1.06e-10, 1.024e-10, 9.69e-11,
00207
00208
                     8.93e-11, 8.078e-11, 7.213e-11, 6.307e-11, 5.383e-11, 4.49e-11,
                    3.609e-11, 2.705e-11, 1.935e-11, 1.385e-11, 8.35e-12, 5.485e-12, 3.853e-12, 2.22e-12, 5.875e-13, 3.445e-13, 1.015e-13, 6.075e-14, 4.383e-14, 2.692e-14, 1e-14, 1e-1
00209
00210
00211
                     1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14,
00213
00214
                     1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14,
00215
                     1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14,
00216
                     1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14,
                     le-14, le
00217
                     1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14,
00219
00220
                     1e-14, 1e-14, 1e-14
00221
00222
00223
                static double ch4[121] = {
                   1.864e-06, 1.835e-06, 1.819e-06, 1.805e-06, 1.796e-06, 1.788e-06,
00225
                     1.782e-06, 1.776e-06, 1.769e-06, 1.761e-06, 1.749e-06, 1.734e-06,
                     1.716e-06, 1.692e-06, 1.654e-06, 1.61e-06, 1.567e-06, 1.502e-06,
00226
00227
                     1.433e-06, 1.371e-06, 1.323e-06, 1.277e-06, 1.232e-06, 1.188e-06,
                    1.147e-06, 1.108e-06, 1.07e-06, 1.027e-06, 9.854e-07, 9.416e-07, 8.933e-07, 8.478e-07, 7.988e-07, 7.515e-07, 7.07e-07, 6.64e-07, 6.239e-07, 5.864e-07, 5.512e-07, 5.184e-07, 4.87e-07, 4.571e-07,
00228
00229
00230
                     3.032e-07, 4.04e-07, 3.802e-07, 3.578e-07, 3.383e-07, 3.203e-07, 3.032e-07, 2.889e-07, 2.76e-07, 2.635e-07, 2.519e-07, 2.409e-07,
00231
00232
                     2.302e-07, 2.219e-07, 2.144e-07, 2.071e-07, 1.999e-07, 1.93e-07, 1.862e-07, 1.795e-07, 1.731e-07, 1.668e-07, 1.607e-07, 1.548e-07,
00233
00234
                     1.49e-07, 1.434e-07, 1.38e-07, 1.328e-07, 1.277e-07, 1.227e-07, 1.18e-07, 1.134e-07, 1.089e-07, 1.046e-07, 1.004e-07, 9.635e-08,
00235
00236
                     9.245e-08, 8.867e-08, 8.502e-08, 8.15e-08, 7.809e-08, 7.48e-08,
                     7.159e-08, 6.849e-08, 6.55e-08, 6.262e-08, 5.98e-08, 5.708e-08,
00238
00239
                     5.448e-08, 5.194e-08, 4.951e-08, 4.72e-08, 4.5e-08, 4.291e-08,
00240
                     4.093e-08, 3.905e-08, 3.729e-08, 3.563e-08, 3.408e-08, 3.265e-08,
                     3.128e-08, 2.996e-08, 2.87e-08, 2.76e-08, 2.657e-08, 2.558e-08, 2.467e-08, 2.385e-08, 2.307e-08, 2.234e-08, 2.168e-08, 2.108e-08,
00241
00242
```

```
2.05e-08, 1.998e-08, 1.947e-08, 1.902e-08, 1.86e-08, 1.819e-08,
00244
00245
00246
00247
             static double clo[121] = {
                 7.419e-15, 1.061e-14, 1.518e-14, 2.195e-14, 3.175e-14, 4.666e-14,
00248
                 6.872e-14, 1.03e-13, 1.553e-13, 2.375e-13, 3.664e-13, 5.684e-13,
                 8.915e-13, 1.402e-12, 2.269e-12, 4.125e-12, 7.501e-12, 1.257e-11,
00250
00251
                 2.048e-11, 3.338e-11, 5.44e-11, 8.846e-11, 1.008e-10, 1.082e-10,
00252
                 1.157e-10, 1.232e-10, 1.312e-10, 1.539e-10, 1.822e-10, 2.118e-10,
                 2.387e-10, 2.687e-10, 2.875e-10, 3.031e-10, 3.23e-10, 3.648e-10,
00253
00254
                 4.117e-10, 4.477e-10, 4.633e-10, 4.794e-10, 4.95e-10, 5.104e-10,
                 5.259e-10, 5.062e-10, 4.742e-10, 4.443e-10, 4.051e-10, 3.659e-10,
00255
                 3.305e-10, 2.911e-10, 2.54e-10, 2.215e-10, 1.927e-10, 1.675e-10,
00256
00257
                 1.452e-10, 1.259e-10, 1.09e-10, 9.416e-11, 8.119e-11, 6.991e-11,
00258
                  6.015 e^{-11}, \ 5.163 e^{-11}, \ 4.43 e^{-11}, \ 3.789 e^{-11}, \ 3.24 e^{-11}, \ 2.769 e^{-11}, \\
00259
                 2.361e-11, 2.011e-11, 1.71e-11, 1.453e-11, 1.233e-11, 1.045e-11,
                8.851e-12, 7.48e-12, 6.316e-12, 5.326e-12, 4.487e-12, 3.778e-12, 3.176e-12, 2.665e-12, 2.234e-12, 1.87e-12, 1.563e-12, 1.304e-12,
00260
00262
                 1.085e-12, 9.007e-13, 7.468e-13, 6.179e-13, 5.092e-13, 4.188e-13,
                 3.442e-13, 2.816e-13, 2.304e-13, 1.885e-13, 1.542e-13,
00263
                 1.035e-13, 8.5e-14, 7.004e-14, 5.783e-14, 4.795e-14, 4.007e-14,
00264
                3.345e-14, 2.792e-14, 2.33e-14, 1.978e-14, 1.686e-14, 1.438e-14, 1.234e-14, 1.07e-14, 9.312e-15, 8.131e-15, 7.164e-15, 6.367e-15, 5.67e-15, 5.088e-15, 4.565e-15, 4.138e-15, 3.769e-15, 3.432e-15,
00265
00266
00267
00268
                 3.148e-15
00269
00270
00271
             static double clono2[121] = {
                1.011e-13, 1.515e-13, 2.272e-13, 3.446e-13, 5.231e-13, 8.085e-13, 1.253e-12, 1.979e-12, 3.149e-12, 5.092e-12, 8.312e-12, 1.366e-11, 2.272e-11, 3.791e-11, 6.209e-11, 9.101e-11, 1.334e-10, 1.951e-10,
00272
00273
00274
00275
                 2.853e-10, 3.94e-10, 4.771e-10, 5.771e-10, 6.675e-10, 7.665e-10,
00276
                 8.504e-10, 8.924e-10, 9.363e-10, 8.923e-10, 8.411e-10, 7.646e-10,
00277
                 6.525e-10, 5.576e-10, 4.398e-10, 3.403e-10, 2.612e-10, 1.915e-10,
                1.407e-10, 1.028e-10, 7.455e-11, 5.42e-11, 3.708e-11, 2.438e-11, 1.618e-11, 1.075e-11, 7.17e-12, 4.784e-12, 3.205e-12, 2.147e-12, 1.44e-12, 9.654e-13, 6.469e-13, 4.332e-13, 2.891e-13, 1.926e-13,
00278
00279
00281
                 1.274e-13, 8.422e-14, 5.547e-14, 3.636e-14, 2.368e-14, 1.536e-14,
                 9.937e-15, 6.39e-15, 4.101e-15, 2.61e-15, 1.659e-15, 1.052e-15,
00282
00283
                 6.638e-16, 4.172e-16, 2.61e-16, 1.63e-16, 1.013e-16, 6.275e-17,
                 3.879e-17, 2.383e-17, 1.461e-17, 8.918e-18, 5.43e-18, 3.301e-18,
00284
                 1.997e-18, 1.203e-18, 7.216e-19, 4.311e-19, 2.564e-19, 1.519e-19,
00285
00286
                 8.911e-20, 5.203e-20, 3.026e-20, 1.748e-20, 9.99e-21, 5.673e-21,
                 3.215e-21, 1.799e-21, 1.006e-21, 5.628e-22, 3.146e-22, 1.766e-22, 9.94e-23, 5.614e-23, 3.206e-23, 1.841e-23, 1.071e-23, 6.366e-24,
00288
00289
                 3.776e-24, 2.238e-24, 1.326e-24, 8.253e-25, 5.201e-25, 3.279e-25,
                2.108e-25, 1.395e-25, 9.326e-26, 6.299e-26, 4.365e-26, 3.104e-26, 2.219e-26, 1.621e-26, 1.185e-26, 8.92e-27, 6.804e-27, 5.191e-27,
00290
00291
00292
                 4.041e-27
00293
00294
00295
             static double co[121] = {
                1.907e-07, 1.553e-07, 1.362e-07, 1.216e-07, 1.114e-07, 1.036e-07, 9.737e-08, 9.152e-08, 8.559e-08, 7.966e-08, 7.277e-08, 6.615e-08,
00296
00297
                 5.884e-08, 5.22e-08, 4.699e-08, 4.284e-08, 3.776e-08, 3.274e-08,
00298
                 2.845e-08, 2.479e-08, 2.246e-08, 2.054e-08, 1.991e-08, 1.951e-08,
                 1.94e-08, 2.009e-08, 2.1e-08, 2.201e-08, 2.322e-08, 2.45e-08,
00300
00301
                 2.602e-08, 2.73e-08, 2.867e-08, 2.998e-08, 3.135e-08, 3.255e-08,
00302
                 3.352e-08, 3.426e-08, 3.484e-08, 3.53e-08, 3.593e-08, 3.671e-08,
00303
                 3.759e-08, 3.945e-08, 4.192e-08, 4.49e-08, 5.03e-08, 5.703e-08,
                 6.538e-08, 7.878e-08, 9.644e-08, 1.196e-07, 1.498e-07, 1.904e-07,
00304
00305
                 2.422e-07, 3.055e-07, 3.804e-07, 4.747e-07, 5.899e-07, 7.272e-07,
                 8.91e-07, 1.071e-06, 1.296e-06, 1.546e-06, 1.823e-06, 2.135e-06, 2.44e-06, 2.714e-06, 2.967e-06, 3.189e-06, 3.391e-06, 3.58e-06,
00306
00307
00308
                 3.773e-06, 4.022e-06, 4.346e-06, 4.749e-06, 5.199e-06, 5.668e-06,
00309
                 6.157e-06, 6.688e-06, 7.254e-06, 7.867e-06, 8.539e-06, 9.26e-06,
                 1.009e-05, 1.119e-05, 1.228e-05, 1.365e-05, 1.506e-05, 1.641e-05,
00310
00311
                 1.784e-05, 1.952e-05, 2.132e-05, 2.323e-05, 2.531e-05, 2.754e-05,
                 3.047e-05, 3.459e-05, 3.922e-05, 4.439e-05, 4.825e-05, 5.077e-05,
00313
                 5.34e-05, 5.618e-05, 5.909e-05, 6.207e-05, 6.519e-05, 6.845e-05,
                 6.819e-05, 6.726e-05, 6.622e-05, 6.512e-05, 6.671e-05, 6.862e-05, 7.048e-05, 7.264e-05, 7.3e-05, 7.2e-05, 7.2e-
00314
00315
00316
00317
00318
             static double cof2[121] = {
                 7.5e-14, 1.055e-13, 1.485e-13, 2.111e-13, 3.001e-13, 4.333e-13,
00319
                 6.269e-13, 9.221e-13, 1.364e-12, 2.046e-12, 3.093e-12, 4.703e-12, 7.225e-12, 1.113e-11, 1.66e-11, 2.088e-11, 2.626e-11, 3.433e-11,
00320
00321
                 4.549e-11, 5.886e-11, 7.21e-11, 8.824e-11, 1.015e-10, 1.155e-10, 1.288e-10, 1.388e-10, 1.497e-10, 1.554e-10, 1.606e-10, 1.639e-10,
00322
00323
                 1.64e-10, 1.64e-10, 1.596e-10, 1.542e-10, 1.482e-10, 1.382e-10,
                 1.289e-10, 1.198e-10, 1.109e-10, 1.026e-10, 9.484e-11, 8.75e-11, 8.086e-11, 7.49e-11, 6.948e-11, 6.446e-11, 5.961e-11, 5.505e-11,
00325
00326
00327
                 5.085 e^{-11},\ 4.586 e^{-11},\ 4.1 e^{-11},\ 3.665 e^{-11},\ 3.235 e^{-11},\ 2.842 e^{-11},
                 2.491e-11, 2.11e-11, 1.769e-11, 1.479e-11, 1.197e-11, 9.631e-12, 7.74e-12, 6.201e-12, 4.963e-12, 3.956e-12, 3.151e-12, 2.507e-12,
00328
00329
```

```
1.99e-12, 1.576e-12, 1.245e-12, 9.83e-13, 7.742e-13, 6.088e-13,
                        4.782e-13, 3.745e-13, 2.929e-13, 2.286e-13, 1.782e-13, 1.388e-13, 1.079e-13, 8.362e-14, 6.471e-14, 4.996e-14, 3.85e-14, 2.96e-14,
00331
00332
00333
                         2.265e-14, 1.729e-14, 1.317e-14, 9.998e-15, 7.549e-15, 5.683e-15,
                         4.273e-15, 3.193e-15, 2.385e-15, 1.782e-15, 1.331e-15, 9.957e-16,
00334
                         7.461e-16, 5.601e-16, 4.228e-16, 3.201e-16, 2.438e-16, 1.878e-16,
00335
                         1.445e-16, 1.111e-16, 8.544e-17, 6.734e-17, 5.341e-17, 4.237e-17,
                         3.394e-17, 2.759e-17, 2.254e-17, 1.851e-17, 1.54e-17, 1.297e-17,
00337
00338
                         1.096e-17, 9.365e-18, 8e-18, 6.938e-18, 6.056e-18, 5.287e-18,
00339
                         4.662e-18
00340
00341
00342
                   static double f11[121] = {
                      2.65e-10, 2.65e-10, 2.65e-10, 2.65e-10, 2.65e-10, 2.65e-10, 2.65e-10,
00343
00344
                         2.65e-10, 2.65e-10, 2.65e-10, 2.65e-10, 2.65e-10, 2.635e-10, 2.536e-10,
00345
                         2.44e-10, 2.348e-10, 2.258e-10, 2.153e-10, 2.046e-10, 1.929e-10,
                         1.782e-10, 1.648e-10, 1.463e-10, 1.291e-10, 1.1e-10, 8.874e-11,
00346
                         7.165e-11, 5.201e-11, 3.744e-11, 2.577e-11, 1.64e-11, 1.048e-11,
00347
                         5.993e-12, 3.345e-12, 1.839e-12, 9.264e-13, 4.688e-13, 2.329e-13,
00349
                         1.129e-13, 5.505e-14, 2.825e-14, 1.492e-14, 7.997e-15, 5.384e-15,
00350
                         3.988e-15, 2.955e-15, 2.196e-15, 1.632e-15, 1.214e-15, 9.025e-16,
00351
                         6.708e-16, 4.984e-16, 3.693e-16, 2.733e-16, 2.013e-16, 1.481e-16,
                        1.087e-16, 7.945e-17, 5.782e-17, 4.195e-17, 3.038e-17, 2.19e-17, 1.577e-17, 1.128e-17, 8.063e-18, 5.753e-18, 4.09e-18, 2.899e-18, 2.048e-18, 1.444e-18, 1.015e-18, 7.12e-19, 4.985e-19, 3.474e-19,
00352
00353
00354
                         2.417e-19, 1.677e-19, 1.161e-19, 8.029e-20, 5.533e-20, 3.799e-20,
00355
                         2.602e-20, 1.776e-20, 1.209e-20, 8.202e-21, 5.522e-21, 3.707e-21,
00356
00357
                         2.48e-21, 1.652e-21, 1.091e-21, 7.174e-22, 4.709e-22, 3.063e-22,
00358
                         1.991e-22, 1.294e-22, 8.412e-23, 5.483e-23, 3.581e-23, 2.345e-23,
00359
                         1.548e-23, 1.027e-23, 6.869e-24, 4.673e-24, 3.173e-24, 2.153e-24,
00360
                         1.461e-24, 1.028e-24, 7.302e-25, 5.188e-25, 3.739e-25, 2.753e-25,
00361
                         2.043e-25, 1.528e-25, 1.164e-25, 9.041e-26, 7.051e-26, 5.587e-26,
                         4.428e-26, 3.588e-26, 2.936e-26, 2.402e-26, 1.995e-26
00362
00363
00364
                   static double f12[121] = {
00365
                       5.45e-10, 5.45e-
00366
00368
                         5.155e-10, 5.022e-10, 4.893e-10, 4.772e-10, 4.655e-10, 4.497e-10,
00369
                         4.249e-10, 4.015e-10, 3.632e-10, 3.261e-10, 2.858e-10,
                                                                                                                                                                  2.408e-10
00370
                         2.03e-10, 1.685e-10, 1.4e-10, 1.163e-10, 9.65e-11, 8.02e-11, 6.705e-11,
                        5.624e-11, 4.764e-11, 4.249e-11, 3.792e-11, 3.315e-11, 2.819e-11, 2.4e-11, 1.999e-11, 1.64e-11, 1.352e-11, 1.14e-11, 9.714e-12,
00371
00372
                         8.28e-12, 7.176e-12, 6.251e-12, 5.446e-12, 4.72e-12, 4.081e-12,
00373
                         3.528e-12, 3.08e-12, 2.699e-12, 2.359e-12, 2.111e-12, 1.901e-12,
00374
00375
                         1.709e-12, 1.534e-12, 1.376e-12, 1.233e-12, 1.103e-12, 9.869e-13,
00376
                         8.808e-13, 7.859e-13, 7.008e-13, 6.241e-13, 5.553e-13, 4.935e-13,
                         4.383e-13, 3.889e-13, 3.447e-13, 3.054e-13, 2.702e-13, 2.389e-13, 2.11e-13, 1.862e-13, 1.643e-13, 1.448e-13, 1.274e-13, 1.121e-13, 9.844e-14, 8.638e-14, 7.572e-14, 6.62e-14, 5.782e-14, 5.045e-14,
00377
00378
00379
                         4.394e-14, 3.817e-14, 3.311e-14, 2.87e-14, 2.48e-14, 2.142e-14,
                         1.851e-14, 1.599e-14, 1.383e-14, 1.196e-14, 1.036e-14, 9e-15,
00381
00382
                         7.828e-15, 6.829e-15, 5.992e-15, 5.254e-15, 4.606e-15, 4.037e-15,
                        3.583e-15, 3.19e-15, 2.841e-15, 2.542e-15, 2.291e-15, 2.07e-15, 1.875e-15, 1.71e-15, 1.57e-15, 1.442e-15, 1.333e-15, 1.232e-15,
00383
00384
00385
                         1.147e-15, 1.071e-15, 1.001e-15, 9.396e-16
00387
00388
                   static double f14[121] = {
                        9e-11, 8.73e-11, 8.46e-11, 8.19e-11, 7.92e-11, 7.74e-11, 7.65e-11, 7
00389
00390
00391
00392
                         7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11,
                         7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-
00393
00394
00395
                         7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11,
                         7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11,
                                                                                                                                                                                7.65e-11,
00396
                         7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11,
00397
                                                                                                                                                                                 7.65e-11,
00398
                         7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11,
                         7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11,
00399
00400
                         7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11,
                                                                                                                                                                                7.65e-11,
                                                                                                                                                                                7.65e-11,
00401
                         7.65e-11, 7.65e-11, 7.65e-11,
                                                                                                    7.65e-11, 7.65e-11, 7.65e-11,
00402
                         7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11,
                         7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11
00403
00404
00405
00406
00407
00408
                    static double f22[121] = {
                        1.4e-10, 1.4e-10, 1.4e-10, 1.4e-10, 1.4e-10, 1.4e-10, 1.4e-10,
00409
                         1.4e-10, 1.4e-10, 1.4e-10, 1.372e-10, 1.317e-10, 1.235e-10, 1.153e-10,
00410
                         1.075e-10, 1.002e-10, 9.332e-11, 8.738e-11, 8.194e-11, 7.7e-11,
                         7.165e-11, 6.753e-11, 6.341e-11, 5.971e-11, 5.6e-11, 5.229e-11, 4.859e-11, 4.488e-11, 4.118e-11, 3.83e-11, 3.568e-11, 3.308e-11,
00412
00413
00414
                         3.047e-11,\ 2.82e-11,\ 2.594e-11,\ 2.409e-11,\ 2.237e-11,\ 2.065e-11,
                         1.894e-11, 1.771e-11, 1.647e-11, 1.532e-11, 1.416e-11, 1.332e-11, 1.246e-11, 1.161e-11, 1.087e-11, 1.017e-11, 9.471e-12, 8.853e-12,
00415
00416
```

```
8.235e-12, 7.741e-12, 7.247e-12, 6.836e-12, 6.506e-12, 6.176e-12,
            5.913e-12, 5.65e-12, 5.419e-12, 5.221e-12, 5.024e-12, 4.859e-12,
00418
00419
            4.694e-12, 4.546e-12, 4.414e-12, 4.282e-12, 4.15e-12, 4.019e-12,
            3.903e-12, 3.805e-12, 3.706e-12, 3.607e-12, 3.508e-12, 3.41e-12, 3.31e-12, 3.212e-12, 3.129e-12, 3.047e-12, 2.964e-12, 2.882e-12, 2.8e-12, 2.734e-12, 2.668e-12, 2.602e-12, 2.537e-12, 2.471e-12,
00420
00421
00422
            2.421e-12, 2.372e-12, 2.322e-12, 2.273e-12, 2.224e-12, 2.182e-12,
            2.141e-12, 2.1e-12, 2.059e-12, 2.018e-12, 1.977e-12, 1.935e-12,
00424
00425
            1.894e-12, 1.853e-12, 1.812e-12, 1.77e-12, 1.73e-12, 1.688e-12,
00426
            1.647e-12, 1.606e-12, 1.565e-12, 1.524e-12, 1.483e-12, 1.441e-12,
           1.4e-12, 1.359e-12, 1.317e-12, 1.276e-12, 1.235e-12, 1.194e-12, 1.153e-12, 1.112e-12, 1.071e-12, 1.029e-12, 9.883e-13
00427
00428
00429
00430
00431
         static double h2o[121] = {
           0.01166, 0.008269, 0.005742, 0.003845, 0.00277, 0.001897, 0.001272, 0.000827, 0.000539, 0.0003469, 0.0001579, 3.134e-05, 1.341e-05,
00432
00433
            6.764e-06, 4.498e-06, 3.703e-06, 3.724e-06, 3.899e-06, 4.002e-06,
00434
            4.122e-06, 4.277e-06, 4.438e-06, 4.558e-06, 4.673e-06, 4.763e-06,
00436
            4.809e-06, 4.856e-06, 4.936e-06, 5.021e-06, 5.114e-06, 5.222e-06,
            5.331e-06, 5.414e-06, 5.488e-06, 5.563e-06, 5.633e-06, 5.704e-06,
00437
00438
            5.767e-06, 5.819e-06, 5.872e-06, 5.914e-06, 5.949e-06, 5.984e-06,
00439
            6.015e-06, 6.044e-06, 6.073e-06, 6.104e-06, 6.136e-06, 6.167e-06,
           6.189e-06, 6.208e-06, 6.226e-06, 6.212e-06, 6.185e-06, 6.158e-06, 6.114e-06, 6.066e-06, 6.018e-06, 5.877e-06, 5.728e-06, 5.582e-06,
00440
00441
            5.437e-06, 5.296e-06, 5.156e-06, 5.02e-06, 4.886e-06, 4.754e-06,
            4.625e-06, 4.498e-06, 4.374e-06, 4.242e-06, 4.096e-06, 3.955e-06,
00443
00444
            3.817e-06, 3.683e-06, 3.491e-06, 3.204e-06, 2.94e-06, 2.696e-06,
           2.47e-06, 2.252e-06, 2.019e-06, 1.808e-06, 1.618e-06, 1.445e-06, 1.285e-06, 1.105e-06, 9.489e-07, 8.121e-07, 6.938e-07, 5.924e-07,
00445
00446
            5.04e-07, 4.288e-07, 3.648e-07, 3.103e-07, 2.642e-07, 2.252e-07,
00447
00448
            1.921e-07, 1.643e-07, 1.408e-07, 1.211e-07, 1.048e-07, 9.063e-08,
            7.835e-08, 6.774e-08, 5.936e-08, 5.221e-08, 4.592e-08, 4.061e-08,
00449
00450
            3.62e-08, 3.236e-08, 2.902e-08, 2.62e-08, 2.383e-08, 2.171e-08,
00451
           1.989e-08, 1.823e-08, 1.684e-08, 1.562e-08, 1.449e-08, 1.351e-08
00452
00453
         static double h2o2[121] =
00455
           1.779e-10, 7.938e-10, 8.953e-10, 8.032e-10, 6.564e-10, 5.159e-10,
            4.003e-10, 3.026e-10, 2.222e-10, 1.58e-10, 1.044e-10, 6.605e-11,
00456
00457
           3.413e-11, 1.453e-11, 1.062e-11, 1.009e-11, 9.597e-12, 1.175e-11,
           1.572e-11, 2.091e-11, 2.746e-11, 3.603e-11, 4.791e-11, 6.387e-11, 8.239e-11, 1.007e-10, 1.23e-10, 1.363e-10, 1.489e-10, 1.585e-10, 1.608e-10, 1.632e-10, 1.576e-10, 1.502e-10, 1.423e-10, 1.302e-10,
00458
00459
00460
            1.192e-10, 1.085e-10, 9.795e-11, 8.854e-11, 8.057e-11, 7.36e-11, 6.736e-11, 6.362e-11, 6.087e-11, 5.825e-11, 5.623e-11, 5.443e-11,
00462
00463
           5.27e-11, 5.098e-11, 4.931e-11, 4.769e-11, 4.611e-11, 4.458e-11,
00464
            4.308e-11, 4.102e-11, 3.887e-11, 3.682e-11, 3.521e-11, 3.369e-11,
            3.224e-11, 3.082e-11, 2.946e-11, 2.814e-11, 2.687e-11, 2.566e-11,
00465
            2.449e-11, 2.336e-11, 2.227e-11, 2.123e-11, 2.023e-11, 1.927e-11,
00466
            1.835e-11, 1.746e-11, 1.661e-11, 1.58e-11, 1.502e-11, 1.428e-11,
            1.357e-11, 1.289e-11, 1.224e-11, 1.161e-11, 1.102e-11, 1.045e-11,
00468
00469
            9.895e-12, 9.369e-12, 8.866e-12, 8.386e-12, 7.922e-12, 7.479e-12,
           7.06e-12, 6.656e-12, 6.274e-12, 5.914e-12, 5.575e-12, 5.257e-12, 4.959e-12, 4.679e-12, 4.42e-12, 4.178e-12, 3.954e-12, 3.75e-12,
00470
00471
            3.557e-12, 3.372e-12, 3.198e-12, 3.047e-12, 2.908e-12, 2.775e-12, 2.653e-12, 2.544e-12, 2.442e-12, 2.346e-12, 2.26e-12, 2.183e-12,
00472
            2.11e-12, 2.044e-12, 1.98e-12, 1.924e-12, 1.871e-12, 1.821e-12,
00474
00475
           1.775e-12
00476
00477
00478
         static double hcn[121] = {
00479
           5.5e-10, 5.5e-10, 5.5e-10, 5.5e-10, 5.5e-10, 5.5e-10,
            5.5e-10, 5.5e-10, 5.5e-10, 5.5e-10, 5.498e-10, 5.495e-10, 5.493e-10,
00480
00481
            5.49e-10, 5.488e-10, 4.717e-10, 3.946e-10, 3.174e-10, 2.4e-10,
00482
           1.626e-10, 1.619e-10, 1.612e-10, 1.602e-10, 1.593e-10, 1.582e-10,
00483
            1.572e-10, 1.56e-10, 1.549e-10, 1.539e-10, 1.53e-10, 1.519e-10,
            1.506e-10, 1.487e-10, 1.467e-10, 1.449e-10, 1.43e-10, 1.413e-10,
00484
            1.397e-10, 1.382e-10, 1.368e-10, 1.354e-10, 1.337e-10, 1.315e-10,
00485
            1.292e-10, 1.267e-10, 1.241e-10, 1.215e-10, 1.19e-10, 1.165e-10,
            1.141e-10, 1.118e-10, 1.096e-10, 1.072e-10, 1.047e-10, 1.021e-10,
00487
00488
            9.968e-11, 9.739e-11, 9.539e-11, 9.339e-11, 9.135e-11, 8.898e-11,
00489
            8.664e-11, 8.439e-11, 8.249e-11, 8.075e-11, 7.904e-11, 7.735e-11,
            7.565e-11, 7.399e-11, 7.245e-11, 7.109e-11, 6.982e-11, 6.863e-11,
00490
            6.755e-11, 6.657e-11, 6.587e-11, 6.527e-11, 6.476e-11, 6.428e-11,
00491
            6.382e-11, 6.343e-11, 6.307e-11, 6.272e-11, 6.238e-11, 6.205e-11,
00492
            6.17e-11, 6.137e-11, 6.102e-11, 6.072e-11, 6.046e-11, 6.03e-11,
00493
00494
            6.018e-11, 6.01e-11, 6.001e-11, 5.992e-11, 5.984e-11, 5.975e-11,
00495
            5.967e-11, 5.958e-11, 5.95e-11, 5.941e-11, 5.933e-11, 5.925e-11,
           5.916e-11, 5.908e-11, 5.899e-11, 5.891e-11, 5.883e-11, 5.874e-11, 5.866e-11, 5.858e-11, 5.85e-11, 5.841e-11, 5.833e-11, 5.825e-11, 5.817e-11, 5.808e-11, 5.8e-11, 5.792e-11, 5.784e-11
00496
00497
00498
00499
00500
00501
         static double hno3[121] = {
           1.809e-10, 7.234e-10, 5.899e-10, 4.342e-10, 3.277e-10, 2.661e-10, 2.35e-10, 2.267e-10, 2.389e-10, 2.651e-10, 3.255e-10, 4.099e-10,
00502
00503
```

```
5.42e-10, 6.978e-10, 8.807e-10, 1.112e-09, 1.405e-09, 2.04e-09,
            3.111e-09, 4.5e-09, 5.762e-09, 7.37e-09, 7.852e-09, 8.109e-09,
00505
           8.067e-09, 7.554e-09, 7.076e-09, 6.268e-09, 5.524e-09, 4.749e-09,
00506
00507
           3.909e-09, 3.223e-09, 2.517e-09, 1.942e-09, 1.493e-09, 1.122e-09,
           8.449e-10, 6.361e-10, 4.787e-10, 3.611e-10, 2.804e-10, 2.215e-10,
00508
00509
            1.758e-10, 1.441e-10, 1.197e-10, 9.953e-11, 8.505e-11, 7.334e-11,
            6.325e-11, 5.625e-11, 5.058e-11, 4.548e-11, 4.122e-11, 3.748e-11,
           3.402e-11, 3.088e-11, 2.8e-11, 2.536e-11, 2.293e-11, 2.072e-11,
00511
00512
           1.871e-11, 1.687e-11, 1.52e-11, 1.368e-11, 1.23e-11, 1.105e-11,
00513
           9.922e-12, 8.898e-12, 7.972e-12, 7.139e-12, 6.385e-12, 5.708e-12,
           5.099e-12, 4.549e-12, 4.056e-12, 3.613e-12, 3.216e-12, 2.862e-12,
00514
00515
           2.544e-12, 2.259e-12, 2.004e-12, 1.776e-12, 1.572e-12, 1.391e-12,
            1.227e-12, 1.082e-12, 9.528e-13, 8.379e-13, 7.349e-13, 6.436e-13,
            5.634e-13, 4.917e-13, 4.291e-13, 3.745e-13, 3.267e-13, 2.854e-13,
00517
00518
           2.494e-13, 2.181e-13, 1.913e-13, 1.68e-13, 1.479e-13, 1.31e-13,
           1.159e-13, 1.025e-13, 9.067e-14, 8.113e-14, 7.281e-14, 6.535e-14, 5.892e-14, 5.348e-14, 4.867e-14, 4.439e-14, 4.073e-14, 3.76e-14,
00519
00520
           3.476e-14, 3.229e-14, 3e-14, 2.807e-14, 2.635e-14, 2.473e-14,
00521
           2.332e-14
00523
00524
00525
         static double hno4[121] = {
           6.118e-12, 3.594e-12, 2.807e-12, 3.04e-12, 4.458e-12, 7.986e-12,
00526
           1.509e-11, 2.661e-11, 3.738e-11, 4.652e-11, 4.429e-11, 3.992e-11, 3.347e-11, 3.005e-11, 3.173e-11, 4.055e-11, 5.812e-11, 8.489e-11,
00527
00528
            1.19e-10, 1.482e-10, 1.766e-10, 2.103e-10, 2.35e-10, 2.598e-10,
00529
00530
           2.801e-10, 2.899e-10, 3e-10, 2.817e-10, 2.617e-10, 2.332e-10,
00531
           1.933e-10, 1.605e-10, 1.232e-10, 9.285e-11, 6.941e-11, 4.951e-11,
           3.539e-11, 2.402e-11, 1.522e-11, 9.676e-12, 6.056e-12, 3.745e-12, 2.34e-12, 1.463e-12, 9.186e-13, 5.769e-13, 3.322e-13, 1.853e-13, 1.035e-13, 7.173e-14, 5.382e-14, 4.036e-14, 3.401e-14, 2.997e-14,
00532
00533
00534
            2.635e-14, 2.316e-14, 2.034e-14, 1.783e-14, 1.56e-14, 1.363e-14,
00536
            1.19e-14, 1.037e-14, 9.032e-15, 7.846e-15, 6.813e-15, 5.912e-15,
00537
           5.121e-15, 4.431e-15, 3.829e-15, 3.306e-15, 2.851e-15, 2.456e-15,
00538
           2.114e-15, 1.816e-15, 1.559e-15, 1.337e-15, 1.146e-15, 9.811e-16,
           8.389e-16, 7.162e-16, 6.109e-16, 5.203e-16, 4.425e-16, 3.76e-16,
00539
           3.184e-16, 2.692e-16, 2.274e-16, 1.917e-16, 1.61e-16, 1.35e-16, 1.131e-16, 9.437e-17, 7.874e-17, 6.57e-17, 5.481e-17, 4.579e-17,
00540
00542
            3.828e-17, 3.204e-17, 2.691e-17, 2.264e-17, 1.912e-17, 1.626e-17,
00543
            1.382e-17, 1.174e-17, 9.972e-18, 8.603e-18, 7.45e-18, 6.453e-18,
00544
           5.623e-18, 4.944e-18, 4.361e-18, 3.859e-18, 3.443e-18, 3.096e-18,
00545
           2.788e-18, 2.528e-18, 2.293e-18, 2.099e-18, 1.929e-18, 1.773e-18,
00546
           1.64e-18
00547
00548
00549
         static double hocl[121] = {
00550
          1.056e-12, 1.194e-12, 1.35e-12, 1.531e-12, 1.737e-12, 1.982e-12,
           2.263e-12, 2.599e-12, 2.991e-12, 3.459e-12, 4.012e-12, 4.662e-12, 5.438e-12, 6.35e-12, 7.425e-12, 8.686e-12, 1.016e-11, 1.188e-11, 1.389e-11, 1.659e-11, 2.087e-11, 2.621e-11, 3.265e-11, 4.064e-11, 4.859e-11, 5.441e-11, 6.09e-11, 6.373e-11, 6.611e-11, 6.94e-11,
00551
00552
00553
00555
           7.44e-11, 7.97e-11, 8.775e-11, 9.722e-11, 1.064e-10, 1.089e-10
00556
           1.114e-10, 1.106e-10, 1.053e-10, 1.004e-10, 9.006e-11, 7.778e-11,
00557
            6.739 e-11, \ 5.636 e-11, \ 4.655 e-11, \ 3.845 e-11, \ 3.042 e-11, \ 2.368 e-11,
00558
           1.845e-11, 1.442e-11, 1.127e-11, 8.814e-12, 6.544e-12, 4.763e-12,
           3.449e-12, 2.612e-12, 1.999e-12, 1.526e-12, 1.16e-12, 8.793e-13,
00559
            6.655e-13, 5.017e-13, 3.778e-13, 2.829e-13, 2.117e-13, 1.582e-13,
           1.178e-13, 8.755e-14, 6.486e-14, 4.799e-14, 3.54e-14, 2.606e-14,
00561
00562
           1.916e-14, 1.403e-14, 1.026e-14, 7.48e-15, 5.446e-15, 3.961e-15,
00563
           2.872e-15, 2.076e-15, 1.498e-15, 1.077e-15, 7.726e-16, 5.528e-16,
           3.929e-16, 2.785e-16, 1.969e-16, 1.386e-16, 9.69e-17, 6.747e-17,
00564
00565
            4.692e-17, 3.236e-17, 2.232e-17, 1.539e-17, 1.061e-17, 7.332e-18,
            5.076e-18, 3.522e-18, 2.461e-18, 1.726e-18, 1.22e-18, 8.75e-19,
           6.264e-19, 4.482e-19, 3.207e-19, 2.368e-19, 1.762e-19, 1.312e-19, 9.891e-20, 7.595e-20, 5.87e-20, 4.567e-20, 3.612e-20, 2.904e-20,
00567
00568
00569
           2.343e-20, 1.917e-20, 1.568e-20, 1.308e-20, 1.1e-20, 9.25e-21,
00570
           7.881e-21
00571
00572
         static double n2o[121] =
          3.17e-07, 3.03e-07,
00574
00575
           2.984e-07, 2.938e-07, 2.892e-07, 2.847e-07, 2.779e-07, 2.705e-07, 2.631e-07, 2.557e-07, 2.484e-07, 2.345e-07, 2.201e-07, 2.01e-07,
00576
00577
00578
            1.754e-07, 1.532e-07, 1.329e-07, 1.154e-07, 1.003e-07, 8.735e-08,
            7.617e-08, 6.512e-08, 5.547e-08, 4.709e-08, 3.915e-08, 3.259e-08,
00580
           2.738e-08, 2.327e-08, 1.98e-08, 1.711e-08, 1.493e-08, 1.306e-08,
00581
           1.165e-08, 1.049e-08, 9.439e-09, 8.375e-09, 7.391e-09, 6.525e-09,
00582
           5.759e-09, 5.083e-09, 4.485e-09, 3.953e-09, 3.601e-09, 3.27e-09,
           2.975e-09, 2.757e-09, 2.556e-09, 2.37e-09, 2.195e-09, 2.032e-09, 1.912e-09, 1.79e-09, 1.679e-09, 1.572e-09, 1.482e-09, 1.402e-09,
00583
00584
            1.326e-09, 1.254e-09, 1.187e-09, 1.127e-09, 1.071e-09, 1.02e-09,
            9.673e-10, 9.193e-10, 8.752e-10, 8.379e-10, 8.017e-10, 7.66e-10,
00586
00587
            7.319e-10, 7.004e-10, 6.721e-10, 6.459e-10, 6.199e-10, 5.942e-10,
00588
           5.703e-10, 5.488e-10, 5.283e-10, 5.082e-10, 4.877e-10, 4.696e-10,
           4.52e-10, 4.355e-10, 4.198e-10, 4.039e-10, 3.888e-10, 3.754e-10, 3.624e-10, 3.499e-10, 3.381e-10, 3.267e-10, 3.163e-10, 3.058e-10,
00589
00590
```

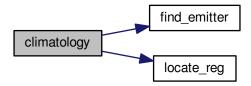
```
2.959e-10, 2.864e-10, 2.77e-10, 2.686e-10, 2.604e-10, 2.534e-10,
                               2.462e-10, 2.386e-10, 2.318e-10, 2.247e-10, 2.189e-10, 2.133e-10, 2.071e-10, 2.014e-10, 1.955e-10, 1.908e-10, 1.86e-10, 1.817e-10
00592
00593
00594
00595
00596
                         static double n2o5[121] = {
                            1.231e-11, 3.035e-12, 1.702e-12, 9.877e-13, 8.081e-13, 9.039e-13,
                                 1.169e-12, 1.474e-12, 1.651e-12, 1.795e-12, 1.998e-12, 2.543e-12,
00598
00599
                                 4.398e-12, 7.698e-12, 1.28e-11, 2.131e-11, 3.548e-11, 5.894e-11,
                               7.645e-11, 1.089e-10, 1.391e-10, 1.886e-10, 2.386e-10, 2.986e-10, 3.487e-10, 3.994e-10, 4.5e-10, 4.6e-10, 4.591e-10, 4.1e-10, 3.488e-10,
00600
00601
                                 2.846e-10, 2.287e-10, 1.696e-10, 1.011e-10, 6.428e-11, 4.324e-11,
00602
                                 2.225e-11, 6.214e-12, 3.608e-12, 8.793e-13, 4.491e-13, 1.04e-13,
00603
                                 6.1e-14, 3.436e-14, 6.671e-15, 1.171e-15, 5.848e-16, 1.212e-16,
00604
00605
                                 le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16,
00606
                                 1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16,
00607
                                 1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16,
00608
                                 1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16,
                                 1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16,
00610
                                 le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16,
                                 1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16,
00611
00612
                                1e-16, 1e-16
00613
00614
                         static double nh3[121] = {
00615
                             1e-10, 1e-10, 1e-10, 1e-10, 1e-10, 1e-10, 1e-10, 1e-10, 1e-10, 1e-10,
                                 1e-10, 1e-10, 1e-10, 1e-10, 9.444e-11, 8.488e-11, 7.241e-11, 5.785e-11,
00617
00618
                                 4.178e-11, 3.018e-11, 2.18e-11, 1.574e-11, 1.137e-11, 8.211e-12,
00619
                               5.973e-12, 4.327e-12, 3.118e-12, 2.234e-12, 1.573e-12, 1.04e-12,
00620
                                 6.762e-13, 4.202e-13, 2.406e-13, 1.335e-13, 6.938e-14, 3.105e-14,
                               1.609e-14, 1.033e-14, 6.432e-15, 4.031e-15, 2.555e-15, 1.656e-15, 1.15e-15, 7.904e-16, 5.63e-16, 4.048e-16, 2.876e-16, 2.004e-16,
00621
00622
                                 1.356e-16, 9.237e-17, 6.235e-17, 4.223e-17, 3.009e-17, 2.328e-17,
00623
00624
                                 2.002e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17,
                                1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 
00625
00626
                                1.914e-17, 
00627
00629
                                 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17,
                               1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 
00630
00631
                                1.914e-17, 
00632
00633
00634
                                 1.914e-17
00636
00637
00638
                         static double no[121] = {
                               2.586e-10, 4.143e-11, 1.566e-11, 9.591e-12, 8.088e-12, 8.462e-12,
00639
                                 1.013e-11, 1.328e-11, 1.855e-11, 2.678e-11, 3.926e-11, 5.464e-11,
00640
                                 7.012e-11, 8.912e-11, 1.127e-10, 1.347e-10, 1.498e-10, 1.544e-10,
                               1.602e-10, 1.824e-10, 2.078e-10, 2.366e-10, 2.691e-10, 5.141e-10,
00642
00643
                                8.259e-10, 1.254e-09, 1.849e-09, 2.473e-09, 3.294e-09, 4.16e-09,
                               5.095e-09, 6.11e-09, 6.93e-09, 7.888e-09, 8.903e-09, 9.713e-09, 1.052e-08, 1.115e-08, 1.173e-08, 1.21e-08, 1.228e-08, 1.239e-08,
00644
00645
                                 1.231e-08, 1.213e-08, 1.192e-08, 1.138e-08, 1.085e-08, 1.008e-08,
00646
                                 9.224e-09, 8.389e-09, 7.262e-09, 6.278e-09, 5.335e-09, 4.388e-09,
                                 3.589e-09, 2.761e-09, 2.129e-09, 1.633e-09, 1.243e-09, 9.681e-10,
00648
                                8.355e-10, 7.665e-10, 7.442e-10, 8.584e-10, 9.732e-10, 1.063e-09,
00649
00650
                                1.163e-09, 1.286e-09, 1.472e-09, 1.707e-09, 2.032e-09, 2.474e-09,
00651
                                 2.977e-09, 3.506e-09, 4.102e-09, 5.013e-09, 6.493e-09, 8.414e-09,
00652
                                 1.077e-08, 1.367e-08, 1.777e-08, 2.625e-08, 3.926e-08, 5.545e-08,
                                  7.195e-08, 9.464e-08, 1.404e-07, 2.183e-07, 3.329e-07, 4.535e-07,
                                 6.158e-07, 8.187e-07, 1.075e-06, 1.422e-06, 1.979e-06, 2.71e-06,
00655
                                3.58e-06, 4.573e-06, 5.951e-06, 7.999e-06, 1.072e-05, 1.372e-05,
00656
                                1.697e-05, 2.112e-05, 2.643e-05, 3.288e-05, 3.994e-05, 4.794e-05,
                                 5.606e-05, 6.383e-05, 7.286e-05, 8.156e-05, 8.883e-05, 9.469e-05, 9.848e-05, 0.0001023, 0.0001066, 0.0001115, 0.0001145, 0.0001142,
00657
00658
00659
                               0.0001133
00661
00662
                         static double no2[121] = {
                               3.036e-09, 2.945e-10, 9.982e-11, 5.069e-11, 3.485e-11, 2.982e-11,
00663
                                 2.947e-11, 3.164e-11, 3.714e-11, 4.586e-11, 6.164e-11, 8.041e-11, 9.982e-11, 1.283e-10, 1.73e-10, 2.56e-10, 3.909e-10, 5.959e-10,
00664
00665
                                 9.081e-10, 1.384e-09, 1.788e-09, 2.189e-09, 2.686e-09, 3.091e-09,
                                 3.49e-09, 3.796e-09, 4.2e-09, 5.103e-09, 6.005e-09, 6.3e-09, 6.706e-09,
00667
00668
                                 7.07e-09, 7.434e-09, 7.663e-09, 7.788e-09, 7.8e-09, 7.597e-09,
                                7.482e-09, 7.227e-09, 6.403e-09, 5.585e-09, 4.606e-09, 3.703e-09, 2.984e-09, 2.183e-09, 1.48e-09, 8.441e-10, 5.994e-10, 3.799e-10,
00669
00670
00671
                                 2.751e-10, 1.927e-10, 1.507e-10, 1.102e-10, 6.971e-11, 5.839e-11,
                                 3.904e-11, 3.087e-11, 2.176e-11, 1.464e-11, 1.209e-11, 8.497e-12,
                                 6.477e-12, 4.371e-12, 2.914e-12, 2.424e-12, 1.753e-12, 1.35e-12,
00673
00674
                                 9.417e-13, 6.622e-13, 5.148e-13, 3.841e-13, 3.446e-13, 3.01e-13,
00675
                                2.551e-13, 2.151e-13, 1.829e-13, 1.64e-13, 1.475e-13, 1.352e-13,
                                 1.155e-13, 9.963e-14, 9.771e-14, 9.577e-14, 9.384e-14, 9.186e-14, 9e-14, 9e-14,
00676
00677
```

```
9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14,
                    9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14
00679
00680
00681
00682
00683
                static double o3[121] = {
                    2.218e-08, 3.394e-08, 3.869e-08, 4.219e-08, 4.501e-08, 4.778e-08,
                     5.067e-08, 5.402e-08, 5.872e-08, 6.521e-08, 7.709e-08, 9.461e-08,
00685
00686
                     1.269e-07, 1.853e-07, 2.723e-07, 3.964e-07, 5.773e-07, 8.2e-07,
                    1.155e-06, 1.59e-06, 2.076e-06, 2.706e-06, 3.249e-06, 3.848e-06, 4.459e-06, 4.986e-06, 5.573e-06, 5.958e-06, 6.328e-06, 6.661e-06, 6.9e-06, 7.146e-06, 7.276e-06, 7.374e-06, 7.447e-06, 7.383e-06,
00687
00688
00689
00690
                      7.321e-06, 7.161e-06, 6.879e-06, 6.611e-06, 6.216e-06, 5.765e-06,
                     5.355e-06, 4.905e-06, 4.471e-06, 4.075e-06, 3.728e-06, 3.413e-06,
00691
00692
                     3.125e-06, 2.856e-06, 2.607e-06, 2.379e-06, 2.17e-06, 1.978e-06,
                     1.8e-06, 1.646e-06, 1.506e-06, 1.376e-06, 1.233e-06, 1.102e-06, 9.839e-07, 8.771e-07, 7.814e-07, 6.947e-07, 6.102e-07, 5.228e-07, 4.509e-07, 3.922e-07, 3.501e-07, 3.183e-07, 2.909e-07, 2.686e-07,
00693
00694
00695
                     2.476e-07, 2.284e-07, 2.109e-07, 2.003e-07, 2.013e-07, 2.022e-07,
                     2.032e-07, 2.042e-07, 2.097e-07, 2.361e-07, 2.656e-07,
00697
                     3.37e-07, 3.826e-07, 4.489e-07, 5.26e-07, 6.189e-07, 7.312e-07, 8.496e-07, 8.444e-07, 8.392e-07, 8.339e-07, 8.286e-07, 8.234e-07,
00698
00699
                    8.181e-07, 8.129e-07, 8.077e-07, 8.026e-07, 6.918e-07, 5.176e-07, 3.865e-07, 2.885e-07, 2.156e-07, 1.619e-07, 1.219e-07, 9.161e-08, 6.972e-08, 5.399e-08, 3.498e-08, 2.111e-08, 1.322e-08, 8.482e-09,
00700
00701
00702
                     5.527e-09, 3.423e-09, 2.071e-09, 1.314e-09, 8.529e-10, 5.503e-10,
00703
00704
                    3.665e-10
00705
                };
00706
00707
                static double ocs[121] = {
00708
                   6e-10, 6e-10, 6e-10, 6e-10, 6e-10, 6e-10, 6e-10, 6e-10, 5.997e-10,
                     5.989e-10, 5.881e-10, 5.765e-10, 5.433e-10, 5.074e-10, 4.567e-10,
00710
                     4.067e-10, 3.601e-10, 3.093e-10, 2.619e-10, 2.232e-10,
                                                                                                                                          1.805e-10
00711
                     1.46 e^{-10}, \ 1.187 e^{-10}, \ 8.03 e^{-11}, \ 5.435 e^{-11}, \ 3.686 e^{-11}, \ 2.217 e^{-11},
00712
                     1.341e-11, 8.756e-12, 4.511e-12, 2.37e-12, 1.264e-12, 8.28e-13,
                     5.263e-13, 3.209e-13, 1.717e-13, 9.068e-14, 4.709e-14, 2.389e-14,
00713
00714
                     1.236e-14, 1.127e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14,
                     1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14,
00716
                     1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14,
00717
                     1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14,
00718
                    1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14,
00719
                    1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14,
                    1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 
00720
                     1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14,
00722
00723
                     1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14,
00724
                    1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14,
                    1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14
00725
00726
00727
00728
00729
00730
                static double sf6[121] = {
                  4.103e-12, 4.103e-12, 4.103e-12, 4.103e-12, 4.103e-12, 4.103e-12,
00731
                     4.103e-12, 4.103e-12, 4.103e-12, 4.087e-12, 4.064e-12, 4.023e-12, 3.988e-12, 3.941e-12, 3.884e-12, 3.755e-12, 3.622e-12, 3.484e-12, 3.32e-12, 3.144e-12, 2.978e-12, 2.811e-12, 2.653e-12, 2.489e-12,
00732
00733
00735
                     2.332e-12, 2.199e-12, 2.089e-12, 2.013e-12, 1.953e-12, 1.898e-12,
00736
                    1.859e-12, 1.826e-12, 1.798e-12, 1.776e-12, 1.757e-12, 1.742e-12,
00737
                    1.728e-12, 1.717e-12, 1.707e-12, 1.698e-12, 1.691e-12, 1.685e-12,
                     1.679e-12, 1.675e-12, 1.671e-12, 1.668e-12, 1.665e-12, 1.663e-12,
00738
00739
                    1.661e-12, 1.659e-12, 1.658e-12, 1.657e-12, 1.656e-12, 1.655e-12,
                     1.654e-12, 1.653e-12, 1.653e-12, 1.652e-12, 1.652e-12, 1.652e-12,
00741
                     1.651e-12, 1.651e-12, 1.651e-12, 1.651e-12, 1.651e-12,
00742
                     1.651e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12,
00743
                    1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12,
00744
                     1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12,
00745
                     1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12,
                     1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12,
00747
                     1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12,
00748
                     1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12,
00749
                    1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12
00750
                };
00751
00752
                static double so2[121] = {
                   le-10, le-10, le-10, le-10, le-10, le-10, le-10, le-10, le-10, le-10,
00753
00754
                     1e-10, 1e-10, 9.867e-11, 9.537e-11, 9e-11, 8.404e-11, 7.799e-11,
00755
                     7.205e-11, 6.616e-11, 6.036e-11, 5.475e-11, 5.007e-11, 4.638e-11,
00756
                     4.346e-11, 4.055e-11, 3.763e-11, 3.471e-11, 3.186e-11, 2.905e-11,
                    2.631e-11, 2.358e-11, 2.415e-11, 2.949e-11, 3.952e-11, 5.155e-11, 6.76e-11, 8.741e-11, 1.099e-10, 1.278e-10, 1.414e-10, 1.512e-10, 1.607e-10, 1.699e-10, 1.774e-10, 1.832e-10, 1.871e-10, 1.907e-10,
00757
00758
00760
                     1.943e-10, 1.974e-10, 1.993e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10,
00761
                     2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10,
00762
                     2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10,
00763
                     2e-10, 2e
00764
```

```
2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10,
          2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10
00766
00767
00768
00769
00770
        static int ig co2 = -999;
00771
00772
        double co2, *q[NG] = {NULL};
00773
00774
        int ig, ip, iw, iz;
00775
00776
        /* Find emitter index of CO2... */
00777
        if (ig_co2 == -999)
00778
          ig_co2 = find_emitter(ct1, "CO2");
00779
00780
        /* Identify variable... */
        for (ig = 0; ig < ctl->ng; ig++) {
   q[ig] = NULL;
00781
00782
          if (strcasecmp(ctl->emitter[ig], "C2H2") == 0)
00784
            q[ig] = c2h2;
00785
          if (strcasecmp(ctl->emitter[ig], "C2H6") == 0)
            q[ig] = c2h6;
00786
00787
          if (strcasecmp(ctl->emitter[ig], "CC14") == 0)
00788
            q[ig] = ccl4;
00789
          if (strcasecmp(ctl->emitter[iq], "CH4") == 0)
00790
            q[ig] = ch4;
             (strcasecmp(ctl->emitter[ig], "ClO") == 0)
00791
          if
            q[ig] = clo;
00792
          if (strcasecmp(ctl->emitter[ig], "ClONO2") == 0)
00793
00794
           q[ig] = clono2;
          if (strcasecmp(ctl->emitter[ig], "CO") == 0)
00795
00796
            q[ig] = co;
00797
             (strcasecmp(ctl->emitter[ig], "COF2") == 0)
            q[ig] = cof2;
00798
00799
          if (strcasecmp(ctl->emitter[ig], "F11") == 0)
00800
            q[ig] = f11;
00801
          if (strcasecmp(ctl->emitter[iq], "F12") == 0)
            q[ig] = f12;
00803
          if (strcasecmp(ctl->emitter[ig], "F14") == 0)
00804
            q[ig] = f14;
          if (strcasecmp(ctl->emitter[ig], "F22") == 0)
00805
00806
            q[ig] = f22;
          if (strcasecmp(ctl->emitter[ig], "H2O") == 0)
00807
80800
            q[ig] = h2o;
          if (strcasecmp(ctl->emitter[ig], "H2O2") == 0)
00809
            q[ig] = h2o2;
00810
00811
          if (strcasecmp(ctl->emitter[ig], "HCN") == 0)
00812
            q[ig] = hcn;
          if (strcasecmp(ctl->emitter[ig], "HNO3") == 0)
00813
            q[ig] = hno3;
00814
00815
             (strcasecmp(ctl->emitter[ig], "HNO4") == 0)
00816
            q[ig] = hno4;
00817
          if (strcasecmp(ctl->emitter[ig], "HOC1") == 0)
          q[ig] = hocl;
if (strcasecmp(ctl->emitter[ig], "N2O") == 0)
00818
00819
00820
            q[ig] = n2o;
          if (strcasecmp(ctl->emitter[ig], "N2O5") == 0)
            q[ig] = n2o5;
00822
00823
          if (strcasecmp(ctl->emitter[ig], "NH3") == 0)
00824
            q[ig] = nh3;
00825
          if (strcasecmp(ctl->emitter[iq], "NO") == 0)
00826
            q[ig] = no;
00827
          if (strcasecmp(ctl->emitter[ig], "NO2") == 0)
            q[ig] = no2;
00828
00829
          if (strcasecmp(ctl->emitter[ig], "03") == 0)
            q[ig] = 03;
00830
          if (strcasecmp(ctl->emitter[ig], "OCS") == 0)
00831
00832
            q[iq] = ocs;
          if (strcasecmp(ctl->emitter[ig], "SF6") == 0)
00833
            q[ig] = sf6;
00835
              (strcasecmp(ctl->emitter[ig], "SO2") == 0)
00836
            q[ig] = so2;
00837
00838
00839
        /* Loop over atmospheric data points... */
        for (ip = 0; ip < atm->np; ip++) {
00840
00841
00842
          /\star Get altitude index... \star/
00843
          iz = locate_reg(z, 121, atm->z[ip]);
00844
00845
          /* Interpolate pressure... */
00846
          atm \rightarrow p[ip] = EXP(z[iz], pre[iz], z[iz + 1], pre[iz + 1], atm \rightarrow z[ip]);
00847
00848
          /* Interpolate temperature... */
00849
          atm \rightarrow t[ip] = LIN(z[iz], tem[iz], z[iz + 1], tem[iz + 1], atm \rightarrow z[ip]);
00850
00851
          /* Interpolate trace gases... */
```

```
for (ig = 0; ig < ctl->ng; ig++)
           if (q[ig] != NULL)
00854
              atm->q[ig][ip]
00855
               LIN(z[iz], q[ig][iz], z[iz + 1], q[ig][iz + 1], atm->z[ip]);
00856
00857
              atm->\alpha[iq][ip] = 0;
00859
          /* Set CO2... *
00860
          if (ig_co2 >= 0) {
00861
            co2 =
              371.789948e-6 + 2.026214e-6 * (atm->time[ip] - 63158400.) / 31557600.;
00862
            atm->q[ig\_co2][ip] = co2;
00863
00864
00865
00866
          /* Set extinction to zero... */
00867
          for (iw = 0; iw < ctl->nw; iw++)
            atm->k[iw][ip] = 0;
00868
00869
00870 }
```

Here is the call graph for this function:



5.3.2.6 double ctmco2 (double nu, double p, double t, double u)

Compute carbon dioxide continuum (optical depth).

Definition at line 874 of file jurassic.c.

```
00878
00879
              static double co2296[2001] = { 9.3388e-5, 9.7711e-5, 1.0224e-4, 1.0697e-4,
00880
                  1.1193e-4, 1.1712e-4, 1.2255e-4, 1.2824e-4, 1.3419e-4, 1.4043e-4,
                   1.4695e-4, 1.5378e-4, 1.6094e-4, 1.6842e-4, 1.7626e-4, 1.8447e-4,
                  1.9307e-4, 2.0207e-4, 2.1149e-4, 2.2136e-4, 2.3169e-4, 2.4251e-4, 2.5384e-4, 2.657e-4, 2.7813e-4, 2.9114e-4, 3.0477e-4, 3.1904e-4,
00883
00884
                  3.3399e-4, 3.4965e-4, 3.6604e-4, 3.8322e-4, 4.0121e-4, 4.2006e-4, 4.398e-4, 4.6047e-4, 4.8214e-4, 5.0483e-4, 5.286e-4, 5.535e-4, 5.7959e-4, 6.0693e-4, 6.3557e-4, 6.6558e-4, 6.9702e-4, 7.2996e-4,
00885
00886
00887
                  7.6449e-4, 8.0066e-4, 8.3856e-4, 8.7829e-4, 9.1991e-4,
                                                                                                                        9.6354e-4,
                  .0010093, .0010572, .0011074, .00116, .0012152, .001273, .0013336, .0013972, .0014638, .0015336, .0016068, .0016835,
00889
00890
                  .001764, .0018483, .0019367, .0020295, .0021267, .0022286, .0023355, .0024476, .0025652, .0026885, .0028178, .0029534,
00891
00892
00893
                   .0030956, .0032448, .0034012, .0035654, .0037375, .0039181,
                  .0041076, .0043063, .0045148, .0047336, .0049632, .005204, .0054567, .0057219, .0060002, .0062923, .0065988, .0069204,
00894
00895
                  .007258, .0076123, .0079842, .0083746, .0087844, .0092146, .0096663, .01014, .010638, .011161, .01171, .012286, .012891,
00896
00897
                   .013527, .014194, .014895, .015631, .016404, .017217, .01807,
00898
                  .018966, .019908, .020897, .021936, .023028, .024176, .025382, .026649, .027981, .02938, .030851, .032397, .034023, .035732, .037528, .039416, .041402, .04349, .045685, .047994, .050422, .052975, .055661, .058486, .061458, .064584, .067873, .071334,
00899
00900
00901
00902
                  .074975, .078807, .082839, .087082, .091549, .09649, .1012, .10641, .11189, .11767, .12375, .13015, .13689, .14399, .15147, .15935, .16765, .17639, .18561, .19531, .20554, .21632, .22769, .23967, .25229, .2656, .27964, .29443, .31004, .3265, .34386, .36218, .3815, .40188, .42339, .44609, .47004, .49533, .52202,
00903
00904
00905
00906
```

```
.5502, .57995, .61137, .64455, .6796, .71663, .75574, .79707,
               .84075, .88691, .9357, .98728, 1.0418, 1.0995, 1.1605, 1.225, 1.2932, 1.3654, 1.4418, 1.5227, 1.6083, 1.6989, 1.7948, 1.8964, 2.004, 2.118, 2.2388, 2.3668, 2.5025, 2.6463, 2.7988, 2.9606,
00909
00910
00911
00912
               3.1321, 3.314, 3.5071, 3.712, 3.9296, 4.1605, 4.4058, 4.6663, 4.9431, 5.2374, 5.5501, 5.8818, 6.2353, 6.6114, 7.0115, 7.4372,
00913
               7.8905, 8.3731, 8.8871, 9.4349, 10.019, 10.641, 11.305, 12.013,
               12.769, 13.576, 14.437, 15.358, 16.342, 17.39, 18.513, 19.716,
00915
00916
               21.003, 22.379, 23.854, 25.436, 27.126, 28.942, 30.89, 32.973,
               35.219, 37.634, 40.224, 43.021, 46.037, 49.29, 52.803, 56.447, 60.418, 64.792, 69.526, 74.637, 80.182, 86.193, 92.713, 99.786, 107.47, 115.84, 124.94, 134.86, 145.69, 157.49, 170.3, 184.39, 199.83, 216.4, 234.55, 254.72, 276.82, 299.85, 326.16, 354.99, 386.51, 416.68, 449.89, 490.12, 534.35, 578.25, 632.26, 692.61,
00917
00918
00919
00920
00921
00922
               756.43, 834.75, 924.11, 1016.9, 996.96, 1102.7, 1219.2, 1351.9,
               1494.3, 1654.1, 1826.5, 2027.9, 2249., 2453.8, 2714.4, 2999.4, 3209.5, 3509., 3840.4, 3907.5, 4190.7, 4533.5, 4648.3, 5059.1, 5561.6, 6191.4, 6820.8, 7905.9, 9362.2, 2431.3, 2211.3, 2046.8,
00923
00924
00925
               2023.8, 1985.9, 1905.9, 1491.1, 1369.8, 1262.2, 1200.7, 887.74,
               820.25, 885.23, 887.21, 816.73, 1126.9, 1216.2, 1272.4, 1579.5,
               1634.2, 1656.3, 1657.9, 1789.5, 1670.8, 1509.5, 8474.6, 7489.2, 6793.6, 6117., 5574.1, 5141.2, 5084.6, 4745.1, 4413.2, 4102.8,
00928
00929
               4024.7, 3715., 3398.6, 3100.8, 2900.4, 2629.2, 2374., 2144.7, 1955.8, 1760.8, 1591.2, 1435.2, 1296.2, 1174., 1065.1, 967.76, 999.48, 897.45, 809.23, 732.77, 670.26, 611.93, 560.11, 518.77,
00930
00931
00932
                476.84, 438.8, 408.48, 380.21, 349.24, 322.71, 296.65, 272.85,
               251.96, 232.04, 213.88, 197.69, 182.41, 168.41, 155.79, 144.05,
00934
              133.31, 123.48, 114.5, 106.21, 98.591, 91.612, 85.156, 79.204, 73.719, 68.666, 63.975, 59.637, 56.35, 52.545, 49.042, 45.788, 42.78, 39.992, 37.441, 35.037, 32.8, 30.744, 28.801, 26.986, 25.297, 23.731, 22.258, 20.883, 19.603, 18.403, 17.295, 16.249,
00935
00936
00937
00938
00939
               15.271, 14.356, 13.501, 12.701, 11.954, 11.254, 10.6, 9.9864,
               9.4118, 8.8745, 8.3714, 7.8997, 7.4578, 7.0446, 6.6573, 6.2949,
00940
00941
               5.9577, 5.6395, 5.3419, 5.063, 4.8037, 4.5608, 4.3452, 4.1364,
               3.9413, 3.7394, 3.562, 3.3932, 3.2325, 3.0789, 2.9318, 2.7898, 2.6537, 2.5225, 2.3958, 2.2305, 2.1215, 2.0245, 1.9427, 1.8795, 1.8336, 1.7604, 1.7016, 1.6419, 1.5282, 1.4611, 1.3443, 1.27,
00942
00943
00944
               1.1675, 1.0824, 1.0534, .99833, .95854, .92981, .90887, .89346,
00946
               .88113, .87068, .86102, .85096, .88262, .86151, .83565, .80518,
               .77045, .73736, .74744, .74954, .75773, .82267, .83493, .89402,
00947
00948
                .89725, .93426, .95564, .94045, .94174, .93404, .92035, .90456,
                .88621, .86673, .78117, .7515, .72056, .68822, .65658, .62764,
00949
               .55984, .55598, .57407, .60963, .63763, .66198, .61132, .60972, .52496, .50649, .41872, .3964, .32422, .27276, .24048, .23772,
00950
               .2286, .22711, .23999, .32038, .34371, .36621, .38561, .39953,
00952
00953
                .40636, .44913, .42716, .3919, .35477, .33935, .3351, .39746,
00954
                .40993, .49398, .49956, .56157, .54742, .57295, .57386, .55417,
               . 50745, .471, .43446, .39102, .34993, .31269, .27888, .24912, .22291, .19994, .17972, .16197, .14633, .13252, .12029, .10942, .099745, .091118, .083404, .076494, .070292, .064716, .059697, .055173, .051093, .047411, .044089, .041092, .038392, .035965,
00955
00956
00957
               .033789, .031846, .030122, .028607, .02729, .026169, .025209,
00959
00960
               .024405, .023766, .023288, .022925, .022716, .022681, .022685,
               .022768, .023133, .023325, .023486, .024004, .024126, .024083, .023785, .024023, .023029, .021649, .021108, .019454, .017809,
00961
00962
               .017292, .016635, .017037, .018068, .018977, .018756, .017847, .016557, .016142, .014459, .012869, .012381, .010875, .0098701, .009285, .0091698, .0091701, .0096145, .010553, .01106, .012613,
00963
00965
               .014362, .015017, .016507, .017741, .01768, .017784, .0171, .016357, .016172, .017257, .018978, .020935, .021741, .023567,
00966
00967
00968
                .025183, .025589, .026732, .027648, .028278, .028215, .02856,
               .029163, .029062, .028851, .028497, .027825, .027801, .026523, .02487, .022967, .022168, .020194, .018605, .017903, .018439, .019697, .020311, .020855, .020057, .018608, .016738, .015963, .013844, .011801, .011134, .0097573, .0086007, .0086226,
00969
00971
00972
00973
                .0083721, .0090978, .0097616, .0098426, .011317, .012853,
                                                                                                           .01447.
               .014657, .015771, .016351, .016079, .014829, .013431, .013185, .013207, .01448, .016176, .017971, .018265, .019526, .020455, .019797, .019802, .0194, .018176, .017505, .016197, .015339,
00974
00975
               .014401, .013213, .012203, .011186, .010236, .0093288,
               .0076837, .0069375, .0062614, .0056628, .0051153, .0046015,
00978
00979
               .0041501, .003752, .0033996, .0030865, .0028077, .0025586,
               .0023355, .0021353, .0019553, .0017931, .0016466, .0015141, .0013941, .0012852, .0011862, .0010962, .0010142, 9.3935e-4, 8.71e-4, 8.0851e-4, 7.5132e-4, 6.9894e-4, 6.5093e-4, 6.0689e-4,
00980
00981
00982
               5.6647e-4, 5.2935e-4, 4.9525e-4, 4.6391e-4, 4.3509e-4, 4.086e-4,
               3.8424e-4, 3.6185e-4, 3.4126e-4, 3.2235e-4, 3.0498e-4, 2.8904e-4,
00984
00985
               2.7444e-4, 2.6106e-4, 2.4883e-4, 2.3766e-4, 2.275e-4, 2.1827e-4,
00986
               2.0992e-4, 2.0239e-4, 1.9563e-4, 1.896e-4, 1.8427e-4, 1.796e-4,
               1.7555e-4, 1.7209e-4, 1.692e-4, 1.6687e-4, 1.6505e-4, 1.6375e-4,
00987
               1.6294e-4, 1.6261e-4, 1.6274e-4, 1.6334e-4, 1.6438e-4, 1.6587e-4,
00988
                1.678e-4, 1.7017e-4, 1.7297e-4, 1.762e-4, 1.7988e-4, 1.8399e-4,
                1.8855e-4, 1.9355e-4, 1.9902e-4, 2.0494e-4, 2.1134e-4, 2.1823e-4,
00990
00991
               2.2561e-4, 2.335e-4, 2.4192e-4, 2.5088e-4, 2.604e-4, 2.705e-4,
00992
               2.8119e-4, 2.9251e-4, 3.0447e-4, 3.171e-4, 3.3042e-4, 3.4447e-4,
               3.5927e-4, 3.7486e-4, 3.9127e-4, 4.0854e-4, 4.267e-4, 4.4579e-4, 4.6586e-4, 4.8696e-4, 5.0912e-4, 5.324e-4, 5.5685e-4, 5.8253e-4,
00993
00994
```

```
6.0949e-4, 6.378e-4, 6.6753e-4, 6.9873e-4, 7.3149e-4, 7.6588e-4,
                   8.0198e-4, 8.3987e-4, 8.7964e-4, 9.2139e-4, 9.6522e-4, .0010112,
00996
00997
                   .0010595, .0011102, .0011634, .0012193, .001278, .0013396,
00998
                    .0014043, .0014722, .0015436, .0016185, .0016972, .0017799,
                   .0018668, .001958, .0020539, .0021547, .0022606, .0023719, .002489, .002612, .0027414, .0028775, .0030206, .0031712, .0033295, .0034962, .0036716, .0038563, .0040506, .0042553,
00999
01000
                    .0044709, .004698, .0049373, .0051894, .0054552, .0057354,
01002
                   .006031, .0063427, .0066717, .0070188, .0073854, .0077726, .0081816, .0086138, .0090709, .0095543, .010066, .010607, .011181, .011789, .012433, .013116, .013842, .014613, .015432,
01003
01004
01005
                   .016304, .017233, .018224, .019281, .020394, .021574, .022836, .024181, .025594, .027088, .028707, .030401, .032245, .034219, .036262, .038539, .040987, .043578, .04641, .04949, .052726, .056326, .0602, .064093, .068521, .073278, .077734, .083064,
01006
01008
01009
                   .088731, .093885, .1003, .1072, .11365, .12187, .13078, .13989, .15095, .16299, .17634, .19116, .20628, .22419, .24386, .26587,
01010
01011
                   . 28811, .31399, .34321, .36606, .39675, .42742, .44243, .47197, .49993, .49027, .51147, .52803, .48931, .49729, .5026, .43854,
01012
                   .441, .44766, .43414, .46151, .50029, .55247, .43855, .32115, .32607, .3431, .36119, .38029, .41179, .43996, .47144, .51853, .55362, .59122, .66338, .69877, .74001, .82923, .86907, .90361,
01014
01015
01016
                   1.0025, 1.031, 1.0559, 1.104, 1.1178, 1.1341, 1.1547, 1.351, 1.4772, 1.4812, 1.4907, 1.512, 1.5442, 1.5853, 1.6358, 1.6963, 1.7674, 1.8474, 1.9353, 2.0335, 2.143, 2.2592, 2.3853, 2.5217,
01017
01018
                   2.6686, 2.8273, 2.9998, 3.183, 3.3868, 3.6109, 3.8564, 4.1159,
                   4.4079, 4.7278, 5.0497, 5.3695, 5.758, 6.0834, 6.4976, 6.9312,
01021
01022
                   7.38, 7.5746, 7.9833, 8.3791, 8.3956, 8.7501, 9.1067, 9.072,
                   9.4649, 9.9112, 10.402, 10.829, 11.605, 12.54, 12.713, 10.443, 10.825, 11.375, 11.955, 12.623, 13.326, 14.101, 15.041, 15.547, 16.461, 17.439, 18.716, 19.84, 21.036, 22.642, 23.901, 25.244, 27.03, 28.411, 29.871, 31.403, 33.147, 34.744, 36.456, 39.239,
01023
01024
01025
                   43.605, 45.162, 47.004, 49.093, 51.391, 53.946, 56.673, 59.629, 63.167, 66.576, 70.254, 74.222, 78.477, 83.034, 87.914, 93.18,
01027
01028
01029
                   98.77, 104.74, 111.15, 117.95, 125.23, 133.01, 141.33, 150.21,
                   159.71, 169.89, 180.93, 192.54, 204.99, 218.34, 232.65, 248., 264.47, 282.14, 301.13, 321.53, 343.48, 367.08, 392.5, 419.88, 449.4, 481.26, 515.64, 552.79, 592.99, 636.48, 683.61, 734.65,
01030
01031
01033
                   789.99, 850.02, 915.14, 985.81, 1062.5, 1147.1, 1237.8, 1336.4,
                   1443.2, 1558.9, 1684.2, 1819.2, 1965.2, 2122.6, 2291.7, 2470.8,
01034
                   2665.7, 2874.9, 3099.4, 3337.9, 3541., 3813.3, 4111.9, 4439.3, 4798.9, 5196., 5639.2, 6087.5, 6657.7, 7306.7, 8040.7, 8845.5,
01035
01036
                   9702.2, 10670., 11739., 12842., 14141., 15498., 17068., 18729., 20557., 22559., 25248., 27664., 30207., 32915., 35611., 38081.,
01037
                   40715., 43191., 41651., 42750., 43785., 44353., 44366., 44189., 43618., 42862., 41878., 35133., 35215., 36383., 39420., 44055.,
01040
01041
                   44155., 45850., 46853., 39197., 38274., 29942., 28553., 21792.,
                   21228., 17106., 14955., 18181., 19557., 21427., 23728., 26301.,
01042
                   28584., 30775., 32536., 33867., 40089., 39204., 37329., 34452., 31373., 33921., 34800., 36043., 44415., 45162., 52181., 50895., 54140., 50840., 50468., 48302., 44915., 40910., 36754., 32755.,
01043
01044
                   29093., 25860., 22962., 20448., 18247., 16326., 14645., 13165., 11861., 10708., 9686.9, 8779.7, 7971.9, 7250.8, 6605.7, 6027.2,
01046
01047
                   5507.3, 5039.1, 4616.6, 4234.8, 3889., 3575.4, 3290.5, 3031.3, 2795.2, 2579.9, 2383.1, 2203.3, 2038.6, 1887.6, 1749.1, 1621.9, 1505., 1397.4, 1298.3, 1207., 1122.8, 1045., 973.1, 906.64, 845.16, 788.22, 735.48, 686.57, 641.21, 599.1, 559.99, 523.64, 489.85, 458.42, 429.16, 401.92, 376.54, 352.88, 330.82, 310.24,
01048
01049
01050
01052
                   291.03, 273.09, 256.34, 240.69, 226.05, 212.37, 199.57, 187.59
01053
                   176.37, 165.87, 156.03, 146.82, 138.17, 130.07, 122.47, 115.34, 108.65, 102.37, 96.473, 90.934, 85.73, 80.84, 76.243, 71.922, 67.858, 64.034, 60.438, 57.052, 53.866, 50.866, 48.04, 45.379, 42.872, 40.51, 38.285, 36.188, 34.211, 32.347, 30.588, 28.929,
01054
01055
01056
                   27.362, 25.884, 24.489, 23.171, 21.929, 20.755, 19.646, 18.599, 17.61, 16.677, 15.795, 14.961, 14.174, 13.43, 12.725, 12.06,
01059
01060
                   11.431, 10.834, 10.27, 9.7361, 9.2302, 8.7518, 8.2997, 7.8724,
                   7.4674, 7.0848, 6.7226, 6.3794, 6.054, 5.745, 5.4525, 5.1752, 4.9121, 4.6625, 4.4259, 4.2015, 3.9888, 3.7872, 3.5961, 3.4149, 3.2431, 3.0802, 2.9257, 2.7792, 2.6402, 2.5084, 2.3834, 2.2648, 2.1522, 2.0455, 1.9441, 1.848, 1.7567, 1.6701, 1.5878, 1.5097,
01061
01062
01063
                   1.4356, 1.3651, 1.2981, 1.2345, 1.174, 1.1167, 1.062, 1.0101,
01065
01066
                   .96087, .91414, .86986, .82781, .78777, .74971, .71339, .67882,
01067
                    .64604, .61473, .58507, .55676, .52987, .5044, .48014, .45715,
                   .43527, .41453, .3948, .37609, .35831, .34142, .32524, .30995, .29536, .28142, .26807, .25527, .24311, .23166, .22077, .21053, .20081, .19143, .18261, .17407, .16603, .15833, .15089, .14385, .13707, .13065, .12449, .11865, .11306, .10774, .10266, .097818
01068
01069
01071
                   .093203, .088815, .084641, .080671, .076892, .073296, .069873, .066613, .06351, .060555, .05774, .055058, .052504, .050071, .047752, .045543, .043438, .041432, .039521, .037699, .035962,
01072
01073
01074
01075
                   .034307, .032729, .031225, .029791, .028423, .02712, .025877, .024692, .023563, .022485, .021458, .020478, .019543, .018652,
                   .017802, .016992, .016219, .015481, .014778, .014107, .013467, .012856, .012274, .011718, .011188, .010682, .0102, .0097393,
01077
01078
                   .0093001, .008881, .0084812, .0080997, .0077358, .0073885, .0070571, .0067409, .0064393, .0061514, .0058768, .0056147, .0053647, .0051262, .0048987, .0046816, .0044745, .0042769,
01079
01080
01081
```

```
.0040884, .0039088, .0037373, .0035739, .003418, .0032693,
                .0031277, .0029926, .0028639, .0027413, .0026245, .0025133, .0024074, .0023066, .0022108, .0021196, .002033, .0019507,
01084
01085
                 .0018726, .0017985, .0017282, .0016617, .0015988, .0015394,
01086
                .0014834, .0014306, .0013811, .0013346, .0012911, .0012506,
                .0012131, .0011784, .0011465, .0011175, .0010912, .0010678, .0010472, .0010295, .0010147, .001003, 9.9428e-4, 9.8883e-4
01087
                9.8673e-4, 9.8821e-4, 9.9343e-4, .0010027, .0010164, .0010348,
01089
01090
                .0010586, .0010882, .0011245, .0011685, .0012145, .0012666,
                .0013095, .0013688, .0014048, .0014663, .0015309, .0015499, .0016144, .0016312, .001705, .0017892, .0018499, .0019715,
01091
01092
               .002102, .0022442, .0024284, .0025893, .0027703, .0029445, .0031193, .003346, .0034552, .0036906, .0037584, .0040084, .0041934, .0044587, .0047093, .0049759, .0053421, .0055134, .0059048, .0058663, .0061036, .0063259, .0059657, .0060653,
01093
01094
01095
01096
01097
                .0060972, .0055539, .0055653, .0055772, .005331, .0054953,
                .0055919, .0058684, .006183, .0066675, .0069808, .0075142,
01098
                .0078536, .0084282, .0089454, .0094625, .0093703, .0095857, .0099283, .010063, .010521, .0097778, .0098175, .010379, .010447,
01099
                .0105, .010617, .010706, .01078, .011177, .011212, .011304,
                .011446, .011603, .011816, .012165, .012545, .013069, .013539, .01411, .014776, .016103, .017016, .017994, .018978, .01998,
01102
01103
                 .021799, .022745, .023681, .024627, .025562, .026992, .027958,
01104
               .029013, .030154, .031402, .03228, .033651, .035272, .037088, .039021, .041213, .043597, .045977, .04877, .051809, .054943, .058064, .061528, .06537, .069309, .071928, .075752, .079589, .083352, .084096, .087497, .090817, .091198, .094966, .099045
01105
01106
01108
01109
                 .10429, .10867, .11518, .12269, .13126, .14087, .15161, .16388,
01110
                .16423, .1759, .18721, .19994, .21275, .22513, .23041, .24231,
                25299, 26396, 26396, 27696, 27929, 2908, 30595, 31433, 3282, 3429, 35944, 37467, 39277, 41245, 43326, 45649, 48152, 51897, 54686, 57877, 61263, 64962, 68983, 73945, 78619, 83537, 89622, 95002, 1.0067, 1.0742, 1.1355, 1.2007,
01111
01112
01113
01114
01115
                1.2738, 1.347, 1.4254, 1.5094, 1.6009, 1.6976, 1.8019, 1.9148,
                2.0357, 2.166, 2.3066, 2.4579, 2.6208, 2.7966, 2.986, 3.188, 3.4081, 3.6456, 3.9, 4.1747, 4.4712, 4.7931, 5.1359, 5.5097,
01116
01117
                5.9117, 6.3435, 6.8003, 7.3001, 7.8385, 8.3945, 9.011, 9.6869, 10.392, 11.18, 12.036, 12.938, 13.944, 14.881, 16.029, 17.255,
01118
                18.574, 19.945, 21.38, 22.9, 24.477, 26.128, 27.87, 29.037,
                30.988, 33.145, 35.506, 37.76, 40.885, 44.487, 48.505, 52.911, 57.56, 61.964, 67.217, 72.26, 78.343, 85.08, 91.867, 99.435,
01121
01122
                107.68, 116.97, 127.12, 138.32, 150.26, 163.04, 174.81, 189.26, 205.61, 224.68, 240.98, 261.88, 285.1, 307.58, 334.35, 363.53, 394.68, 427.85, 458.85, 489.25, 472.87, 486.93, 496.27, 501.52,
01123
01124
                501.57, 497.14, 488.09, 476.32, 393.76, 388.51, 393.42, 414.45, 455.12, 514.62, 520.38, 547.42, 562.6, 487.47, 480.83, 391.06,
01127
01128
                376.92, 303.7, 295.91, 256.03, 236.73, 280.38, 310.71, 335.53,
               367.88, 401.94, 435.52, 469.13, 497.94, 588.82, 597.94, 597.2, 588.28, 571.2, 555.75, 603.56, 638.15, 680.75, 801.72, 848.01, 962.15, 990.06, 1068.1, 1076.2, 1115.3, 1134.2, 1136.6, 1119.1, 1108.9, 1090.6, 1068.7, 1041.9, 1005.4, 967.98, 927.08, 780.1, 751.41, 733.12, 742.65, 785.56, 855.16, 852.45, 878.1, 784.59, 787.51
01129
01130
01131
01133
01134
                777.81, 765.13, 622.93, 498.09, 474.89, 386.9, 378.48, 336.17,
01135
                322.04, 329.57, 350.5, 383.38, 420.02, 462.39, 499.71, 531.98,
                554.29, 653.43, 639.99, 605.16, 554.16, 504.42, 540.64, 552.33, 679.46, 699.51, 713.91, 832.17, 919.91, 884.96, 907.57, 846.56, 818.56, 768.93, 706.71, 642.17, 575.95, 515.38, 459.07, 409.02, 364.61, 325.46, 291.1, 260.89, 234.39, 211.01, 190.38, 172.11,
01136
01137
01139
                155.91, 141.49, 128.63, 117.13, 106.84, 97.584, 89.262, 81.756,
01140
               74.975, 68.842, 63.28, 58.232, 53.641, 49.46, 45.649, 42.168, 38.991, 36.078, 33.409, 30.96, 28.71, 26.642, 24.737, 22.985, 21.37, 19.882, 18.512, 17.242, 16.073, 14.987, 13.984, 13.05, 12.186, 11.384, 10.637, 9.9436, 9.2988, 8.6991, 8.141, 7.6215,
01141
01142
01143
01144
                7.1378, 6.6872, 6.2671, 5.8754, 5.51, 5.1691, 4.851, 4.5539,
01146
                4.2764, 4.0169, 3.7742, 3.5472, 3.3348, 3.1359, 2.9495, 2.7749,
01147
                2.6113, 2.4578, 2.3139, 2.1789, 2.0523, 1.9334, 1.8219, 1.7171,
01148
                1.6188, 1.5263, 1.4395, 1.3579, 1.2812, 1.209, 1.1411, 1.0773,
               1.0171, .96048, .90713, .85684, .80959, .76495, .72282, .68309, .64563, .61035, .57707, .54573, .51622, .48834, .46199, .43709, .41359, .39129, .37034, .35064, .33198, .31442, .29784, .28218, .26732, .25337, .24017, .22774, .21601, .20479, .19426
01149
01150
01152
01153
01154
            static double co2260[2001] = { 5.7971e-5, 6.0733e-5, 6.3628e-5, 6.6662e-5,
01155
               6.9843e-5, 7.3176e-5, 7.6671e-5, 8.0334e-5, 8.4175e-5, 8.8201e-5, 9.2421e-5, 9.6846e-5, 1.0149e-4, 1.0635e-4, 1.1145e-4, 1.1679e-4,
01156
                1.224e-4, 1.2828e-4, 1.3444e-4, 1.409e-4, 1.4768e-4, 1.5479e-4,
01158
01159
                1.6224e-4, 1.7006e-4, 1.7826e-4, 1.8685e-4, 1.9587e-4, 2.0532e-4,
01160
                2.1524e-4, 2.2565e-4, 2.3656e-4, 2.48e-4, 2.6001e-4, 2.7261e-4,
                2.8582e-4, 2.9968e-4, 3.1422e-4, 3.2948e-4, 3.4548e-4, 3.6228e-4,
01161
                3.799e-4, 3.9838e-4, 4.1778e-4, 4.3814e-4, 4.595e-4, 4.8191e-4, 5.0543e-4, 5.3012e-4, 5.5603e-4, 5.8321e-4, 6.1175e-4, 6.417e-4,
01162
                 6.7314e-4, 7.0614e-4, 7.4078e-4, 7.7714e-4, 8.1531e-4, 8.5538e-4,
01164
01165
                8.9745e-4, 9.4162e-4, 9.8798e-4, .0010367, .0010878,
                                                                                                         .0011415,
01166
                 .0011978, .001257, .0013191, .0013844, .001453, .0015249,
                .0016006, .00168, .0017634, .001851, .001943, .0020397, .0021412, .0022479, .00236, .0024778, .0026015, .0027316, .0028682,
01167
01168
```

```
.0030117, .0031626, .0033211, .0034877, .0036628, .0038469,
                          .0040403, .0042436, .0044574, .004682, .0049182, .0051665, .0054276, .0057021, .0059907, .0062942, .0066133, .0069489,
01170
01171
                            .0073018, .0076729, .0080632, .0084738, .0089056, .0093599,
01172
                          .0098377, .01034, .010869, .011426, .012011, .012627, .013276, .013958, .014676, .015431, .016226, .017063, .017944, .018872, .019848, .020876, .021958, .023098, .024298, .025561, .026892, .028293, .029769, .031323, .032961, .034686, .036503, .038418,
01173
01174
01176
01177
                            .040435, .042561, .044801, .047161, .049649, .052271, .055035,
                           .057948, .061019, .064256, .06767, .07127, .075066, .079069, .083291, .087744, .092441, .097396, .10262, .10814, .11396, .1201, .12658, .13342, .14064, .14826, .1563, .1648, .17376, .18323, .19324, .2038, .21496, .22674, .23919, .25234, .26624, .28093, .29646, .31287, .33021, .34855, .36794, .38844, .41012, .2347, .2348, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .
01178
01179
01180
01182
01183
                           .43305, .45731, .48297, .51011, .53884, .56924, .60141, .63547,
                           .67152, .70969, .75012, .79292, .83826, .8863, .93718, .99111, 1.0482, 1.1088, 1.173, 1.2411, 1.3133, 1.3898, 1.471, 1.5571, 1.6485, 1.7455, 1.8485, 1.9577, 2.0737, 2.197, 2.3278, 2.4668,
01184
01185
01186
                           2.6145, 2.7715, 2.9383, 3.1156, 3.3042, 3.5047, 3.7181, 3.9451,
                            4.1866, 4.4437, 4.7174, 5.0089, 5.3192, 5.65, 6.0025, 6.3782,
01188
                          6.7787, 7.206, 7.6617, 8.1479, 8.6669, 9.221, 9.8128, 10.445, 11.12, 11.843, 12.615, 13.441, 14.325, 15.271, 16.283, 17.367
01189
01190
01191
                           18.529, 19.776, 21.111, 22.544, 24.082, 25.731, 27.504, 29.409,
                           31.452, 33.654, 36.024, 38.573, 41.323, 44.29, 47.492, 50.951, 54.608, 58.588, 62.929, 67.629, 72.712, 78.226, 84.207, 90.699,
01192
01193
                           97.749, 105.42, 113.77, 122.86, 132.78, 143.61, 155.44, 168.33,
                           182.48, 198.01, 214.87, 233.39, 253.86, 276.34, 300.3, 327.28,
01195
01196
                           356.89, 389.48, 422.29, 458.99, 501.39, 548.13, 595.62, 652.74,
                           716.54, 784.57, 866.78, 960.59, 1062.8, 1072.5, 1189.5, 1319.4, 1467.6, 1630.2, 1813.7, 2016.9, 2253., 2515.3, 2773.5, 3092.8, 3444.4, 3720.4, 4104.3, 4527.5, 4645.9, 5021.7, 5462.2, 5597.,
01197
01198
01199
                           6110.6, 6732.5, 7513.8, 8270.6, 9640.6, 11487., 2796.1, 2680.1,
                           2441.6, 2404.2, 2334.8, 2215.2, 1642.5, 1477.9, 1328.1, 1223.5,
01201
01202
                           843.34, 766.96, 831.65, 834.84, 774.85, 1156.3, 1275.6, 1366.1,
                           1795.6, 1885., 1936.5, 1953.4, 2154.4, 2002.7, 1789.8, 10381., 9040., 8216.5, 7384.7, 6721.9, 6187.7, 6143.8, 5703.9, 5276.6, 4873.1, 4736., 4325.3, 3927., 3554.1, 3286.1, 2950.1, 2642.4, 2368.7, 2138.9, 1914., 1719.6, 1543.9, 1388.6, 1252.1, 1132.2,
01203
01204
01205
01207
                            1024.1, 1025.4, 920.58, 829.59, 750.54, 685.01, 624.25, 570.14,
                           525.81, 481.85, 441.95, 408.71, 377.23, 345.86, 318.51, 292.26,
01208
01209
                           268.34, 247.04, 227.14, 209.02, 192.69, 177.59, 163.78, 151.26,
                           139.73, 129.19, 119.53, 110.7, 102.57, 95.109, 88.264, 81.948, 76.13, 70.768, 65.827, 61.251, 57.022, 53.495, 49.824, 46.443, 43.307, 40.405, 37.716, 35.241, 32.923, 30.77, 28.78, 26.915,
01210
01211
                           25.177, 23.56, 22.059, 20.654, 19.345, 18.126, 16.988, 15.93,
01213
01214
                           14.939, 14.014, 13.149, 12.343, 11.589, 10.884, 10.225, 9.6093,
01215
                           9.0327, 8.4934, 7.9889, 7.5166, 7.0744, 6.6604, 6.2727, 5.9098,
                          5.5701, 5.2529, 4.955, 4.676, 4.4148, 4.171, 3.9426, 3.7332, 3.5347, 3.3493, 3.1677, 3.0025, 2.8466, 2.6994, 2.5601, 2.4277, 2.3016, 2.1814, 2.0664, 1.9564, 1.8279, 1.7311, 1.6427, 1.5645,
01216
01217
01218
                           1.4982, 1.443, 1.374, 1.3146, 1.2562, 1.17, 1.1105, 1.0272,
                           .96863, .89718, .83654, .80226, .75908, .72431, .69573, .67174,
01220
01221
                           .65126, .63315, .61693, .60182, .58715, .59554, .57649, .55526,
                           .53177, .50622, .48176, .4813, .47642, .47492, .50273, .50293, .52687, .52239, .53419, .53814, .52626, .52211, .51492, .50622,
01222
01223
                           .49746, .48841, .4792, .43534, .41999, .40349, .38586, .36799, .35108, .31089, .30803, .3171, .33599, .35041, .36149, .32924, .32462, .27309, .25961, .20922, .19504, .15683, .13098, .11588,
01224
01226
                           .11478, .11204, .11363, .12135, .16423, .17785, .19094, .20236,
01227
                          .11476, .11204, .11363, .12135, .16425, .17785, .19094, .20236, .21084, .2154, .24108, .22848, .20871, .18797, .17963, .17834, .21552, .22284, .26945, .27052, .30108, .28977, .29772, .29224, .27658, .24956, .22777, .20654, .18392, .16338, .1452, .12916, .1152, .10304, .092437, .083163, .075031, .067878, .061564, .055976, .051018, .046609, .042679, .03917, .036032, .033223, .030706, .02845, .026428, .024617, .022998, .021554, .02027, .010136, .01841, .013738, .016541, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026, .015026,
01228
01229
01230
01232
01233
01234
                            .019136, .018141, .017278, .016541, .015926, .015432, .015058,
                          .014807, .014666, .014635, .014728, .014947, .01527, .015728, .016345, .017026, .017798, .018839, .019752, .020636, .021886, .022695, .02327, .023478, .024292, .023544, .022222, .021932, .020052, .018143, .017722, .017031, .017782, .01938, .020734,
01235
01236
                           .020476, .019255, .017477, .016878, .014617, .012489, .011765,
01239
                           .0099077, .0086446, .0079446, .0078644, .0079763, .008671,
01240
                           .01001, .0108, .012933, .015349, .016341, .018484, .020254, .020254, .020254, .020478, .019591, .018595, .018385, .019913, .022254,
01241
01242
                           .024847, .025809, .028053, .029924, .030212, .031367, .03222, .032739, .032537, .03286, .033344, .033507, .033499, .033339, .032809, .033041, .031723, .029837, .027511, .026603, .024032,
01243
01244
01245
01246
                           .021914, .020948, .021701, .023425, .024259, .024987, .023818,
01247
                            .021768, .019223, .018144, .015282, .012604, .01163, .0097907,
                           .008336, .0082473, .0079582, .0088077, .009779, .010129, .012145, .014378, .016761, .01726, .018997, .019998, .019809, .01819, .016358, .016099, .01617, .017939, .020223, .022521, .02277,
01248
01249
                           .024279, .025247, .024222, .023989, .023224, .021493, .020362, .018596, .017309, .015975, .014466, .013171, .011921, .01078,
01251
01252
01253
                            .0097229, \ .0087612, \ .0078729, \ .0070682, \ .0063494, \ .0057156,
                           .0051459, .0046273, .0041712, .0037686, .0034119, .003095, .0028126, .0025603, .0023342, .0021314, .0019489, .0017845,
01254
01255
```

```
.001636, .0015017, .00138, .0012697, .0011694, .0010782,
                9.9507e-4, 9.1931e-4, 8.5013e-4, 7.869e-4, 7.2907e-4, 6.7611e-4, 6.2758e-4, 5.8308e-4, 5.4223e-4, 5.0473e-4, 4.7027e-4, 4.3859e-4,
01257
01258
01259
                4.0946e-4, 3.8265e-4, 3.5798e-4, 3.3526e-4, 3.1436e-4, 2.9511e-4,
                2.7739e-4, 2.6109e-4, 2.4609e-4, 2.3229e-4, 2.1961e-4, 2.0797e-4, 1.9729e-4, 1.875e-4, 1.7855e-4, 1.7038e-4, 1.6294e-4, 1.5619e-4,
01260
01261
                1.5007e-4, 1.4456e-4, 1.3961e-4, 1.3521e-4, 1.3131e-4, 1.2789e-4,
                1.2494e-4, 1.2242e-4, 1.2032e-4, 1.1863e-4, 1.1733e-4, 1.1641e-4,
01263
                1.1585e-4, 1.1565e-4, 1.158e-4, 1.1629e-4, 1.1712e-4, 1.1827e-4,
01264
01265
                1.1976e-4, 1.2158e-4, 1.2373e-4, 1.262e-4, 1.2901e-4, 1.3214e-4,
                1.3562e-4, 1.3944e-4, 1.4361e-4, 1.4814e-4, 1.5303e-4, 1.5829e-4,
01266
                1.6394e-4, 1.6999e-4, 1.7644e-4, 1.8332e-4, 1.9063e-4, 1.984e-4,
01267
                2.0663e-4, 2.1536e-4, 2.246e-4, 2.345e-4, 2.4468e-4, 2.5558e-4, 2.6708e-4, 2.7921e-4, 2.92e-4, 3.0548e-4, 3.1968e-4, 3.3464e-4,
01268
01269
01270
                3.5039e-4, 3.6698e-4, 3.8443e-4, 4.0281e-4, 4.2214e-4, 4.4248e-4,
               4.6389e-4, 4.864e-4, 5.1009e-4, 5.3501e-4, 5.6123e-4, 5.888e-4, 6.1781e-4, 6.4833e-4, 6.8043e-4, 7.142e-4, 7.4973e-4, 7.8711e-4, 8.2644e-4, 8.6783e-4, 9.1137e-4, 9.5721e-4, .0010054, .0010562,
01271
01272
01273
               .0011096, .0011659, .0012251, .0012875, .0013532, .0014224, .0014953, .001572, .0016529, .0017381, .0018279, .0019226, .0020224, .0021277, .0022386, .0023557, .0024792, .0026095,
01275
01276
01277
                .002747,\ .0028921,\ .0030453,\ .0032071,\ .003378,\ .0035586,
                .0037494, .003951, .0041642, .0043897, .0046282, .0048805, .0051476, .0054304, .00573, .0060473, .0063837, .0067404,
01278
01279
               .0071188, .0075203, .0079466, .0083994, .0088806, .0093922, .0099366, .010516, .011134, .011792, .012494, .013244, .014046, .014898, .015808, .016781, .017822, .018929, .020108, .02138,
01280
01282
               .014898, .015808, .016781, .017822, .018929, .020108, .02138, .022729, .02419, .02576, .027412, .029233, .031198, .0333301, .035594, .038092, .040767, .04372, .046918, .050246, .053974, .058009, .061976, .066586, .071537, .076209, .081856, .087998, .093821, .10113, .10913, .11731, .12724, .13821, .15025, .1639, .17807, .19472, .21356, .23496, .25758, .28387, .31389, .34104, .37469, .40989, .43309, .46845, .5042, .5023, .52981, .55275,
01283
01284
01285
01286
01288
01289
                .51075, .51976, .52457, .44779, .44721, .4503, .4243, .45244,
01290
                .49491, .55399, .39021, .24802, .2501, .2618, .27475, .28879,
                . 31317, . 33643, . 36257, . 4018, . 43275, . 46525, . 53333, . 56599, . 60557, . 70142, . 74194, . 77736, . 88567, . 91182, . 93294, . 98407, . 98772, . 99176, . 9995, 1.2405, 1.3602, 1.338, 1.3255, 1.3267, 1.3404, 1.3634, 1.3967, 1.4407, 1.4961, 1.5603, 1.6328, 1.7153,
01291
01292
01294
                1.8094, 1.9091, 2.018, 2.1367, 2.264, 2.4035, 2.5562, 2.7179,
01295
01296
                2.9017, 3.1052, 3.3304, 3.5731, 3.8488, 4.1553, 4.4769, 4.7818,
                5.1711, 5.5204, 5.9516, 6.4097, 6.8899, 7.1118, 7.5469, 7.9735, 7.9511, 8.3014, 8.6418, 8.4757, 8.8256, 9.2294, 9.6923, 10.033, 10.842, 11.851, 11.78, 8.8435, 9.1381, 9.5956, 10.076, 10.629,
01297
01298
                11.22, 11.883, 12.69, 13.163, 13.974, 14.846, 16.027, 17.053,
01300
01301
                18.148, 19.715, 20.907, 22.163, 23.956, 25.235, 26.566, 27.94
01302
                29.576, 30.956, 32.432, 35.337, 39.911, 41.128, 42.625, 44.386,
                46.369, 48.619, 51.031, 53.674, 56.825, 59.921, 63.286, 66.929, 70.859, 75.081, 79.618, 84.513, 89.739, 95.335, 101.35, 107.76,
01303
01304
                114.63, 121.98, 129.87, 138.3, 147.34, 157.04, 167.56, 178.67,
01305
                190.61, 203.43, 217.19, 231.99, 247.88, 264.98, 283.37, 303.17,
                324.49, 347.47, 372.25, 398.98, 427.85, 459.06, 492.8, 529.31,
01307
01308
                568.89, 611.79, 658.35, 708.91, 763.87, 823.65, 888.72, 959.58,
               1036.8, 1121.8, 1213.9, 1314.3, 1423.8, 1543., 1672.8, 1813.4, 1966.1, 2131.4, 2309.5, 2499.3, 2705., 2925.7, 3161.6, 3411.3, 3611.5, 3889.2, 4191.1, 4519.3, 4877.9, 5272.9, 5712.9, 6142.7 6719.6, 7385., 8145., 8977.7, 9831.9, 10827., 11934., 13063.,
01309
01310
01311
                14434., 15878., 17591., 19435., 21510., 23835., 26835., 29740.,
01313
                32878., 36305., 39830., 43273., 46931., 50499., 49586., 51598.,
01314
01315
                53429., 54619., 55081., 55102., 54485., 53487., 52042., 42689.,
                42607., 44020., 47994., 54169., 53916., 55808., 56642., 46049.,
01316
                44243., 32929., 30658., 21963., 20835., 15962., 13679., 17652., 19680., 22388., 25625., 29184., 32520., 35720., 38414., 40523.,
01317
                49228., 48173., 45678., 41768., 37600., 41313., 42654., 44465., 55736., 56630., 65409., 63308., 66572., 61845., 60379., 56777.,
01320
01321
                51920., 46601., 41367., 36529., 32219., 28470., 25192., 22362.
               19907., 17772., 15907., 14273., 12835., 11567., 10445., 9450.2, 8565.1, 7776., 7070.8, 6439.2, 5872.3, 5362.4, 4903., 4488.3, 4113.4, 3773.8, 3465.8, 3186.1, 2931.7, 2700.1, 2488.8, 2296., 2119.8, 1958.6, 1810.9, 1675.6, 1551.4, 1437.3, 1332.4, 1236.,
01322
01323
01324
                1147.2, 1065.3, 989.86, 920.22, 855.91, 796.48, 741.53, 690.69, 643.62, 600.02, 559.6, 522.13, 487.35, 455.06, 425.08, 397.21,
01326
01327
                371.3, 347.2, 324.78, 303.9, 284.46, 266.34, 249.45, 233.7, 219.01, 205.3, 192.5, 180.55, 169.38, 158.95, 149.2, 140.07,
01328
01329
                131.54, 123.56, 116.09, 109.09, 102.54, 96.405, 90.655, 85.266, 80.213, 75.475, 71.031, 66.861, 62.948, 59.275, 55.827, 52.587,
01330
01331
                49.544, 46.686, 43.998, 41.473, 39.099, 36.867, 34.768, 32.795,
01332
01333
                30.939, 29.192, 27.546, 25.998, 24.539, 23.164, 21.869, 20.65,
                19.501, 18.419, 17.399, 16.438, 15.532, 14.678, 13.874, 13.115,
01334
                12.4, 11.726, 11.088, 10.488, 9.921, 9.3846, 8.8784, 8.3996, 7.9469, 7.5197, 7.1174, 6.738, 6.379, 6.0409, 5.7213, 5.419, 5.1327, 4.8611, 4.6046, 4.3617, 4.1316, 3.9138, 3.7077, 3.5125,
01335
01336
                3.3281, 3.1536, 2.9885, 2.8323, 2.6846, 2.5447, 2.4124, 2.2871,
01338
01339
                2.1686, 2.0564, 1.9501, 1.8495, 1.7543, 1.6641, 1.5787, 1.4978,
01340
                1.4212, 1.3486, 1.2799, 1.2147, 1.1529, 1.0943, 1.0388, .98602,
                .93596, .8886, .84352, .80078, .76029, .722, .68585, .65161, .61901, .58808, .55854, .53044, .5039, .47853, .45459, .43173,
01341
01342
```

```
01343
                     .41008, .38965, .37021, .35186, .33444, .31797, .30234, .28758,
                     .2736, .26036, .24764, .2357, .22431, .21342, .20295, .19288, .18334, .17444, .166, .15815, .15072, .14348, .13674, .13015,
01344
01345
                     12399, 11807, 11231, 10689, 10164, 096696, 091955, 087476, 083183, 079113, 075229, 071536, 068026, 064698, 06154, 058544, 055699, 052997, 050431, 047993, 045676, 043475, 041382, 039392, 037501, 035702, 033991, 032364, 030817, 029345, 027945, 026613, 025345, 024139, 022991,
01346
01347
01348
01350
                     .03081/, .029345, .027945, .026615, .025345, .024139, .022991, .021899, .02086, .019871, .018929, .018033, .01718, .016368, .015595, .014859, .014158, .013491, .012856, .012251, .011675, .011126, .010604, .010107, .0096331, .009182, .0087523, .0083431, .0079533, .0075821, .0072284, .0068915, .0065706, .0062649, .0059737, .0056963, .005432, .0051802, .0049404, .0047118, .0044941, .0042867, .0040891, .0039009, .0037216, .0035507, .003388, .0032329, .0030852, .0029445, .0028105, .0026829, .0025613, .0024455, .0023353, .0023304, .0020353
01351
01352
01353
01354
01356
01357
                     .0025613, .0024455, .0023353, .0022303, .0021304, .0020353, .0019448, .0018587, .0017767, .0016988, .0016247, .0015543,
01358
01359
                     .0014874, .0014238, .0013635, .0013062, .0012519, .0012005, .0011517, .0011057, .0010621, .001021, 9.8233e-4, 9.4589e-4,
01360
                      9.1167e-4, 8.7961e-4, 8.4964e-4, 8.2173e-4, 7.9582e-4, 7.7189e-4,
01362
                      7.499e-4, 7.2983e-4, 7.1167e-4, 6.9542e-4, 6.8108e-4, 6.6866e-4,
01363
01364
                      6.5819e-4, 6.4971e-4, 6.4328e-4, 6.3895e-4, 6.3681e-4, 6.3697e-4,
                     6.3956e-4, 6.4472e-4, 6.5266e-4, 6.6359e-4, 6.78e-4, 6.9563e-4, 7.1749e-4, 7.4392e-4, 7.7556e-4, 8.1028e-4, 8.4994e-4, 8.8709e-4, 9.3413e-4, 9.6953e-4, .0010202, .0010738, .0010976, .0011507, .0011686, .0012264, .001291, .0013346, .0014246, .0015293, .0016359, .0017824, .0019255, .0020854, .002247, .0024148,
01365
01366
01367
01368
01369
01370
                      .0026199, .0027523, .0029704, .0030702, .0033047, .0035013,
01371
                      .0037576, .0040275, .0043089, .0046927, .0049307, .0053486,
                     .0053809, .0056699, .0059325, .0055488, .005634, .0056392, .004946, .0048855, .0048208, .0044386, .0045498, .0046377, .0048939, .0052396, .0057324, .0060859, .0066906, .0071148,
01372
01373
                     .0077224, .0082687, .008769, .0084471, .008572, .0087729, .008775, .0090742, .0080704, .0080288, .0085747, .0086087,
01375
01376
                     .0086408, .0088752, .0089381, .0089757, .0093532, .0092824, .0092566, .0092645, .0092735, .009342, .0095806, .0097991,
01377
01378
                     .010213, .010611, .011129, .011756, .013237, .01412, .015034, .015936, .01682, .018597, .019315, .019995, .020658, .021289, .022363, .022996, .023716, .024512, .025434, .026067, .027118,
01379
01381
                     .028396, .029865, .031442, .033253, .03525, .037296, .039701, .042356, .045154, .048059, .051294, .054893, .058636, .061407
01382
01383
                     .065172, .068974, .072676, .073379, .076547, .079556, .079134, .082308, .085739, .090192, .09359, .099599, .10669, .11496,
01384
01385
                     .1244, .13512, .14752, .14494, .15647, .1668, .17863, .19029, .20124, .20254, .21179, .21982, .21625, .22364, .23405, .23382, .2434, .25708, .26406, .27621, .28909, .30395, .31717, .33271,
01386
01387
01388
01389
                      .3496, .36765, .38774, .40949, .446, .46985, .49846, .5287, .562,
                     59841, .64598, .68834, .7327, .78978, .8373, .88708, .94744, 1.0006, 1.0574, 1.1215, 1.1856, 1.2546, 1.3292, 1.4107, 1.4974, 1.5913, 1.6931, 1.8028, 1.9212, 2.0492, 2.1874, 2.3365, 2.4978, 2.6718, 2.8588, 3.062, 3.2818, 3.5188, 3.7752, 4.0527, 4.3542, 4.6782, 5.0312, 5.4123, 5.8246, 6.2639, 6.7435, 7.2636, 7.8064,
01390
01391
01392
01394
01395
                     8.4091, 9.0696, 9.7677, 10.548, 11.4, 12.309, 13.324, 14.284,
                     15.445, 16.687, 18.019, 19.403, 20.847, 22.366, 23.925, 25.537, 27.213, 28.069, 29.864, 31.829, 33.988, 35.856, 38.829, 42.321, 46.319, 50.606, 55.126, 59.126, 64.162, 68.708, 74.615, 81.176, 87.739, 95.494, 103.83, 113.38, 123.99, 135.8, 148.7, 162.58,
01396
01397
01398
                     176.32, 192.6, 211.47, 232.7, 252.64, 277.41, 305.38, 333.44,
01400
                     366.42, 402.66, 442.14, 484.53, 526.42, 568.15, 558.78, 582.6, 600.98, 613.94, 619.44, 618.24, 609.84, 595.96, 484.86, 475.59,
01401
01402
                     478.49, 501.56, 552.19, 628.44, 630.39, 658.92, 671.96, 562.7, 545.88, 423.43, 400.14, 306.59, 294.13, 246.8, 226.51, 278.21, 314.39, 347.22, 389.13, 433.16, 477.48, 521.67, 560.54, 683.6,
01403
01404
                     696.37, 695.91, 683.1, 658.24, 634.89, 698.85, 742.87, 796.66, 954.49, 1009.5, 1150.5, 1179.1, 1267.9, 1272.4, 1312.7, 1330.4,
01406
01407
01408
                     1331.6, 1315.8, 1308.3, 1293.3, 1274.6, 1249.5, 1213.2, 1172.1,
01409
                     1124.4, 930.33, 893.36, 871.27, 883.54, 940.76, 1036., 1025.6, 1053.1, 914.51, 894.15, 865.03, 670.63, 508.41, 475.15, 370.85,
01410
                     361.06, 319.38, 312.75, 331.87, 367.13, 415., 467.94, 525.49, 578.41, 624.66, 794.82, 796.97, 780.29, 736.49, 670.18, 603.75, 659.67, 679.8, 857.12, 884.05, 900.65, 1046.1, 1141.9, 1083., 1089.2, 1e3, 947.08, 872.31, 787.91, 704.75, 624.93, 553.68,
01411
01413
01414
                     489.91, 434.21, 385.64, 343.3, 306.42, 274.18, 245.94, 221.11, 199.23, 179.88, 162.73, 147.48, 133.88, 121.73, 110.86, 101.1, 92.323, 84.417, 77.281, 70.831, 64.991, 59.694, 54.884, 50.509, 46.526, 42.893, 39.58, 36.549, 33.776, 31.236, 28.907, 26.77,
01415
01416
01417
                     24.805, 23., 21.339, 19.81, 18.404, 17.105, 15.909, 14.801,
01419
                     13.778, 12.83, 11.954, 11.142, 10.389, 9.691, 9.0434, 8.4423, 7.8842, 7.3657, 6.8838, 6.4357, 6.0189, 5.6308, 5.2696, 4.9332,
01420
01421
                     4.6198, 4.3277, 4.0553, 3.8012, 3.5639, 3.3424, 3.1355, 2.9422, 2.7614, 2.5924, 2.4343, 2.2864, 2.148, 2.0184, 1.8971, 1.7835, 1.677, 1.5773, 1.4838, 1.3961, 1.3139, 1.2369, 1.1645, 1.0966,
01422
01423
                     1.0329, .97309, .91686, .86406, .81439, .76767, .72381, .68252, .64359, .60695, .57247, .54008, .50957, .48092, .45401, .42862,
01425
01426
                     .40465, .38202, .36072, .34052, .3216, .30386, .28711, .27135, .25651, .24252, .2293, .21689, .20517, .19416, .18381, .17396,
01427
01428
                      .16469
```

```
01430
01431
01432
             static double co2230[2001] = {2.743e-5, 2.8815e-5, 3.027e-5, 3.1798e-5,}
01433
                3.3405e-5, 3.5094e-5, 3.6869e-5, 3.8734e-5, 4.0694e-5, 4.2754e-5,
                4.492e-5, 4.7196e-5, 4.9588e-5, 5.2103e-5, 5.4747e-5, 5.7525e-5, 6.0446e-5, 6.3516e-5, 6.6744e-5, 7.0137e-5, 7.3704e-5, 7.7455e-5,
01434
01435
                8.1397e-5, 8.5543e-5, 8.9901e-5, 9.4484e-5, 9.9302e-5, 1.0437e-4,
                1.097e-4, 1.153e-4, 1.2119e-4, 1.2738e-4, 1.3389e-4, 1.4074e-4,
01438
                1.4795e-4, 1.5552e-4, 1.6349e-4, 1.7187e-4, 1.8068e-4, 1.8995e-4,
                1.997e-4, 2.0996e-4, 2.2075e-4, 2.321e-4, 2.4403e-4, 2.5659e-4, 2.698e-4, 2.837e-4, 2.9832e-4, 3.137e-4, 3.2988e-4, 3.4691e-4,
01439
01440
                3.6483e-4, 3.8368e-4, 4.0351e-4, 4.2439e-4, 4.4635e-4, 4.6947e-4,
01441
                4.9379e-4, 5.1939e-4, 5.4633e-4, 5.7468e-4, 6.0452e-4,
01442
                6.69e-4, 7.038e-4, 7.4043e-4, 7.79e-4, 8.1959e-4, 8.6233e-4,
01443
01444
                9.0732e-4, 9.5469e-4, .0010046, .0010571, .0011124, .0011706
01445
                .0012319, .0012964, .0013644, .001436, .0015114, .0015908,
                .0016745, .0017625, .0018553, .0019531, .002056, .0021645, .0022788, .0023992, .002526, .0026596, .0028004, .0029488, .0031052, .0032699, .0034436, .0036265, .0038194, .0040227, .0042369, .0044628, .0047008, .0049518, .0052164, .0054953,
01446
01447
                .0057894, .0060995, .0064265, .0067713, .007135, .0075184, .0079228, .0083494, .0087993, .0092738, .0097745, .010303,
01450
01451
                .01086, .011448, .012068, .012722, .013413, .014142, .014911, .015723, .01658, .017484, .018439, .019447, .020511, .021635, .022821, .024074, .025397, .026794, .02827, .029829, .031475, .033215, .035052, .036994, .039045, .041213, .043504, .045926, .048485, .05119, .05405, .057074, .060271, .063651, .067225,
01452
01453
01454
01456
01457
                .071006, .075004, .079233, .083708, .088441, .093449, .098749,
                .07304, .07304, .079233, .003706, .003441, .093449, .093749, .10436, .11029, .11657, .12322, .13026, .13772, .14561, .15397, .16282, .1722, .18214, .19266, .20381, .21563, .22816, .24143, .2555, .27043, .28625, .30303, .32082, .3397, .35972, .38097, .40352, .42746, .45286, .47983, .50847, .53888, .57119, .6055, .64196, .6807, .72187, .76564, .81217, .86165, .91427, .97025,
01458
01459
01460
01462
01463
                1.0298, 1.0932, 1.1606, 1.2324, 1.3088, 1.3902, 1.477, 1.5693,
                1.6678, 1.7727, 1.8845, 2.0038, 2.131, 2.2666, 2.4114, 2.5659, 2.7309, 2.907, 3.0951, 3.2961, 3.5109, 3.7405, 3.986, 4.2485, 4.5293, 4.8299, 5.1516, 5.4961, 5.8651, 6.2605, 6.6842, 7.1385, 7.6256, 8.1481, 8.7089, 9.3109, 9.9573, 10.652, 11.398, 12.2, 13.063, 13.992, 14.99, 16.064, 17.222, 18.469, 19.813, 21.263,
01464
01465
01466
                22.828, 24.516, 26.34, 28.31, 30.437, 32.738, 35.226, 37.914,
01469
01470
                40.824, 43.974, 47.377, 51.061, 55.011, 59.299, 63.961, 69.013,
                74.492, 80.444, 86.919, 93.836, 101.23, 109.25, 117.98, 127.47,
01471
                137.81, 149.07, 161.35, 174.75, 189.42, 205.49, 223.02, 242.26, 263.45, 286.75, 311.94, 340.01, 370.86, 404.92, 440.44, 480.27, 525.17, 574.71, 626.22, 686.8, 754.38, 827.07, 913.38, 1011.7,
01472
01473
01474
01475
                1121.5, 1161.6, 1289.5, 1432.2, 1595.4, 1777., 1983.3, 2216.1,
01476
                2485.7, 2788.3, 3101.5, 3481., 3902.1, 4257.1, 4740., 5272.8,
                5457.9, 5946.2, 6505.3, 6668.4, 7302.4, 8061.6, 9015.8, 9908.3, 11613., 13956., 3249.6, 3243., 2901.5, 2841.3, 2729.6, 2558.2, 1797.8, 1583.2, 1386., 1233.5, 787.74, 701.46, 761.66, 767.21,
01477
01478
01479
                 722.83, 1180.6, 1332.1, 1461.6, 2032.9, 2166., 2255.9, 2294.7,
                2587.2, 2396.5, 2122.4, 12553., 10784., 9832.5, 8827.3, 8029.1
01481
01482
                7377.9, 7347.1, 6783.8, 6239.1, 5721.1, 5503., 4975.1, 4477.8,
01483
                4021.3, 3676.8, 3275.3, 2914.9, 2597.4, 2328.2, 2075.4, 1857.6,
                1663.6, 1493.3, 1343.8, 1213.3, 1095.6, 1066.5, 958.91, 865.15,
01484
                783.31, 714.35, 650.77, 593.98, 546.2, 499.9, 457.87, 421.75, 387.61, 355.25, 326.62, 299.7, 275.21, 253.17, 232.83, 214.31, 197.5, 182.08, 167.98, 155.12, 143.32, 132.5, 122.58, 113.48,
01485
01487
                105.11, 97.415, 90.182, 83.463, 77.281, 71.587, 66.341, 61.493,
01488
                57.014, 53.062, 49.21, 45.663, 42.38, 39.348, 36.547, 33.967,
01489
                31.573, 29.357, 27.314, 25.415, 23.658, 22.03, 20.524, 19.125,
01490
                17.829, 16.627, 15.511, 14.476, 13.514, 12.618, 11.786, 11.013, 10.294, 9.6246, 9.0018, 8.4218, 7.8816, 7.3783, 6.9092, 6.4719,
01491
01492
                6.0641, 5.6838, 5.3289, 4.998, 4.6893, 4.4014, 4.1325, 3.8813, 3.6469, 3.4283, 3.2241, 3.035, 2.8576, 2.6922, 2.5348, 2.3896,
01494
01495
                2.2535, 2.1258, 2.0059, 1.8929, 1.7862, 1.6854, 1.5898, 1.4992,
                1.4017, 1.3218, 1.2479, 1.1809, 1.1215, 1.0693, 1.0116, .96016, .9105, .84859, .80105, .74381, .69982, .65127, .60899, .57843, .54592, .51792, .49336, .47155, .45201, .43426, .41807, .40303,
01496
01497
01498
                .38876, .3863, .37098, .35492, .33801, .32032, .30341, .29874, .29193, .28689, .29584, .29155, .29826, .29195, .29287, .2904, .28199, .27709, .27162, .26622, .26133, .25676, .25235, .23137
01500
01501
01502
                 .22365, .21519, .20597, .19636, .18699, .16485, .16262, .16643,
                .17542, .18198, .18631, .16759, .16338, .13505, .1267, .10053,
01503
                .092554, .074093, .062159, .055523, .054849, .05401, .05528, .058982, .07952, .08647, .093244, .099285, .10393, .10661,
01504
                .12072, .11417, .10396, .093265, .089137, .088909,
01506
                                                                                                       .10902,
                .11277, .13625, .13565, .14907, .14167, .1428, .13744, .12768, .11382, .10244, .091686, .08109, .071739, .063616, .056579, .050504, .045251, .040689, .036715, .033237, .030181, .027488,
01507
01508
01509
                .025107, .022998, .021125, .01946, .017979, .016661, .015489, .014448, .013526, .012712, .011998, .011375, .010839, .010384
01510
                .010007, .0097053, .0094783, .0093257, .0092489, .0092504, .0093346, .0095077, .0097676, .01012, .01058, .011157, .011844,
01512
01513
01514
                 .012672, .013665, .014766, .015999, .017509, .018972, .020444,
                .02311, .023742, .0249, .025599, .026981, .026462, .025143, .025066, .022814, .020458, .020026, .019142, .020189, .022371,
01515
01516
```

```
.024163, .023728, .02199, .019506, .018591, .015576, .012784,
                .011744, .0094777, .0079148, .0070652, .006986, .0071758, .008086, .0098025, .01087, .013609, .016764, .018137, .021061,
01518
01519
01520
                 .023498, .023576, .023965, .022828, .021519, .021283, .023364,
                .026457, .029782, .030856, .033486, .035515, .035543, .036558, .037198, .037472, .037045, .037284, .03777, .038085, .038366, .038526, .038282, .038915, .037697, .035667, .032941, .031959, .028692, .025918, .024596, .025592, .027873, .028935, .02984,
01521
01524
01525
                 .028148, .025305, .021912, .020454, .016732, .013357, .01205,
                .028148, .025305, .021912, .020454, .016/32, .01335/, .01205, .009731, .0079881, .0077704, .0074387, .0083895, .0096776, .010326, .01293, .015955, .019247, .020145, .02267, .024231, .024184, .022131, .019784, .01955, .01971, .022119, .025116, .027978, .028107, .029808, .030701, .029164, .028551, .027286, .024946, .023259, .020982, .019221, .017471, .015643, .014074, .01261, .011301, .010116, .0090582, .0081036, .0072542, .0065034, .0052426, .0052571, .0047321, .004787, .0032607, .0034077
01526
01527
01528
01530
01531
                 .0058436, .0052571, .0047321, .0042697, .0038607, .0034977, .0031747, .0028864, .0026284, .002397, .002189, .0020017,
01532
01533
                 .0018326, .0016798, .0015414, .0014159, .0013019, .0011983, .0011039, .0010177, 9.391e-4, 8.6717e-4, 8.0131e-4, 7.4093e-4,
01534
                 6.8553e-4, 6.3464e-4, 5.8787e-4, 5.4487e-4, 5.0533e-4, 4.69e-4,
01536
                 4.3556e-4, 4.0474e-4, 3.7629e-4, 3.5e-4, 3.2569e-4, 3.032e-4,
01537
01538
                 2.8239e-4, 2.6314e-4, 2.4535e-4, 2.2891e-4, 2.1374e-4, 1.9975e-4,
                 1.8685e-4, 1.7498e-4, 1.6406e-4, 1.5401e-4, 1.4479e-4, 1.3633e-4,
01539
                 1.2858e-4, 1.2148e-4, 1.1499e-4, 1.0907e-4, 1.0369e-4, 9.8791e-5,
01540
                 9.4359e-5, 9.0359e-5, 8.6766e-5, 8.3555e-5, 8.0703e-5, 7.8192e-5,
                 7.6003e-5, 7.4119e-5, 7.2528e-5, 7.1216e-5, 7.0171e-5, 6.9385e-5,
                 6.8848e-5, 6.8554e-5, 6.8496e-5, 6.8669e-5, 6.9069e-5, 6.9694e-5,
01543
01544
                 7.054e-5, 7.1608e-5, 7.2896e-5, 7.4406e-5, 7.6139e-5, 7.8097e-5,
                 8.0283e-5, 8.2702e-5, 8.5357e-5, 8.8255e-5, 9.1402e-5, 9.4806e-5, 9.8473e-5, 1.0241e-4, 1.0664e-4, 1.1115e-4, 1.1598e-4, 1.2112e-4,
01545
01546
01547
                 1.2659e-4, 1.3241e-4, 1.3859e-4, 1.4515e-4, 1.521e-4, 1.5947e-4,
                 1.6728e-4, 1.7555e-4, 1.8429e-4, 1.9355e-4, 2.0334e-4, 2.1369e-4,
                 2.2463e-4, 2.3619e-4, 2.4841e-4, 2.6132e-4, 2.7497e-4, 2.8938e-4,
01549
01550
                 3.0462e-4, 3.2071e-4, 3.3771e-4, 3.5567e-4, 3.7465e-4, 3.947e-4,
                 4.1588e-4, 4.3828e-4, 4.6194e-4, 4.8695e-4, 5.1338e-4, 5.4133e-4, 5.7087e-4, 6.0211e-4, 6.3515e-4, 6.701e-4, 7.0706e-4, 7.4617e-4, 7.8756e-4, 8.3136e-4, 8.7772e-4, 9.2681e-4, 9.788e-4, .0010339,
01551
01552
01553
                .0010922, .001154, .0012195, .0012889, .0013626, .0014407, .0015235, .0016114, .0017048, .0018038, .001909, .0020207,
01555
01556
                 .0021395, .0022657, .0023998, .0025426, .0026944, .002856,
01557
                 .0030281, \ .0032114, \ .0034068, \ .003615, \ .0038371, \ .004074,
                 .004327, .0045971, .0048857, .0051942, .0055239, .0058766, .0062538, .0066573, .0070891, .007551, .0080455, .0085747, .0091412, .0097481, .010397, .011092, .011837, .012638, .013495,
01558
01559
01560
                .014415, .01541, .016475, .017621, .018857, .020175, .02162, .023185, .024876, .02672, .028732, .030916, .033319, .035939
01561
01562
01563
                 .038736, .041847, .04524, .048715, .052678, .056977, .061203,
01564
                 .066184, .07164, .076952, .083477, .090674, .098049, .10697,
                 . 1169, 1277, 14011, 15323, 1684, 18601, 20626, 22831, 25417, 28407, .31405, .34957, .38823, .41923, .46026, .50409, .51227, .54805, .57976, .53818, .55056, .557, .46741, .46403,
01565
01566
                 .4636, .42265, .45166, .49852, .56663, .34306, .17779, .17697
01568
01569
                 .18346, .19129, .20014, .21778, .23604, .25649, .28676, .31238,
                 .33856, .39998, .4288, .46568, .56654, .60786, .64473, .76466, .7897, .80778, .86443, .85736, .84798, .84157, 1.1385, 1.2446, 1.1923, 1.1552, 1.1338, 1.1266, 1.1292, 1.1431, 1.1683, 1.2059,
01570
01571
01572
                 1.2521, 1.3069, 1.3712, 1.4471, 1.5275, 1.6165, 1.7145, 1.8189, 1.9359, 2.065, 2.2007, 2.3591, 2.5362, 2.7346, 2.9515, 3.2021,
01574
                1.5339, 2.063, 2.2007, 2.3391, 2.3392, 2.7346, 2.3313, 3.2021, 3.4851, 3.7935, 4.0694, 4.4463, 4.807, 5.2443, 5.7178, 6.2231, 6.4796, 6.9461, 7.4099, 7.3652, 7.7182, 8.048, 7.7373, 8.0363, 8.3855, 8.8044, 9.0257, 9.8574, 10.948, 10.563, 6.8979, 7.0744, 7.4121, 7.7663, 8.1768, 8.6243, 9.1437, 9.7847, 10.182, 10.849, 11.572, 12.602, 13.482, 14.431, 15.907, 16.983, 18.11, 19.884,
01575
01576
01577
01578
                 21.02, 22.18, 23.355, 24.848, 25.954, 27.13, 30.186, 34.893,
01580
01581
                 35.682, 36.755, 38.111, 39.703, 41.58, 43.606, 45.868, 48.573,
01582
                 51.298, 54.291, 57.559, 61.116, 64.964, 69.124, 73.628, 78.471,
                83.683, 89.307, 95.341, 101.84, 108.83, 116.36, 124.46, 133.18, 142.57, 152.79, 163.69, 175.43, 188.11, 201.79, 216.55, 232.51, 249.74, 268.38, 288.54, 310.35, 333.97, 359.55, 387.26, 417.3, 449.88, 485.2, 523.54, 565.14, 610.28, 659.31, 712.56, 770.43,
01583
01584
01585
                 833.36, 901.82, 976.36, 1057.6, 1146.8, 1243.8, 1350., 1466.3,
01587
                 1593.6, 1732.7, 1884.1, 2049.1, 2228.2, 2421.9, 2629.4, 2853.7,
01588
                 3094.4, 3351.1, 3622.3, 3829.8, 4123.1, 4438.3, 4777.2, 5144.1, 5545.4, 5990.5, 6404.5, 6996.8, 7687.6, 8482.9, 9349.4, 10203.,
01589
01590
                 11223., 12358., 13493., 14916., 16416., 18236., 20222., 22501., 25102., 28358., 31707., 35404., 39538., 43911., 48391., 53193.,
01591
01592
                 58028., 58082., 61276., 64193., 66294., 67480., 67921., 67423.,
01593
01594
                 66254., 64341., 51737., 51420., 53072., 58145., 66195., 65358.,
                 67377., 67869., 53509., 50553., 35737., 32425., 21704., 19974.,
01595
                 14457., 12142., 16798., 19489., 23049., 27270., 31910., 36457., 40877., 44748., 47876., 59793., 58626., 55454., 50337., 44893., 50228., 52216., 54747., 69541., 70455., 81014., 77694., 80533.,
01596
01597
                 73953., 70927., 65539., 59002., 52281., 45953., 40292., 35360., 31124., 27478., 24346., 21647., 19308., 17271., 15491., 13927.,
01599
01600
                 12550., 11331., 10250., 9288.8, 8431.4, 7664.9, 6978.3, 6361.8, 5807.4, 5307.7, 4856.8, 4449., 4079.8, 3744.9, 3440.8, 3164.2, 2912.3, 2682.7, 2473., 2281.4, 2106., 1945.3, 1797.9, 1662.5,
01601
01602
01603
```

```
1538.1, 1423.6, 1318.1, 1221., 1131.5, 1049., 972.99, 902.87,
                 838.01, 777.95, 722.2, 670.44, 622.35, 577.68, 536.21, 497.76, 462.12, 429.13, 398.61, 370.39, 344.29, 320.16, 297.85, 277.2,
01606
01607
                 258.08, 240.38, 223.97, 208.77, 194.66, 181.58, 169.43, 158.15,
                 147.67, 137.92, 128.86, 120.44, 112.6, 105.3, 98.499, 92.166, 86.264, 80.763, 75.632, 70.846, 66.381, 62.213, 58.321, 54.685,
01608
01609
                 51.288, 48.114, 45.145, 42.368, 39.772, 37.341, 35.065, 32.937, 30.943, 29.077, 27.33, 25.693, 24.158, 22.717, 21.367, 20.099,
01610
                 18.909, 17.792, 16.744, 15.761, 14.838, 13.971, 13.157, 12.393, 11.676, 11.003, 10.369, 9.775, 9.2165, 8.6902, 8.1963, 7.7314, 7.2923, 6.8794, 6.4898, 6.122, 5.7764, 5.4525, 5.1484, 4.8611,
01612
01613
01614
                 4.5918, 4.3379, 4.0982, 3.8716, 3.6567, 3.4545, 3.2634, 3.0828,
01615
                 2.9122, 2.7512, 2.5993, 2.4561, 2.3211, 2.1938, 2.0737, 1.9603,
                 1.8534, 1.7525, 1.6572, 1.5673, 1.4824, 1.4022, 1.3265, 1.2551,
01617
01618
                 1.1876, 1.1239, 1.0637, 1.0069, .9532, .90248, .85454, .80921,
                 .76631, .72569, .6872, .65072, .61635, .5836, .55261, .52336, .49581, .46998, .44559, .42236, .40036, .37929, .35924, .34043,
01619
01620
                .49581, .40598, .44559, .42236, .40036, .37929, .35924, .34043, .32238, .30547, .28931, .27405, .25975, .24616, .23341, .22133, .20997, .19924, .18917, .17967, .17075, .16211, .15411, .14646, .13912, .13201, .12509, .11857, .11261, .10698, .10186, .097039, .092236, .087844, .083443, .07938, .075452, .071564, .067931, .064389, .061078, .057901, .054921, .052061, .049364, .046789,
01621
01624
01625
                  .04435, .042044, .039866, .037808, .035863, .034023, .032282,
01626
                .030634, .029073, .027595, .026194, .024666, .023608, .022415, .021283, .02021, .019193, .018228, .017312, .016443, .015619, .014837, .014094, .01339, .012721, .012086, .011483, .010911, .010368, .009852, .0093623, .0088972, .0084556, .0080362,
01627
01628
01629
01630
                 .0076379, .0072596, .0069003, .006559, .0062349, .0059269, .0056344, .0053565, .0050925, .0048417, .0046034, .004377,
01631
01632
                 .0041618, .0039575, .0037633, .0035788, .0034034, .0032368,
01633
                 .0030785, .002928, .0027851, .0026492, .0025201, .0023975, .0022809, .0021701, .0020649, .0019649, .0018699, .0017796, .0016938, .0016122, .0015348, .0014612, .0013913, .001325,
01634
01636
01637
                  .0012619, .0012021, .0011452, .0010913, .0010401, 9.9149e-4
01638
                 9.454e-4, 9.0169e-4, 8.6024e-4, 8.2097e-4, 7.8377e-4, 7.4854e-4,
                 7.1522e-4, 6.8371e-4, 6.5393e-4, 6.2582e-4, 5.9932e-4, 5.7435e-4, 5.5087e-4, 5.2882e-4, 5.0814e-4, 4.8881e-4, 4.7076e-4, 4.5398e-4,
01639
01640
                 4.3843e-4, 4.2407e-4, 4.109e-4, 3.9888e-4, 3.88e-4, 3.7826e-4,
01641
                 3.6963e-4, 3.6213e-4, 3.5575e-4, 3.505e-4, 3.464e-4, 3.4346e-4,
                 3.4173e-4, 3.4125e-4, 3.4206e-4, 3.4424e-4, 3.4787e-4, 3.5303e-4,
01643
01644
                 3.5986e-4, 3.6847e-4, 3.7903e-4, 3.9174e-4, 4.0681e-4, 4.2455e-4,
                 4.4527e-4, 4.6942e-4, 4.9637e-4, 5.2698e-4, 5.5808e-4, 5.9514e-4, 6.2757e-4, 6.689e-4, 7.1298e-4, 7.3955e-4, 7.8403e-4, 8.0449e-4, 8.5131e-4, 9.0256e-4, 9.3692e-4, .0010051, .0010846, .0011678,
01645
01646
01647
                .001282, .0014016, .0015355, .0016764, .0018272, .0020055, .0021455, .0023421, .0024615, .0026786, .0028787, .0031259,
01649
01650
                  .0034046, .0036985, .0040917, .0043902, .0048349, .0049531,
                 .0053044, .0056148, .0052452, .0053357, .005333, .0045069, .0043851, .004253, .003738, .0038084, .0039013, .0041505, .0045372, .0050569, .0054507, .0061267, .0066122, .0072449, .0078012, .0082651, .0076538, .0076573, .0076806, .0075227,
01651
01652
01653
                 .0076269, .0063758, .006254, .0067749, .0067909, .0068231,
01655
01656
                 .0072143, .0072762, .0072954, .007679, .0075107, .0073658,
                .0072441, .0071074, .0070378, .007176, .0072472, .0075844, .0079291, .008412, .0090165, .010688, .011535, .012375, .013166, .013895, .015567, .016011, .016392, .016737, .017043, .017731, .018031, .018419, .018877, .019474, .019868, .020604, .021538, .022653, .023869, .025288, .026879, .028547, .030524, .03274,
01657
01658
01659
01661
                 .035132, .03769, .040567, .043793, .047188, .049962, .053542, .057205, .060776, .061489, .064419, .067124, .065945, .068487
01662
01663
                 .071209, .074783, .077039, .082444, .08902, .09692, .10617, .11687, .12952, .12362, .13498, .14412, .15492, .16519, .1744, .17096, .17714, .18208, .17363, .17813, .18564, .18295, .19045,
01664
01665
                 .20252, .20815, .21844, .22929, .24229, .25321, .26588, .2797, .29465, .31136, .32961, .36529, .38486, .41027, .43694, .4667,
01668
01669
                  .49943, .54542, .58348, .62303, .67633, .71755, .76054, .81371,
                 .85934, .90841, .96438, 1.0207, 1.0821, 1.1491, 1.2226, 1.3018, 1.388, 1.4818, 1.5835, 1.6939, 1.8137, 1.9435, 2.0843, 2.237, 2.4026, 2.5818, 2.7767, 2.9885, 3.2182, 3.4679, 3.7391, 4.0349,
01670
01671
01672
                  4.3554, 4.7053, 5.0849, 5.4986, 5.9436, 6.4294, 6.9598,
                 8.143, 8.8253, 9.5568, 10.371, 11.267, 12.233, 13.31, 14.357, 15.598, 16.93, 18.358, 19.849, 21.408, 23.04, 24.706, 26.409,
01674
01675
                28.153, 28.795, 30.549, 32.43, 34.49, 36.027, 38.955, 42.465, 46.565, 50.875, 55.378, 59.002, 63.882, 67.949, 73.693, 80.095, 86.403, 94.264, 102.65, 112.37, 123.3, 135.54, 149.14, 163.83, 179.17, 196.89, 217.91, 240.94, 264.13, 292.39, 324.83, 358.21,
01676
01677
01678
                 397.16, 440.5, 488.6, 541.04, 595.3, 650.43, 652.03, 688.74,
01680
                 719.47, 743.54, 757.68, 762.35, 756.43, 741.42, 595.43, 580.97, 580.83, 605.68, 667.88, 764.49, 759.93, 789.12, 798.17, 645.66,
01681
01682
                 615.65, 455.05, 421.09, 306.45, 289.14, 235.7, 215.52, 274.57,
01683
                 316.53, 357.73, 409.89, 465.06, 521.84, 579.02, 630.64, 794.46,
01684
                 813., 813.56, 796.25, 761.57, 727.97, 812.14, 866.75, 932.5,
                 1132.8, 1194.8, 1362.2, 1387.2, 1482.3, 1479.7, 1517.9, 1533.1
01687
                 1534.2, 1523.3, 1522.5, 1515.5, 1505.2, 1486.5, 1454., 1412.,
01688
                 1358.8, 1107.8, 1060.9, 1033.5, 1048.2, 1122.4, 1248.9, 1227.1,
                 1255.4, 1058.9, 1020.7, 970.59, 715.24, 512.56, 468.47, 349.3, 338.26, 299.22, 301.26, 332.38, 382.08, 445.49, 515.87, 590.85,
01689
01690
```

```
662.3, 726.05, 955.59, 964.11, 945.17, 891.48, 807.11, 720.9,
                803.36, 834.46, 1073.9, 1107.1, 1123.6, 1296., 1393.7, 1303.1, 1284.3, 1161.8, 1078.8, 976.13, 868.72, 767.4, 674.72, 593.73,
01693
                523.12, 462.24, 409.75, 364.34, 325., 290.73, 260.76, 234.46,
01694
               211.28, 190.78, 172.61, 156.44, 142.01, 129.12, 117.57, 107.2, 97.877, 89.47, 81.882, 75.021, 68.807, 63.171, 58.052, 53.396, 49.155, 45.288, 41.759, 38.531, 35.576, 32.868, 30.384, 28.102, 26.003, 24.071, 22.293, 20.655, 19.147, 17.756, 16.476, 15.292,
01695
01696
01697
01698
               14.198, 13.183, 12.241, 11.367, 10.554, 9.7989, 9.0978, 8.4475, 7.845, 7.2868, 6.7704, 6.2927, 5.8508, 5.4421, 5.064, 4.714, 4.3902, 4.0902, 3.8121, 3.5543, 3.315, 3.093, 2.8869, 2.6953, 2.5172, 2.3517, 2.1977, 2.0544, 1.9211, 1.7969, 1.6812, 1.5735,
01699
01700
01701
01702
01703
                1.4731, 1.3794, 1.2921, 1.2107, 1.1346, 1.0637, .99744, .93554,
               87771, .82368, .77313, .72587, .6816, .64014, .60134, .565, .53086, .49883, .46881, .44074, .4144, .38979, .36679, .34513,
01704
01705
                .32474, .30552, .28751, .27045, .25458, .23976, .22584, .21278, .20051, .18899, .17815, .16801, .15846, .14954, .14117, .13328,
01706
01707
01708
                .12584
01709
01710
01711
            double xw, dw, ew, cw296, cw260, cw230, dt230, dt260, dt296, ctw, ctmpth;
01712
01713
            int iw:
01714
01715
             /* Get CO2 continuum absorption... */
01716
            xw = nu / 2 + 1;
01717
            if (xw >= 1 && xw < 2001) {
01718
              iw = (int) xw;
               dw = xw - iw;

ew = 1 - dw;
01719
01720
               cw296 = ew * co2296[iw - 1] + dw * co2296[iw];

cw260 = ew * co2260[iw - 1] + dw * co2260[iw];

cw230 = ew * co2230[iw - 1] + dw * co2230[iw];
01721
01722
01723
               dt230 = t - 230;

dt260 = t - 260;
01724
01725
               dt296 = t - 296;
01726
               ctw = dt260 * 5.050505e-4 * dt296 * cw230 - dt230 * 9.259259e-4
01727
                  * dt296 * cw260 + dt230 * 4.208754e-4 * dt260 * cw296;
01729
               ctmpth = u / NA / 1000 * p / P0 * ctw;
01730
01731
              ctmpth = 0;
01732
            return ctmpth;
01733 }
```

5.3.2.7 double ctmh2o (double nu, double p, double t, double q, double u)

Compute water vapor continuum (optical depth).

Definition at line 1737 of file jurassic.c.

```
01742
01743
           static double h2o296[2001] = { .17, .1695, .172, .168, .1687, .1624, .1606, .1508, .1447, .1344, .1214, .1133, .1009, .09217, .08297, .06989,
01744
             .06513, .05469, .05056, .04417, .03779, .03484, .02994, .0272, .02325, .02063, .01818, .01592, .01405, .01251, .0108, .009647 .008424, .007519, .006555, .00588, .005136, .004511, .003989,
01746
01747
01748
              .003509, .003114, .00274, .002446, .002144, .001895, .001676, .001486, .001312, .001164, .001031, 9.129e-4, 8.106e-4, 7.213e-4, 6.4e-4, 5.687e-4, 5.063e-4, 4.511e-4, 4.029e-4, 3.596e-4,
01749
01750
01751
              3.22e-4, 2.889e-4, 2.597e-4, 2.337e-4, 2.108e-4, 1.907e-4,
01753
              1.728e-4, 1.57e-4, 1.43e-4, 1.305e-4, 1.195e-4, 1.097e-4,
01754
              1.009e-4, 9.307e-5, 8.604e-5, 7.971e-5, 7.407e-5, 6.896e-5,
01755
              6.433e-5, 6.013e-5, 5.631e-5, 5.283e-5, 4.963e-5, 4.669e-5,
              4.398e-5, 4.148e-5, 3.917e-5, 3.702e-5, 3.502e-5, 3.316e-5,
01756
01757
              3.142e-5, 2.978e-5, 2.825e-5, 2.681e-5, 2.546e-5, 2.419e-5,
              2.299e-5, 2.186e-5, 2.079e-5, 1.979e-5, 1.884e-5, 1.795e-5,
              1.711e-5, 1.633e-5, 1.559e-5, 1.49e-5, 1.426e-5, 1.367e-5,
01759
01760
              1.312e-5, 1.263e-5, 1.218e-5, 1.178e-5, 1.143e-5, 1.112e-5,
             1.088e-5, 1.07e-5, 1.057e-5, 1.05e-5, 1.051e-5, 1.059e-5, 1.076e-5, 1.1e-5, 1.133e-5, 1.18e-5, 1.237e-5, 1.308e-5, 1.393e-5, 1.483e-5, 1.614e-5, 1.758e-5, 1.93e-5, 2.123e-5, 2.346e-5, 2.647e-5, 2.93e-5, 3.279e-5, 3.745e-5, 4.152e-5,
01761
01762
01763
01765
              4.813e-5, 5.477e-5, 6.203e-5, 7.331e-5, 8.056e-5, 9.882e-5
01766
              1.05e-4, 1.21e-4, 1.341e-4, 1.572e-4, 1.698e-4, 1.968e-4,
01767
              2.175e-4, 2.431e-4, 2.735e-4, 2.867e-4, 3.19e-4, 3.371e-4,
01768
              3.554e-4, 3.726e-4, 3.837e-4, 3.878e-4, 3.864e-4, 3.858e-4, 3.841e-4, 3.852e-4, 3.815e-4, 3.762e-4, 3.618e-4, 3.579e-4,
01769
01770
              3.45e-4, 3.202e-4, 3.018e-4, 2.785e-4, 2.602e-4, 2.416e-4,
01771
              2.097e-4, 1.939e-4, 1.689e-4, 1.498e-4, 1.308e-4, 1.17e-4,
```

```
1.011e-4, 9.237e-5, 7.909e-5, 7.006e-5, 6.112e-5, 5.401e-5,
             4.914e-5, 4.266e-5, 3.963e-5, 3.316e-5, 3.037e-5, 2.598e-5, 2.294e-5, 2.066e-5, 1.813e-5, 1.583e-5, 1.423e-5, 1.247e-5,
01773
01774
01775
             1.116e-5, 9.76e-6, 8.596e-6, 7.72e-6, 6.825e-6, 6.108e-6,
             5.366e-6, 4.733e-6, 4.229e-6, 3.731e-6, 3.346e-6, 2.972e-6, 2.628e-6, 2.356e-6, 2.102e-6, 1.878e-6, 1.678e-6, 1.507e-6,
01776
01777
             1.348e-6, 1.21e-6, 1.089e-6, 9.806e-7, 8.857e-7, 8.004e-7, 7.261e-7, 6.599e-7, 6.005e-7, 5.479e-7, 5.011e-7, 4.595e-7
01778
01779
01780
             4.219e-7, 3.885e-7, 3.583e-7, 3.314e-7, 3.071e-7, 2.852e-7,
01781
             2.654e-7, 2.474e-7, 2.311e-7, 2.162e-7, 2.026e-7, 1.902e-7
             1.788e-7, 1.683e-7, 1.587e-7, 1.497e-7, 1.415e-7, 1.338e-7,
01782
             1.266e-7, 1.2e-7, 1.138e-7, 1.08e-7, 1.027e-7, 9.764e-8,
01783
01784
             9.296e-8, 8.862e-8, 8.458e-8, 8.087e-8, 7.744e-8, 7.429e-8,
             7.145e-8, 6.893e-8, 6.664e-8, 6.468e-8, 6.322e-8, 6.162e-8,
01785
01786
             6.07e-8, 5.992e-8, 5.913e-8, 5.841e-8, 5.796e-8, 5.757e-8,
             5.746e-8, 5.731e-8, 5.679e-8, 5.577e-8, 5.671e-8, 5.656e-8, 5.594e-8, 5.593e-8, 5.602e-8, 5.62e-8, 5.693e-8, 5.725e-8,
01787
01788
             5.858e-8, 6.037e-8, 6.249e-8, 6.535e-8, 6.899e-8, 7.356e-8,
01789
             7.918e-8, 8.618e-8, 9.385e-8, 1.039e-7, 1.158e-7, 1.29e-7,
             1.437e-7, 1.65e-7, 1.871e-7, 2.121e-7, 2.427e-7, 2.773e-7, 3.247e-7, 3.677e-7, 4.037e-7, 4.776e-7, 5.101e-7, 6.214e-7, 6.936e-7, 7.581e-7, 8.486e-7, 9.355e-7, 9.942e-7, 1.063e-6,
01791
01792
01793
01794
             1.123e-6, 1.191e-6, 1.215e-6, 1.247e-6, 1.26e-6, 1.271e-6,
             1.284e-6, 1.317e-6, 1.323e-6, 1.349e-6, 1.353e-6, 1.362e-6, 1.344e-6, 1.329e-6, 1.336e-6, 1.327e-6, 1.325e-6, 1.359e-6,
01795
01796
01797
             1.374e-6, 1.415e-6, 1.462e-6, 1.526e-6, 1.619e-6, 1.735e-6,
01798
             1.863e-6, 2.034e-6, 2.265e-6, 2.482e-6, 2.756e-6, 3.103e-6,
01799
             3.466e-6, 3.832e-6, 4.378e-6, 4.913e-6, 5.651e-6, 6.311e-6,
             7.169e-6, 8.057e-6, 9.253e-6, 1.047e-5, 1.212e-5, 1.36e-5, 1.569e-5, 1.776e-5, 2.02e-5, 2.281e-5, 2.683e-5, 2.994e-5, 3.488e-5, 3.896e-5, 4.499e-5, 5.175e-5, 6.035e-5, 6.34e-5,
01800
01801
01802
01803
              7.281e-5, 7.923e-5, 8.348e-5, 9.631e-5, 1.044e-4, 1.102e-4,
             1.176e-4, 1.244e-4, 1.283e-4, 1.326e-4, 1.4e-4, 1.395e-4,
01804
01805
             1.387e-4, 1.363e-4, 1.314e-4, 1.241e-4, 1.228e-4, 1.148e-4,
             1.086e-4, 1.018e-4, 8.89e-5, 8.316e-5, 7.292e-5, 6.452e-5, 5.625e-5, 5.045e-5, 4.38e-5, 3.762e-5, 3.29e-5, 2.836e-5,
01806
01807
             2.485e-5, 2.168e-5, 1.895e-5, 1.659e-5, 1.453e-5, 1.282e-5,
01808
             1.132e-5, 1.001e-5, 8.836e-6, 7.804e-6, 6.922e-6, 6.116e-6,
01810
             5.429e-6, 4.824e-6, 4.278e-6, 3.788e-6, 3.371e-6, 2.985e-6,
             2.649e-6, 2.357e-6, 2.09e-6, 1.858e-6, 1.647e-6, 1.462e-6, 1.299e-6, 1.155e-6, 1.028e-6, 9.142e-7, 8.132e-7, 7.246e-7,
01811
01812
             6.451e-7, 5.764e-7, 5.151e-7, 4.603e-7, 4.121e-7, 3.694e-7, 3.318e-7, 2.985e-7, 2.69e-7, 2.428e-7, 2.197e-7, 1.992e-7, 1.81e-7, 1.649e-7, 1.506e-7, 1.378e-7, 1.265e-7, 1.163e-7,
01813
01814
01815
             1.073e-7, 9.918e-8, 9.191e-8, 8.538e-8, 7.949e-8, 7.419e-8,
01817
             6.94e-8, 6.508e-8, 6.114e-8, 5.761e-8, 5.437e-8, 5.146e-8,
01818
             4.89e-8, 4.636e-8, 4.406e-8, 4.201e-8, 4.015e-8, 3.84e-8,
01819
             3.661e-8, 3.51e-8, 3.377e-8, 3.242e-8, 3.13e-8, 3.015e-8,
             2.918e-8, 2.83e-8, 2.758e-8, 2.707e-8, 2.656e-8, 2.619e-8, 2.609e-8, 2.615e-8, 2.63e-8, 2.675e-8, 2.745e-8, 2.842e-8,
01820
01821
             2.966e-8, 3.125e-8, 3.318e-8, 3.565e-8, 3.85e-8, 4.191e-8,
             4.59e-8, 5.059e-8, 5.607e-8, 6.239e-8, 6.958e-8, 7.796e-8, 8.773e-8, 9.88e-8, 1.114e-7, 1.258e-7, 1.422e-7, 1.61e-7,
01823
01824
             1.822e-7, 2.06e-7, 2.337e-7, 2.645e-7, 2.996e-7, 3.393e-7, 3.843e-7, 4.363e-7, 4.935e-7, 5.607e-7, 6.363e-7, 7.242e-7, 8.23e-7, 9.411e-7, 1.071e-6, 1.232e-6, 1.402e-6, 1.6e-6, 1.82e-6,
01825
01826
01827
             2.128e-6, 2.386e-6, 2.781e-6, 3.242e-6, 3.653e-6, 4.323e-6,
             4.747e-6, 5.321e-6, 5.919e-6, 6.681e-6, 7.101e-6, 7.983e-6,
01829
             8.342e-6, 8.741e-6, 9.431e-6, 9.952e-6, 1.026e-5, 1.055e-5,
01830
01831
             1.095e-5, 1.095e-5, 1.087e-5, 1.056e-5, 1.026e-5, 9.715e-6,
01832
             9.252e-6, 8.452e-6, 7.958e-6, 7.268e-6, 6.295e-6, 6.003e-6, 5e-6,
01833
             4.591e-6, 3.983e-6, 3.479e-6, 3.058e-6, 2.667e-6, 2.293e-6,
01834
             1.995e-6, 1.747e-6, 1.517e-6, 1.335e-6, 1.165e-6, 1.028e-6,
             9.007e-7, 7.956e-7, 7.015e-7, 6.192e-7, 5.491e-7, 4.859e-7, 4.297e-7, 3.799e-7, 3.38e-7, 3.002e-7, 2.659e-7, 2.366e-7,
01835
01836
             2.103e-7, 1.861e-7, 1.655e-7, 1.469e-7, 1.309e-7, 1.162e-7,
01837
             1.032e-7, 9.198e-8, 8.181e-8, 7.294e-8, 6.516e-8, 5.787e-8, 5.163e-8, 4.612e-8, 4.119e-8, 3.695e-8, 3.308e-8, 2.976e-8,
01838
01839
             2.67e-8, 2.407e-8, 2.171e-8, 1.965e-8, 1.78e-8, 1.617e-8, 1.47e-8, 1.341e-8, 1.227e-8, 1.125e-8, 1.033e-8, 9.524e-9,
01840
             8.797e-9, 8.162e-9, 7.565e-9, 7.04e-9, 6.56e-9, 6.129e-9,
01842
01843
             5.733e-9, 5.376e-9, 5.043e-9, 4.75e-9, 4.466e-9, 4.211e-9,
             3.977e-9, 3.759e-9, 3.558e-9, 3.373e-9, 3.201e-9, 3.043e-9, 2.895e-9, 2.76e-9, 2.635e-9, 2.518e-9, 2.411e-9, 2.314e-9, 2.23e-9, 2.151e-9, 2.087e-9, 2.035e-9, 1.988e-9, 1.946e-9,
01844
01845
01846
             1.927e-9, 1.916e-9, 1.916e-9, 1.933e-9, 1.966e-9, 2.018e-9,
             2.09e-9, 2.182e-9, 2.299e-9, 2.442e-9, 2.623e-9, 2.832e-9,
01848
01849
             3.079e-9, 3.368e-9, 3.714e-9, 4.104e-9, 4.567e-9, 5.091e-9,
             5.701e-9, 6.398e-9, 7.194e-9, 8.127e-9, 9.141e-9, 1.035e-8,
01850
01851
             1.177e-8, 1.338e-8, 1.508e-8, 1.711e-8, 1.955e-8, 2.216e-8,
             2.534e-8, 2.871e-8, 3.291e-8, 3.711e-8, 4.285e-8, 4.868e-8,
01852
             5.509e-8, 6.276e-8, 7.262e-8, 8.252e-8, 9.4e-8, 1.064e-7,
             1.247e-7, 1.411e-7, 1.626e-7, 1.827e-7, 2.044e-7, 2.284e-7, 2.452e-7, 2.854e-7, 3.026e-7, 3.278e-7, 3.474e-7, 3.693e-7,
01854
01855
             3.93e-7, 4.104e-7, 4.22e-7, 4.439e-7, 4.545e-7, 4.778e-7, 4.812e-7, 5.018e-7, 4.899e-7, 5.075e-7, 5.073e-7, 5.171e-7, 5.131e-7, 5.25e-7, 5.617e-7, 5.846e-7, 6.239e-7, 6.696e-7,
01856
01857
01858
```

```
7.398e-7, 8.073e-7, 9.15e-7, 1.009e-6, 1.116e-6, 1.264e-6,
             1.439e-6, 1.644e-6, 1.856e-6, 2.147e-6, 2.317e-6, 2.713e-6, 2.882e-6, 2.99e-6, 3.489e-6, 3.581e-6, 4.033e-6, 4.26e-6,
01860
01861
             4.543e-6, 4.84e-6, 4.826e-6, 5.013e-6, 5.252e-6, 5.277e-6,
01862
             5.306e-6, 5.236e-6, 5.123e-6, 5.171e-6, 4.843e-6, 4.615e-6, 4.385e-6, 3.97e-6, 3.693e-6, 3.231e-6, 2.915e-6, 2.495e-6,
01863
01864
             2.144e-6, 1.91e-6, 1.639e-6, 1.417e-6, 1.226e-6, 1.065e-6,
             9.29e-7, 8.142e-7, 7.161e-7, 6.318e-7, 5.581e-7, 4.943e-7,
01866
             4.376e-7, 3.884e-7, 3.449e-7, 3.06e-7, 2.712e-7, 2.139e-7, 1.903e-7, 1.689e-7, 1.499e-7, 1.331e-7, 1.183e-7, 1.05e-7, 9.362e-8, 8.306e-8, 7.403e-8, 6.578e-8, 5.853e-8,
01867
01868
01869
             5.216e-8, 4.632e-8, 4.127e-8, 3.678e-8, 3.279e-8, 2.923e-8, 2.612e-8, 2.339e-8, 2.094e-8, 1.877e-8, 1.686e-8, 1.516e-8,
01870
01871
             1.366e-8, 1.234e-8, 1.114e-8, 1.012e-8, 9.182e-9, 8.362e-9,
01872
01873
             7.634e-9, 6.981e-9, 6.406e-9, 5.888e-9, 5.428e-9, 5.021e-9,
             4.65e-9, 4.326e-9, 4.033e-9, 3.77e-9, 3.536e-9, 3.327e-9, 3.141e-9, 2.974e-9, 2.825e-9, 2.697e-9, 2.584e-9, 2.488e-9, 2.406e-9, 2.34e-9, 2.292e-9, 2.259e-9, 2.244e-9, 2.243e-9, 2.272e-9, 2.31e-9, 2.378e-9, 2.454e-9, 2.618e-9, 2.672e-9,
01874
01875
01876
01878
             2.831e-9, 3.05e-9, 3.225e-9, 3.425e-9, 3.677e-9, 3.968e-9,
             4.221e-9, 4.639e-9, 4.96e-9, 5.359e-9, 5.649e-9, 6.23e-9, 6.716e-9, 7.218e-9, 7.746e-9, 7.988e-9, 8.627e-9, 8.999e-9,
01879
01880
             9.442e-9, 9.82e-9, 1.015e-8, 1.06e-8, 1.079e-8, 1.109e-8, 1.137e-8, 1.186e-8, 1.18e-8, 1.187e-8, 1.194e-8, 1.192e-8, 1.224e-8, 1.245e-8, 1.246e-8, 1.318e-8, 1.377e-8, 1.471e-8,
01881
01882
01883
             1.582e-8, 1.713e-8, 1.853e-8, 2.063e-8, 2.27e-8, 2.567e-8,
01885
             2.891e-8, 3.264e-8, 3.744e-8, 4.286e-8, 4.915e-8, 5.623e-8
01886
             6.336e-8, 7.293e-8, 8.309e-8, 9.319e-8, 1.091e-7, 1.243e-7,
             1.348e-7, 1.449e-7, 1.62e-7, 1.846e-7, 1.937e-7, 2.04e-7, 2.179e-7, 2.298e-7, 2.433e-7, 2.439e-7, 2.464e-7, 2.611e-7,
01887
01888
             2.617e-7, 2.582e-7, 2.453e-7, 2.401e-7, 2.349e-7, 2.203e-7, 2.066e-7, 1.939e-7, 1.78e-7, 1.558e-7, 1.391e-7, 1.203e-7,
01889
01890
             1.048e-7, 9.464e-8, 8.306e-8, 7.239e-8, 6.317e-8, 5.52e-8
01891
01892
             4.847e-8, 4.282e-8, 3.796e-8, 3.377e-8, 2.996e-8, 2.678e-8,
01893
             2.4e-8, 2.134e-8, 1.904e-8, 1.705e-8, 1.523e-8, 1.35e-8,
             1.204e-8, 1.07e-8, 9.408e-9, 8.476e-9, 7.47e-9, 6.679e-9,
01894
             5.929e-9, 5.267e-9, 4.711e-9, 4.172e-9, 3.761e-9, 3.288e-9,
01895
             2.929e-9, 2.609e-9, 2.315e-9, 2.042e-9, 1.844e-9, 1.64e-9,
01897
             1.47e-9, 1.31e-9, 1.176e-9, 1.049e-9, 9.377e-10, 8.462e-10,
01898
             7.616e-10, 6.854e-10, 6.191e-10, 5.596e-10, 5.078e-10, 4.611e-10,
01899
             4.197e-10, 3.83e-10, 3.505e-10, 3.215e-10, 2.956e-10, 2.726e-10,
01900
             2.521e-10, 2.338e-10, 2.173e-10, 2.026e-10, 1.895e-10, 1.777e-10,
             1.672e-10, 1.579e-10, 1.496e-10, 1.423e-10, 1.358e-10, 1.302e-10, 1.254e-10, 1.216e-10, 1.187e-10, 1.163e-10, 1.147e-10, 1.145e-10,
01901
             1.15e-10, 1.17e-10, 1.192e-10, 1.25e-10, 1.298e-10, 1.345e-10,
01903
01904
             1.405e-10, 1.538e-10, 1.648e-10, 1.721e-10, 1.872e-10, 1.968e-10,
01905
             2.089e-10, 2.172e-10, 2.317e-10, 2.389e-10, 2.503e-10, 2.585e-10,
             2.686e-10, 2.8e-10, 2.895e-10, 3.019e-10, 3.037e-10, 3.076e-10, 3.146e-10, 3.198e-10, 3.332e-10, 3.397e-10, 3.54e-10, 3.667e-10, 3.895e-10, 4.071e-10, 4.565e-10, 4.983e-10, 5.439e-10, 5.968e-10,
01906
01907
01908
             6.676e-10, 7.456e-10, 8.405e-10, 9.478e-10, 1.064e-9,
             1.386e-9, 1.581e-9, 1.787e-9, 2.032e-9, 2.347e-9, 2.677e-9,
01910
01911
             3.008e-9, 3.544e-9, 4.056e-9, 4.687e-9, 5.331e-9, 6.227e-9,
             6.854e-9, 8.139e-9, 8.945e-9, 9.865e-9, 1.125e-8, 1.178e-8, 1.364e-8, 1.436e-8, 1.54e-8, 1.672e-8, 1.793e-8, 1.906e-8,
01912
01913
             2.036e-8, 2.144e-8, 2.292e-8, 2.371e-8, 2.493e-8, 2.606e-8,
01914
             2.706e-8, 2.866e-8, 3.036e-8, 3.136e-8, 3.405e-8, 3.665e-8,
01916
             3.837e-8, 4.229e-8, 4.748e-8, 5.32e-8, 5.763e-8, 6.677e-8,
01917
             7.216e-8, 7.716e-8, 8.958e-8, 9.419e-8, 1.036e-7, 1.108e-7,
             1.189e-7, 1.246e-7, 1.348e-7, 1.31e-7, 1.361e-7, 1.364e-7, 1.363e-7, 1.343e-7, 1.293e-7, 1.254e-7, 1.235e-7, 1.158e-7,
01918
01919
             1.107e-7, 9.961e-8, 9.011e-8, 7.91e-8, 6.916e-8, 6.338e-8, 5.564e-8, 4.827e-8, 4.198e-8, 3.695e-8, 3.276e-8, 2.929e-8,
01920
             2.633e-8, 2.391e-8, 2.192e-8, 2.021e-8, 1.89e-8, 1.772e-8,
01922
01923
             1.667e-8, 1.603e-8, 1.547e-8, 1.537e-8, 1.492e-8, 1.515e-8,
01924
             1.479e-8, 1.45e-8, 1.513e-8, 1.495e-8, 1.529e-8, 1.565e-8,
01925
             1.564e-8, 1.553e-8, 1.569e-8, 1.584e-8, 1.57e-8, 1.538e-8, 1.513e-8, 1.472e-8, 1.425e-8, 1.349e-8, 1.328e-8, 1.249e-8,
01926
             1.17e-8, 1.077e-8, 9.514e-9, 8.614e-9, 7.46e-9, 6.621e-9,
01927
             5.775e-9, 5.006e-9, 4.308e-9, 3.747e-9, 3.24e-9, 2.84e-9,
01928
             2.481e-9, 2.184e-9, 1.923e-9, 1.71e-9, 1.504e-9, 1.334e-9
01929
01930
             1.187e-9, 1.053e-9, 9.367e-10, 8.306e-10, 7.419e-10, 6.63e-10,
             5.918e-10, 5.277e-10, 4.717e-10, 4.222e-10, 3.783e-10, 3.39e-10,
01931
             3.036e-10, 2.729e-10, 2.455e-10, 2.211e-10, 1.995e-10, 1.804e-10,
01932
             1.635e-10, 1.485e-10, 1.355e-10, 1.24e-10, 1.139e-10, 1.051e-10,
01933
             9.757e-11, 9.114e-11, 8.577e-11, 8.139e-11, 7.792e-11, 7.52e-11, 7.31e-11, 7.277e-11, 7.482e-11, 7.698e-11, 8.162e-11,
01934
01935
01936
             8.517e-11, 8.968e-11, 9.905e-11, 1.075e-10, 1.187e-10, 1.291e-10,
             1.426 e^{-10},\ 1.573 e^{-10},\ 1.734 e^{-10},\ 1.905 e^{-10},\ 2.097 e^{-10},\ 2.28 e^{-10},
01937
             2.473e-10. 2.718e-10. 2.922e-10. 3.128e-10. 3.361e-10. 3.641e-10.
01938
             3.91e-10, 4.196e-10, 4.501e-10, 4.322e-10, 5.258e-10, 5.755e-10, 6.253e-10, 6.664e-10, 7.344e-10, 7.985e-10, 8.877e-10, 1.005e-9,
01939
01941
             1.118e-9, 1.251e-9, 1.428e-9, 1.61e-9, 1.888e-9, 2.077e-9,
01942
             2.331e-9, 2.751e-9, 3.061e-9, 3.522e-9, 3.805e-9, 4.181e-9,
01943
             4.575e-9, 5.167e-9, 5.634e-9, 6.007e-9, 6.501e-9, 6.829e-9,
             7.211e-9, 7.262e-9, 7.696e-9, 7.832e-9, 7.799e-9, 7.651e-9, 7.304e-9, 7.15e-9, 6.677e-9, 6.603e-9, 6.209e-9, 5.69e-9,
01944
01945
```

```
5.432e-9, 4.764e-9, 4.189e-9, 3.64e-9, 3.203e-9, 2.848e-9,
            2.51e-9, 2.194e-9, 1.946e-9, 1.75e-9, 1.567e-9, 1.426e-9,
01947
            1.302e-9, 1.197e-9, 1.109e-9, 1.035e-9, 9.719e-10, 9.207e-10,
01948
01949
            8.957e-10, 8.578e-10, 8.262e-10, 8.117e-10, 7.987e-10, 7.875e-10,
            7.741e-10, 7.762e-10, 7.537e-10, 7.424e-10, 7.474e-10, 7.294e-10, 7.216e-10, 7.233e-10, 7.075e-10, 6.892e-10, 6.618e-10, 6.314e-10,
01950
01951
            6.208e-10, 5.689e-10, 5.55e-10, 4.984e-10, 4.6e-10, 4.078e-10,
            3.879e-10, 3.459e-10, 2.982e-10, 2.626e-10, 2.329e-10, 1.988e-10,
01953
01954
            1.735e-10, 1.487e-10, 1.297e-10, 1.133e-10, 9.943e-11, 8.736e-11,
           7.726e-11, 6.836e-11, 6.053e-11, 5.384e-11, 4.789e-11, 4.267e-11, 3.804e-11, 3.398e-11, 3.034e-11, 2.71e-11, 2.425e-11, 2.173e-11, 1.95e-11, 1.752e-11, 1.574e-11, 1.418e-11, 1.278e-11, 1.154e-11, 1.044e-11, 9.463e-12, 8.602e-12, 7.841e-12, 7.171e-12, 6.584e-12,
01955
01956
01957
01958
            6.073e-12, 5.631e-12, 5.254e-12, 4.937e-12, 4.679e-12, 4.476e-12,
01959
01960
            4.328e-12, 4.233e-12, 4.194e-12, 4.211e-12, 4.286e-12, 4.424e-12,
01961
            4.628e-12, 4.906e-12, 5.262e-12, 5.708e-12, 6.254e-12, 6.914e-12,
            7.714e-12, 8.677e-12, 9.747e-12, 1.101e-11, 1.256e-11, 1.409e-11,
01962
            1.597e-11, 1.807e-11, 2.034e-11, 2.316e-11, 2.622e-11, 2.962e-11,
01963
01964
            3.369e-11, 3.819e-11, 4.329e-11, 4.932e-11, 5.589e-11, 6.364e-11,
            7.284e-11, 8.236e-11, 9.447e-11, 1.078e-10, 1.229e-10, 1.417e-10,
            1.614e-10, 1.843e-10, 2.107e-10, 2.406e-10, 2.728e-10, 3.195e-10,
01966
01967
            3.595e-10, 4.153e-10, 4.736e-10, 5.41e-10, 6.088e-10, 6.769e-10,
01968
            7.691e-10, 8.545e-10, 9.621e-10, 1.047e-9, 1.161e-9, 1.296e-9,
            1.424e-9, 1.576e-9, 1.739e-9, 1.893e-9, 2.08e-9, 2.336e-9, 2.604e-9, 2.76e-9, 3.001e-9, 3.365e-9, 3.55e-9, 3.895e-9,
01969
            4.183e-9, 4.614e-9, 4.846e-9, 5.068e-9, 5.427e-9, 5.541e-9,
01971
            5.864e-9, 5.997e-9, 5.997e-9, 6.061e-9, 5.944e-9, 5.855e-9,
01972
01973
            5.661e-9, 5.523e-9, 5.374e-9, 4.94e-9, 4.688e-9, 4.17e-9,
           3.913e-9, 3.423e-9, 2.997e-9, 2.598e-9, 2.253e-9, 1.946e-9, 1.71e-9, 1.507e-9, 1.336e-9, 1.19e-9, 1.068e-9, 9.623e-10,
01974
01975
            8.772e-10, 8.007e-10, 7.42e-10, 6.884e-10, 6.483e-10, 6.162e-10,
01976
            5.922e-10, 5.688e-10, 5.654e-10, 5.637e-10, 5.701e-10, 5.781e-10,
            5.874e-10, 6.268e-10, 6.357e-10, 6.525e-10, 7.137e-10, 7.441e-10,
01978
01979
            8.024e-10, 8.485e-10, 9.143e-10, 9.536e-10, 9.717e-10,
            1.042e-9, 1.054e-9, 1.092e-9, 1.079e-9, 1.064e-9, 1.043e-9, 1.02e-9, 9.687e-10, 9.273e-10, 9.208e-10, 9.068e-10, 7.687e-10,
01980
01981
            7.385e-10, 6.595e-10, 5.87e-10, 5.144e-10, 4.417e-10, 3.804e-10,
01982
            3.301e-10, 2.866e-10, 2.509e-10, 2.202e-10, 1.947e-10, 1.719e-10,
01984
            1.525e-10, 1.361e-10, 1.21e-10, 1.084e-10, 9.8e-11, 8.801e-11,
            7.954e-11, 7.124e-11, 6.335e-11, 5.76e-11, 5.132e-11, 4.601e-11,
01985
01986
            4.096e-11, 3.657e-11, 3.25e-11, 2.909e-11, 2.587e-11, 2.297e-11,
           2.05e-11, 1.828e-11, 1.632e-11, 1.462e-11, 1.314e-11, 1.185e-11, 1.073e-11, 9.76e-12, 8.922e-12, 8.206e-12, 7.602e-12, 7.1e-12, 6.694e-12, 6.378e-12, 6.149e-12, 6.004e-12, 5.941e-12, 5.962e-12,
01987
01988
01990
            6.069e-12, 6.265e-12, 6.551e-12, 6.935e-12, 7.457e-12, 8.074e-12,
01991
            8.811e-12, 9.852e-12, 1.086e-11, 1.207e-11, 1.361e-11, 1.553e-11,
01992
            1.737e-11, 1.93e-11, 2.175e-11, 2.41e-11, 2.706e-11, 3.023e-11,
01993
            3.313 e-11, \ 3.657 e-11, \ 4.118 e-11, \ 4.569 e-11, \ 5.025 e-11, \ 5.66 e-11, \\
01994
            6.231e-11, 6.881e-11, 7.996e-11, 8.526e-11, 9.694e-11, 1.106e-10,
            1.222e-10, 1.355e-10, 1.525e-10, 1.775e-10, 1.924e-10, 2.181e-10,
01995
            2.379e-10, 2.662e-10, 2.907e-10, 3.154e-10, 3.366e-10, 3.579e-10,
            3.858e-10, 4.046e-10, 4.196e-10, 4.166e-10, 4.457e-10, 4.466e-10,
01997
01998
            4.404e-10, 4.337e-10, 4.15e-10, 4.083e-10, 3.91e-10, 3.723e-10,
01999
            3.514e-10, 3.303e-10, 2.847e-10, 2.546e-10, 2.23e-10, 1.994e-10,
            1.733e-10, 1.488e-10, 1.297e-10, 1.144e-10, 1.004e-10, 8.741e-11,
02000
            7.928e-11, 7.034e-11, 6.323e-11, 5.754e-11, 5.25e-11, 4.85e-11,
02001
            4.502e-11, 4.286e-11, 4.028e-11, 3.899e-11, 3.824e-11, 3.761e-11,
            3.804e-11, 3.839e-11, 3.845e-11, 4.244e-11, 4.382e-11, 4.582e-11,
02003
            4.847e-11, 5.209e-11, 5.384e-11, 5.887e-11, 6.371e-11, 6.737e-11,
02004
02005
            7.168e-11, 7.415e-11, 7.827e-11, 8.037e-11, 8.12e-11, 8.071e-11,
02006
            8.008e-11,\ 7.851e-11,\ 7.544e-11,\ 7.377e-11,\ 7.173e-11,\ 6.801e-11,
           6.267e-11, 5.727e-11, 5.288e-11, 4.853e-11, 4.082e-11, 3.645e-11, 3.136e-11, 2.672e-11, 2.304e-11, 1.986e-11, 1.725e-11, 1.503e-11,
02007
            1.315e-11, 1.153e-11, 1.014e-11, 8.942e-12, 7.901e-12, 6.993e-12, 6.199e-12, 5.502e-12, 4.89e-12, 4.351e-12, 3.878e-12, 3.461e-12,
02009
02010
02011
           3.094e-12, 2.771e-12, 2.488e-12, 2.241e-12, 2.025e-12, 1.838e-12,
02012
            1.677e-12, 1.541e-12, 1.427e-12, 1.335e-12, 1.262e-12, 1.209e-12,
            1.176e-12, 1.161e-12, 1.165e-12, 1.189e-12, 1.234e-12, 1.3e-12,
02013
            1.389e-12, 1.503e-12, 1.644e-12, 1.814e-12, 2.017e-12, 2.255e-12,
02014
            2.534e-12, 2.858e-12, 3.231e-12, 3.661e-12, 4.153e-12, 4.717e-12,
            5.36e-12, 6.094e-12, 6.93e-12, 7.882e-12, 8.966e-12, 1.02e-11
02016
02017
            1.162e-11, 1.324e-11, 1.51e-11, 1.72e-11, 1.965e-11, 2.237e-11,
02018
            2.56e-11, 2.927e-11, 3.371e-11, 3.842e-11, 4.429e-11, 5.139e-11,
            5.798e-11, 6.697e-11, 7.626e-11, 8.647e-11, 1.022e-10, 1.136e-10,
02019
            1.3e-10, 1.481e-10, 1.672e-10, 1.871e-10, 2.126e-10, 2.357e-10, 2.583e-10, 2.997e-10, 3.289e-10, 3.702e-10, 4.012e-10, 4.319e-10,
02020
02021
            4.527e-10, 5.001e-10, 5.448e-10, 5.611e-10, 5.76e-10, 5.965e-10,
02022
02023
            6.079e-10, 6.207e-10, 6.276e-10, 6.222e-10, 6.137e-10, 6e-10,
            5.814e-10, 5.393e-10, 5.35e-10, 4.947e-10, 4.629e-10, 4.117e-10, 3.712e-10, 3.372e-10, 2.923e-10, 2.55e-10, 2.232e-10, 1.929e-10,
02024
02025
            1.679e-10, 1.46e-10, 1.289e-10, 1.13e-10, 9.953e-11, 8.763e-11,
02026
            7.76e-11, 6.9e-11, 6.16e-11, 5.525e-11, 4.958e-11, 4.489e-11,
            4.072e-11, 3.728e-11, 3.438e-11, 3.205e-11, 3.006e-11, 2.848e-11, 2.766e-11, 2.688e-11, 2.664e-11, 2.67e-11, 2.696e-11, 2.786e-11,
02028
02029
02030
            2.861e-11, 3.009e-11, 3.178e-11, 3.389e-11, 3.587e-11, 3.819e-11,
            4.054e-11, 4.417e-11, 4.703e-11, 5.137e-11, 5.46e-11, 6.055e-11, 6.333e-11, 6.773e-11, 7.219e-11, 7.717e-11, 8.131e-11, 8.491e-11,
02031
02032
```

```
8.574e-11, 9.01e-11, 9.017e-11, 8.999e-11, 8.959e-11, 8.838e-11,
            8.579e-11, 8.162e-11, 8.098e-11, 7.472e-11, 7.108e-11, 6.559e-11, 5.994e-11, 5.172e-11, 4.424e-11, 3.951e-11, 3.34e-11, 2.902e-11,
02034
02035
            2.54le-11, 2.215e-11, 1.945e-11, 1.716e-11, 1.503e-11, 1.339e-11, 1.185e-11, 1.05e-11, 9.336e-12, 8.307e-12, 7.312e-12, 6.55e-12, 5.836e-12, 5.178e-12, 4.6e-12, 4.086e-12, 3.639e-12, 3.247e-12,
02036
02037
02038
            2.904e-12, 2.604e-12, 2.341e-12, 2.112e-12, 1.914e-12, 1.744e-12,
            1.598e-12, 1.476e-12, 1.374e-12, 1.293e-12, 1.23e-12, 1.185e-12,
02040
02041
            1.158e-12, 1.147e-12, 1.154e-12, 1.177e-12, 1.219e-12, 1.28e-12,
02042
            1.36 e^{-12}, \ 1.463 e^{-12}, \ 1.591 e^{-12}, \ 1.75 e^{-12}, \ 1.94 e^{-12}, \ 2.156 e^{-12},
            2.43e-12, 2.748e-12, 3.052e-12, 3.533e-12, 3.967e-12, 4.471e-12, 5.041e-12, 5.86e-12, 6.664e-12, 7.522e-12, 8.342e-12, 9.412e-12,
02043
02044
02045
            1.072e-11, 1.213e-11, 1.343e-11, 1.496e-11, 1.664e-11, 1.822e-11,
            2.029e-11, 2.233e-11, 2.457e-11, 2.709e-11, 2.928e-11, 3.115e-11,
02046
02047
            3.356e-11, 3.592e-11, 3.818e-11, 3.936e-11, 4.061e-11, 4.149e-11,
02048
            4.299e-11, 4.223e-11, 4.251e-11, 4.287e-11, 4.177e-11, 4.094e-11,
            3.942e-11, 3.772e-11, 3.614e-11, 3.394e-11, 3.222e-11, 2.791e-11,
02049
            2.665e-11, 2.309e-11, 2.032e-11, 1.74e-11, 1.535e-11, 1.323e-11, 1.151e-11, 9.803e-12, 8.65e-12, 7.54e-12, 6.619e-12, 5.832e-12,
02050
            5.113e-12, 4.503e-12, 3.975e-12, 3.52e-12, 3.112e-12, 2.797e-12,
02052
            2.5e-12, 2.24e-12, 2.013e-12, 1.819e-12, 1.653e-12, 1.513e-12,
02053
02054
            1.395e-12, 1.299e-12, 1.225e-12, 1.168e-12, 1.124e-12, 1.148e-12,
            1.107e-12, 1.128e-12, 1.169e-12, 1.233e-12, 1.307e-12, 1.359e-12,
02055
            1.543e-12, 1.686e-12, 1.794e-12, 2.028e-12, 2.21e-12, 2.441e-12, 2.653e-12, 2.828e-12, 3.093e-12, 3.28e-12, 3.551e-12, 3.677e-12,
02056
02057
            3.803e-12, 3.844e-12, 4.068e-12, 4.093e-12, 4.002e-12, 3.904e-12,
02058
02059
            3.624e-12, 3.633e-12, 3.622e-12, 3.443e-12, 3.184e-12, 2.934e-12,
            2.476e-12, 2.212e-12, 1.867e-12, 1.594e-12, 1.37e-12, 1.192e-12, 1.045e-12, 9.211e-13, 8.17e-13, 7.29e-13, 6.55e-13, 5.929e-13, 5.415e-13, 4.995e-13, 4.661e-13, 4.406e-13, 4.225e-13, 4.116e-13,
02060
02061
02062
02063
            4.075e-13, 4.102e-13, 4.198e-13, 4.365e-13, 4.606e-13, 4.925e-13,
02064
            5.326e-13, 5.818e-13, 6.407e-13, 7.104e-13, 7.92e-13, 8.868e-13,
02065
            9.964e-13, 1.123e-12, 1.268e-12, 1.434e-12, 1.626e-12, 1.848e-12,
02066
            2.107e-12, 2.422e-12, 2.772e-12, 3.145e-12, 3.704e-12, 4.27e-12,
02067
            4.721e-12, 5.361e-12, 6.083e-12,
                                                       7.095e-12, 7.968e-12, 9.228e-12,
            1.048e-11, 1.187e-11, 1.336e-11, 1.577e-11, 1.772e-11, 2.017e-11,
02068
            2.25e-11, 2.63e-11, 2.911e-11, 3.356e-11, 3.82e-11, 4.173e-11, 4.811e-11, 5.254e-11, 5.839e-11, 6.187e-11, 6.805e-11, 7.118e-11,
02069
02071
            7.369e-11, 7.664e-11, 7.794e-11, 7.947e-11, 8.036e-11, 7.954e-11,
02072
            7.849e-11, 7.518e-11, 7.462e-11, 6.926e-11, 6.531e-11, 6.197e-11,
02073
            5.421e-11, 4.777e-11, 4.111e-11, 3.679e-11, 3.166e-11, 2.786e-11,
            2.436e-11, 2.144e-11, 1.859e-11, 1.628e-11, 1.414e-11, 1.237e-11,
02074
02075
            1.093e-11, 9.558e-12
02076
02077
02078
          static double h2o260[2001] = { .2752, .2732, .2749, .2676, .2667, .2545,
            .2497, .2327, .2218, .2036, .1825, .1694, .1497, .1353, .121, .1014, .09405, .07848, .07195, .06246, .05306, .04853, .04138,
02079
02080
            .03735, .03171, .02785, .02431, .02111, .01845, .0164, .0140
.01255, .01098, .009797, .008646, .007779, .006898, .006099,
02081
                                                                                   .01405.
02082
            .005453, .004909, .004413, .003959, .003581, .003199, .002871,
            .002583, .00233, .002086, .001874, .001684, .001512, .001361, .001225, .0011, 9.89e-4, 8.916e-4, 8.039e-4, 7.256e-4, 6.545e-4,
02084
02085
02086
            5.918e-4, 5.359e-4, 4.867e-4, 4.426e-4, 4.033e-4, 3.682e-4,
02087
            3.366e-4, 3.085e-4, 2.833e-4, 2.605e-4, 2.403e-4, 2.221e-4,
02088
            2.055e-4, 1.908e-4, 1.774e-4, 1.653e-4, 1.544e-4, 1.443e-4,
             1.351e-4, 1.267e-4, 1.19e-4, 1.119e-4, 1.053e-4, 9.922e-5,
            9.355e-5, 8.831e-5, 8.339e-5, 7.878e-5, 7.449e-5, 7.043e-5,
02090
02091
            6.664e-5, 6.307e-5, 5.969e-5, 5.654e-5, 5.357e-5, 5.075e-5
            4.81e-5, 4.56e-5, 4.322e-5, 4.102e-5, 3.892e-5, 3.696e-5, 3.511e-5, 3.339e-5, 3.177e-5, 3.026e-5, 2.886e-5, 2.756e-5, 2.636e-5, 2.527e-5, 2.427e-5, 2.337e-5, 2.257e-5, 2.185e-5,
02092
02093
02094
            2.127e-5, 2.08e-5, 2.041e-5, 2.013e-5, 2e-5, 1.997e-5, 2.009e-5,
            2.031e-5, 2.068e-5, 2.124e-5, 2.189e-5, 2.267e-5, 2.364e-5,
02096
02097
            2.463e-5, 2.618e-5, 2.774e-5, 2.937e-5, 3.144e-5, 3.359e-5,
02098
            3.695e-5, 4.002e-5, 4.374e-5, 4.947e-5, 5.431e-5, 6.281e-5,
02099
            7.169e-5, 8.157e-5, 9.728e-5, 1.079e-4, 1.337e-4, 1.442e-4,
            1.683e-4, 1.879e-4, 2.223e-4, 2.425e-4, 2.838e-4, 3.143e-4,
02100
02101
            3.527e-4, 4.012e-4, 4.237e-4, 4.747e-4, 5.057e-4, 5.409e-4,
            5.734e-4, 5.944e-4, 6.077e-4, 6.175e-4, 6.238e-4, 6.226e-4,
             6.248e-4, 6.192e-4, 6.098e-4, 5.818e-4, 5.709e-4, 5.465e-4,
02103
02104
            5.043e-4, 4.699e-4, 4.294e-4, 3.984e-4, 3.672e-4, 3.152e-4,
02105
            2.883e-4, 2.503e-4, 2.211e-4, 1.92e-4, 1.714e-4, 1.485e-4,
            1.358e-4, 1.156e-4, 1.021e-4, 8.887e-5, 7.842e-5, 7.12e-5, 6.186e-5, 5.73e-5, 4.792e-5, 4.364e-5, 3.72e-5, 3.28e-5, 2.946e-5, 2.591e-5, 2.261e-5, 2.048e-5, 1.813e-5, 1.63e-5,
02106
02107
02108
02109
            1.447e-5, 1.282e-5, 1.167e-5, 1.041e-5, 9.449e-6, 8.51e-6,
02110
            7.596e-6, 6.961e-6, 6.272e-6, 5.728e-6, 5.198e-6, 4.667e-6,
02111
            4.288e-6, 3.897e-6, 3.551e-6, 3.235e-6, 2.952e-6, 2.688e-6,
02112
            2.449e-6, 2.241e-6, 2.05e-6, 1.879e-6, 1.722e-6, 1.582e-6,
            1.456e-6, 1.339e-6, 1.236e-6, 1.144e-6, 1.06e-6, 9.83e-7, 9.149e-7, 8.535e-7, 7.973e-7, 7.466e-7, 6.999e-7, 6.574e-7,
02113
            6.18e-7, 5.821e-7, 5.487e-7, 5.18e-7, 4.896e-7, 4.631e-7, 4.386e-7, 4.16e-7, 3.945e-7, 3.748e-7, 3.562e-7, 3.385e-7,
02115
02116
            3.222e-7, 3.068e-7, 2.922e-7, 2.788e-7, 2.659e-7, 2.539e-7,
02117
            2.425e-7, 2.318e-7, 2.219e-7, 2.127e-7, 2.039e-7, 1.958e-7, 1.885e-7, 1.818e-7, 1.758e-7, 1.711e-7, 1.662e-7, 1.63e-7,
02118
02119
```

```
1.605e-7, 1.58e-7, 1.559e-7, 1.545e-7, 1.532e-7, 1.522e-7,
             1.51e-7, 1.495e-7, 1.465e-7, 1.483e-7, 1.469e-7, 1.448e-7, 1.444e-7, 1.436e-7, 1.426e-7, 1.431e-7, 1.425e-7, 1.445e-7,
02121
02122
             1.477e-7, 1.515e-7, 1.567e-7, 1.634e-7, 1.712e-7, 1.802e-7,
02123
             1.914e-7, 2.024e-7, 2.159e-7, 2.295e-7, 2.461e-7, 2.621e-7, 2.868e-7, 3.102e-7, 3.394e-7, 3.784e-7, 4.223e-7, 4.864e-7,
02124
02125
             5.501e-7, 6.039e-7, 7.193e-7, 7.728e-7, 9.514e-7, 1.073e-6,
             1.18e-6, 1.333e-6, 1.472e-6, 1.566e-6, 1.677e-6, 1.784e-6,
02127
02128
             1.904e-6, 1.953e-6, 2.02e-6, 2.074e-6, 2.128e-6, 2.162e-6,
             2.219e-6, 2.221e-6, 2.249e-6, 2.239e-6, 2.235e-6, 2.185e-6, 2.141e-6, 2.124e-6, 2.09e-6, 2.068e-6, 2.1e-6, 2.104e-6,
02129
02130
             2.142e-6, 2.181e-6, 2.257e-6, 2.362e-6, 2.5e-6, 2.664e-6, 2.884e-6, 3.189e-6, 3.48e-6, 3.847e-6, 4.313e-6, 4.79e-6, 5.25e-6, 5.989e-6, 6.692e-6, 7.668e-6, 8.52e-6, 9.606e-6,
02131
02132
02133
02134
             1.073e-5, 1.225e-5, 1.377e-5, 1.582e-5, 1.761e-5, 2.029e-5,
             2.284e-5, 2.602e-5, 2.94e-5, 3.483e-5, 3.928e-5, 4.618e-5, 5.24e-5, 6.132e-5, 7.183e-5, 8.521e-5, 9.111e-5, 1.07e-4,
02135
02136
             1.184e-4, 1.264e-4, 1.475e-4, 1.612e-4, 1.704e-4, 1.818e-4,
02137
             1.924e-4, 1.994e-4, 2.061e-4, 2.18e-4, 2.187e-4, 2.2e-4,
02139
             2.196e-4, 2.131e-4, 2.015e-4, 1.988e-4, 1.847e-4, 1.729e-4,
             1.597e-4, 1.373e-4, 1.262e-4, 1.087e-4, 9.439e-5, 8.061e-5,
02140
02141
             7.093e-5, 6.049e-5, 5.12e-5, 4.435e-5, 3.817e-5, 3.34e-5,
             2.927e-5, 2.573e-5, 2.291e-5, 2.04e-5, 1.827e-5, 1.636e-5,
02142
             1.463e-5, 1.309e-5, 1.17e-5, 1.047e-5, 9.315e-6, 8.328e-6, 7.458e-6, 6.665e-6, 5.94e-6, 5.316e-6, 4.752e-6, 4.252e-6,
02143
02144
             3.825e-6, 3.421e-6, 3.064e-6, 2.746e-6, 2.465e-6, 2.216e-6,
             1.99e-6, 1.79e-6, 1.609e-6, 1.449e-6, 1.306e-6, 1.177e-6,
02146
             1.063e-6, 9.607e-7, 8.672e-7, 7.855e-7, 7.118e-7, 6.46e-7,
02147
             5.871e-7, 5.34e-7, 4.868e-7, 4.447e-7, 4.068e-7, 3.729e-7, 3.423e-7, 3.151e-7, 2.905e-7, 2.686e-7, 2.484e-7, 2.306e-7,
02148
02149
             2.142e-7, 1.995e-7, 1.86e-7, 1.738e-7, 1.626e-7, 1.522e-7, 1.427e-7, 1.338e-7, 1.258e-7, 1.183e-7, 1.116e-7, 1.056e-7
02150
02151
             9.972e-8, 9.46e-8, 9.007e-8, 8.592e-8, 8.195e-8, 7.816e-8,
02152
02153
             7.483e-8, 7.193e-8, 6.892e-8, 6.642e-8, 6.386e-8, 6.154e-8,
             5.949e-8, 5.764e-8, 5.622e-8, 5.479e-8, 5.364e-8, 5.301e-8, 5.267e-8, 5.263e-8, 5.313e-8, 5.41e-8, 5.55e-8, 5.745e-8,
02154
02155
             6.003e-8, 6.311e-8, 6.713e-8, 7.173e-8, 7.724e-8, 8.368e-8,
02156
             9.121e-8, 9.986e-8, 1.097e-7, 1.209e-7, 1.338e-7, 1.486e-7,
02158
             1.651e-7, 1.837e-7, 2.048e-7, 2.289e-7, 2.557e-7, 2.857e-7,
             3.195e-7, 3.587e-7, 4.015e-7, 4.497e-7, 5.049e-7, 5.665e-7, 6.366e-7, 7.121e-7, 7.996e-7, 8.946e-7, 1.002e-6, 1.117e-6,
02159
02160
             1.262e-6, 1.416e-6, 1.611e-6, 1.807e-6, 2.056e-6, 2.351e-6,
02161
             2.769e-6, 3.138e-6, 3.699e-6, 4.386e-6, 5.041e-6, 6.074e-6, 6.812e-6, 7.79e-6, 8.855e-6, 1.014e-5, 1.095e-5, 1.245e-5,
02162
02163
             1.316e-5, 1.39e-5, 1.504e-5, 1.583e-5, 1.617e-5, 1.652e-5,
02164
02165
             1.713e-5, 1.724e-5, 1.715e-5, 1.668e-5, 1.629e-5, 1.552e-5,
02166
             1.478e-5, 1.34e-5, 1.245e-5, 1.121e-5, 9.575e-6, 8.956e-6,
             7.345e-6, 6.597e-6, 5.612e-6, 4.818e-6, 4.165e-6, 3.579e-6, 3.04le-6, 2.623e-6, 2.29e-6, 1.984e-6, 1.748e-6, 1.534e-6,
02167
02168
             1.369e-6, 1.219e-6, 1.092e-6, 9.8e-7, 8.762e-7, 7.896e-7, 7.104e-7, 6.364e-7, 5.691e-7, 5.107e-7, 4.575e-7, 4.09e-7
02169
             3.667e-7, 3.287e-7, 2.931e-7, 2.633e-7, 2.356e-7, 2.111e-7
02171
02172
             1.895e-7, 1.697e-7, 1.525e-7, 1.369e-7, 1.233e-7, 1.114e-7,
02173
             9.988e-8, 9.004e-8, 8.149e-8, 7.352e-8, 6.662e-8, 6.03e-8,
             5.479e-8, 4.974e-8, 4.532e-8, 4.129e-8, 3.781e-8, 3.462e-8,
02174
02175
             3.176e-8, 2.919e-8, 2.687e-8, 2.481e-8, 2.292e-8, 2.119e-8,
             1.967e-8, 1.828e-8, 1.706e-8, 1.589e-8, 1.487e-8, 1.393e-8,
             1.307e-8, 1.228e-8, 1.156e-8, 1.089e-8, 1.028e-8, 9.696e-9,
02177
02178
             9.159e-9, 8.658e-9, 8.187e-9, 7.746e-9, 7.34e-9, 6.953e-9,
             6.594e-9, 6.259e-9, 5.948e-9, 5.66e-9, 5.386e-9, 5.135e-9,
02179
02180
             4.903e-9, 4.703e-9, 4.515e-9, 4.362e-9, 4.233e-9, 4.117e-9,
             4.017e-9, 3.962e-9, 3.924e-9, 3.905e-9, 3.922e-9, 3.967e-9,
02181
02182
             4.046e-9, 4.165e-9, 4.32e-9, 4.522e-9, 4.769e-9, 5.083e-9,
             5.443e-9, 5.872e-9, 6.366e-9, 6.949e-9, 7.601e-9, 8.371e-9,
02183
02184
             9.22e-9, 1.02e-8, 1.129e-8, 1.251e-8, 1.393e-8, 1.542e-8,
02185
             1.72e-8, 1.926e-8, 2.152e-8, 2.392e-8, 2.678e-8, 3.028e-8,
             3.39e-8, 3.836e-8, 4.309e-8, 4.9e-8, 5.481e-8, 6.252e-8, 7.039e-8, 7.883e-8, 8.849e-8, 1.012e-7, 1.142e-7, 1.3e-7, 1.475e-7, 1.732e-7, 1.978e-7, 2.304e-7, 2.631e-7, 2.988e-7,
02186
02187
02188
             3.392e-7, 3.69e-7, 4.355e-7, 4.672e-7, 5.11e-7, 5.461e-7, 5.828e-7, 6.233e-7, 6.509e-7, 6.672e-7, 6.969e-7, 7.104e-7, 7.439e-7, 7.463e-7, 7.708e-7, 7.466e-7, 7.668e-7, 7.549e-7,
02190
02191
02192
             7.586e-7, 7.384e-7, 7.439e-7, 7.785e-7, 7.915e-7, 8.31e-7,
             8.745e-7, 9.558e-7, 1.038e-6, 1.173e-6, 1.304e-6, 1.452e-6,
02193
             1.67le-6, 1.93le-6, 2.239e-6, 2.578e-6, 3.032e-6, 3.334e-6, 3.98e-6, 4.3e-6, 4.518e-6, 5.32le-6, 5.508e-6, 6.21le-6, 6.59e-6,
02194
02195
             7.046e-6, 7.555e-6, 7.558e-6, 7.875e-6, 8.319e-6, 8.433e-6,
02196
02197
             8.59e-6, 8.503e-6, 8.304e-6, 8.336e-6, 7.739e-6, 7.301e-6,
02198
             6.827e-6, 6.078e-6, 5.551e-6, 4.762e-6, 4.224e-6, 3.538e-6,
             2.984e-6, 2.619e-6, 2.227e-6, 1.923e-6, 1.669e-6, 1.462e-6,
02199
             1.294e-6, 1.155e-6, 1.033e-6, 9.231e-7, 8.238e-7, 7.36e-7, 6.564e-7, 5.869e-7, 5.236e-7, 4.673e-7, 4.174e-7, 3.736e-7,
02200
             3.33e-7, 2.976e-7, 2.657e-7, 2.367e-7, 2.106e-7, 1.877e-7, 1.671e-7, 1.494e-7, 1.332e-7, 1.192e-7, 1.065e-7, 9.558e-8,
02202
02203
02204
             8.586e-8, 7.717e-8, 6.958e-8, 6.278e-8, 5.666e-8, 5.121e-8,
             4.647e-8, 4.213e-8, 3.815e-8, 3.459e-8, 3.146e-8, 2.862e-8, 2.604e-8, 2.375e-8, 2.162e-8, 1.981e-8, 1.817e-8, 1.67e-8,
02205
02206
```

```
02207
             1.537e-8, 1.417e-8, 1.31e-8, 1.215e-8, 1.128e-8, 1.05e-8,
             9.793e-9, 9.158e-9, 8.586e-9, 8.068e-9, 7.595e-9, 7.166e-9, 6.778e-9, 6.427e-9, 6.108e-9, 5.826e-9, 5.571e-9, 5.347e-9,
02208
02209
02210
             5.144e-9, 4.968e-9, 4.822e-9, 4.692e-9, 4.589e-9, 4.506e-9,
             4.467e-9, 4.44e-9, 4.466e-9, 4.515e-9, 4.718e-9, 4.729e-9,
02211
             4.937e-9, 5.249e-9, 5.466e-9, 5.713e-9, 6.03e-9, 6.436e-9,
02212
             6.741e-9, 7.33e-9, 7.787e-9, 8.414e-9, 8.908e-9, 9.868e-9,
02213
             1.069e-8, 1.158e-8, 1.253e-8, 1.3e-8, 1.409e-8, 1.47e-8,
02214
02215
             1.548e-8, 1.612e-8, 1.666e-8, 1.736e-8, 1.763e-8, 1.812e-8,
             1.852e-8, 1.923e-8, 1.897e-8, 1.893e-8, 1.888e-8, 1.868e-8, 1.895e-8, 1.899e-8, 1.876e-8, 1.96e-8, 2.02e-8, 2.121e-8,
02216
02217
02218
             2.239e-8, 2.379e-8, 2.526e-8, 2.766e-8, 2.994e-8, 3.332e-8,
02219
             3.703e-8, 4.158e-8, 4.774e-8, 5.499e-8, 6.355e-8, 7.349e-8,
             8.414e-8, 9.846e-8, 1.143e-7, 1.307e-7, 1.562e-7, 1.817e-7,
02220
02221
             2.011e-7, 2.192e-7, 2.485e-7, 2.867e-7, 3.035e-7, 3.223e-7
             3.443e-7, 3.617e-7, 3.793e-7, 3.793e-7, 3.839e-7, 4.081e-7, 4.117e-7, 4.085e-7, 3.92e-7, 3.851e-7, 3.754e-7, 3.49e-7, 3.229e-7, 2.978e-7, 2.691e-7, 2.312e-7, 2.029e-7, 1.721e-7,
02222
02223
02224
             1.472e-7, 1.308e-7, 1.132e-7, 9.736e-8, 8.458e-8, 7.402e-8,
02226
             6.534e-8, 5.811e-8, 5.235e-8, 4.762e-8, 4.293e-8, 3.896e-8,
             3.526e-8, 3.165e-8, 2.833e-8, 2.551e-8, 2.288e-8, 2.036e-8,
02227
             1.82e-8, 1.626e-8, 1.438e-8, 1.299e-8, 1.149e-8, 1.03e-8, 9.148e-9, 8.122e-9, 7.264e-9, 6.425e-9, 5.777e-9, 5.06e-9,
02228
02229
             4.502e-9, 4.013e-9, 3.567e-9, 3.145e-9, 2.864e-9, 2.553e-9, 2.311e-9, 2.087e-9, 1.886e-9, 1.716e-9, 1.556e-9, 1.432e-9,
02230
02231
             1.311e-9, 1.202e-9, 1.104e-9, 1.013e-9, 9.293e-10, 8.493e-10,
02232
             7.79e-10, 7.185e-10, 6.642e-10, 6.141e-10, 5.684e-10, 5.346e-10,
02233
02234
             5.032e-10, 4.725e-10, 4.439e-10, 4.176e-10, 3.93e-10, 3.714e-10,
             3.515e-10, 3.332e-10, 3.167e-10, 3.02e-10, 2.887e-10, 2.769e-10,
02235
             2.665e-10, 2.578e-10, 2.503e-10, 2.436e-10, 2.377e-10, 2.342e-10, 2.305e-10, 2.296e-10, 2.278e-10, 2.321e-10, 2.355e-10, 2.402e-10, 2.478e-10, 2.67e-10, 2.848e-10, 2.982e-10, 3.263e-10, 3.438e-10,
02236
02237
             3.649e-10, 3.829e-10, 4.115e-10, 4.264e-10, 4.473e-10, 4.63e-10,
02239
02240
             4.808e-10, 4.995e-10, 5.142e-10, 5.313e-10, 5.318e-10, 5.358e-10,
02241
             5.452e-10, 5.507e-10, 5.698e-10, 5.782e-10, 5.983e-10, 6.164e-10,
             6.532e-10, 6.811e-10, 7.624e-10, 8.302e-10, 9.067e-10, 9.937e-10,
02242
             1.104e-9, 1.221e-9, 1.361e-9, 1.516e-9, 1.675e-9, 1.883e-9, 2.101e-9, 2.349e-9, 2.614e-9, 2.92e-9, 3.305e-9, 3.724e-9,
02243
02245
             4.142e-9, 4.887e-9, 5.614e-9, 6.506e-9, 7.463e-9, 8.817e-9,
02246
             9.849e-9, 1.187e-8, 1.321e-8, 1.474e-8, 1.698e-8, 1.794e-8,
02247
             2.09e-8, 2.211e-8, 2.362e-8, 2.556e-8, 2.729e-8, 2.88e-8,
             3.046e-8, 3.167e-8, 3.367e-8, 3.457e-8, 3.59e-8, 3.711e-8,
02248
             3.826e-8, 4.001e-8, 4.211e-8, 4.315e-8, 4.661e-8, 5.01e-8, 5.249e-8, 5.84e-8, 6.628e-8, 7.512e-8, 8.253e-8, 9.722e-8, 1.067e-7, 1.153e-7, 1.347e-7, 1.428e-7, 1.577e-7, 1.694e-7, 1.833e-7, 1.938e-7, 2.108e-7, 2.059e-7, 2.157e-7, 2.185e-7,
02249
02251
02252
02253
             2.208e-7, 2.182e-7, 2.093e-7, 2.014e-7, 1.962e-7, 1.819e-7
             1.713e-7, 1.51e-7, 1.34e-7, 1.154e-7, 9.89e-8, 8.88e-8, 7.673e-8, 6.599e-8, 5.73e-8, 5.081e-8, 4.567e-8, 4.147e-8, 3.773e-8, 3.46e-8, 3.194e-8, 2.953e-8, 2.759e-8, 2.594e-8, 2.442e-8,
02254
02255
02256
             2.355e-8, 2.283e-8, 2.279e-8, 2.231e-8, 2.279e-8, 2.239e-8,
             2.21e-8, 2.309e-8, 2.293e-8, 2.352e-8, 2.415e-8, 2.43e-8, 2.426e-8, 2.465e-8, 2.5e-8, 2.496e-8, 2.465e-8, 2.445e-8,
02258
02259
             2.383e-8, 2.299e-8, 2.165e-8, 2.113e-8, 1.968e-8, 1.819e-8, 1.644e-8, 1.427e-8, 1.27e-8, 1.082e-8, 9.428e-9, 8.091e-9,
02260
02261
             6.958e-9, 5.988e-9, 5.246e-9, 4.601e-9, 4.098e-9, 3.664e-9,
02262
             3.287e-9, 2.942e-9, 2.656e-9, 2.364e-9, 2.118e-9, 1.903e-9,
             1.703e-9, 1.525e-9, 1.365e-9, 1.229e-9, 1.107e-9, 9.96e-10,
02264
             8.945e-10, 8.08e-10, 7.308e-10, 6.616e-10, 5.994e-10, 5.422e-10, 4.929e-10, 4.478e-10, 4.07e-10, 3.707e-10, 3.379e-10, 3.087e-10,
02265
02266
             2.823e-10, 2.592e-10, 2.385e-10, 2.201e-10, 2.038e-10, 1.897e-10,
02267
02268
             1.774e-10, 1.667e-10, 1.577e-10, 1.502e-10, 1.437e-10, 1.394e-10,
             1.358e-10, 1.324e-10, 1.329e-10, 1.324e-10, 1.36e-10, 1.39e-10,
             1.424e-10, 1.544e-10, 1.651e-10, 1.817e-10, 1.984e-10, 2.195e-10,
02270
02271
             2.438e-10, 2.7e-10, 2.991e-10, 3.322e-10, 3.632e-10, 3.957e-10,
02272
             4.36e-10, 4.701e-10, 5.03e-10, 5.381e-10, 5.793e-10, 6.19e-10,
             6.596e-10, 7.004e-10, 7.561e-10, 7.934e-10, 8.552e-10, 9.142e-10, 9.57e-10, 1.027e-9, 1.097e-9, 1.193e-9, 1.334e-9, 1.47e-9,
02273
02274
             1.636e-9, 1.871e-9, 2.122e-9, 2.519e-9, 2.806e-9, 3.203e-9,
             3.846e-9, 4.362e-9, 5.114e-9, 5.643e-9, 6.305e-9, 6.981e-9,
             7.983e-9, 8.783e-9, 9.419e-9, 1.017e-8, 1.063e-8, 1.121e-8,
02277
02278
             1.13e-8, 1.201e-8, 1.225e-8, 1.232e-8, 1.223e-8, 1.177e-8,
02279
             1.151e-8, 1.116e-8, 1.047e-8, 9.698e-9, 8.734e-9, 8.202e-9,
             7.04le-9, 6.074e-9, 5.172e-9, 4.468e-9, 3.913e-9, 3.414e-9, 2.975e-9, 2.65e-9, 2.406e-9, 2.173e-9, 2.009e-9, 1.86le-9, 1.727e-9, 1.612e-9, 1.514e-9, 1.43e-9, 1.362e-9, 1.333e-9,
02280
02281
02282
             1.288e-9, 1.249e-9, 1.238e-9, 1.228e-9, 1.217e-9, 1.202e-9,
02283
02284
             1.209e-9, 1.177e-9, 1.157e-9, 1.165e-9, 1.142e-9, 1.131e-9,
02285
             1.138e-9, 1.117e-9, 1.1e-9, 1.069e-9, 1.023e-9, 1.005e-9,
             9.159e-10, 8.863e-10, 7.865e-10, 7.153e-10, 6.247e-10, 5.846e-10, 5.133e-10, 4.36e-10, 3.789e-10, 3.335e-10, 2.833e-10, 2.483e-10,
02286
02287
             2.155e-10, 1.918e-10, 1.709e-10, 1.529e-10, 1.374e-10, 1.235e-10,
             1.108e-10, 9.933e-11, 8.932e-11, 8.022e-11, 7.224e-11, 6.52e-11, 5.896e-11, 5.328e-11, 4.813e-11, 4.365e-11, 3.961e-11, 3.594e-11,
02289
02290
02291
             3.266e-11,\ 2.967e-11,\ 2.701e-11,\ 2.464e-11,\ 2.248e-11,\ 2.054e-11,
             1.878e-11, 1.721e-11, 1.579e-11, 1.453e-11, 1.341e-11, 1.241e-11, 1.154e-11, 1.078e-11, 1.014e-11, 9.601e-12, 9.167e-12, 8.838e-12,
02292
02293
```

```
8.614e-12, 8.493e-12, 8.481e-12, 8.581e-12, 8.795e-12, 9.131e-12,
            9.601e-12, 1.021e-11, 1.097e-11, 1.191e-11, 1.303e-11, 1.439e-11, 1.601e-11, 1.778e-11, 1.984e-11, 2.234e-11, 2.474e-11, 2.766e-11,
02295
02296
02297
            3.085 e^{-11}, \ 3.415 e^{-11}, \ 3.821 e^{-11}, \ 4.261 e^{-11}, \ 4.748 e^{-11}, \ 5.323 e^{-11},
            5.935e-11, 6.619e-11, 7.418e-11, 8.294e-11, 9.26e-11, 1.039e-10, 1.156e-10, 1.297e-10, 1.46e-10, 1.641e-10, 1.858e-10, 2.1e-10,
02298
02299
            2.383e-10, 2.724e-10, 3.116e-10, 3.538e-10, 4.173e-10, 4.727e-10,
            5.503e-10, 6.337e-10, 7.32e-10, 8.298e-10, 9.328e-10,
02301
02302
            1.176e-9, 1.328e-9, 1.445e-9, 1.593e-9, 1.77e-9, 1.954e-9,
            2.175e-9, 2.405e-9, 2.622e-9, 2.906e-9, 3.294e-9, 3.713e-9, 3.98e-9, 4.384e-9, 4.987e-9, 5.311e-9, 5.874e-9, 6.337e-9, 7.027e-9, 7.39e-9, 7.769e-9, 8.374e-9, 8.605e-9, 9.165e-9, 9.415e-9, 9.511e-9, 9.704e-9, 9.588e-9, 9.45e-9, 9.086e-9,
02303
02304
02305
02306
            8.798e-9, 8.469e-9, 7.697e-9, 7.168e-9, 6.255e-9, 5.772e-9,
02307
02308
            4.97e-9, 4.271e-9, 3.653e-9, 3.154e-9, 2.742e-9, 2.435e-9,
            2.166e-9, 1.936e-9, 1.731e-9, 1.556e-9, 1.399e-9, 1.272e-9, 1.157e-9, 1.066e-9, 9.844e-10, 9.258e-10, 8.787e-10, 8.421e-10,
02309
02310
            8.083e-10, 8.046e-10, 8.067e-10, 8.181e-10, 8.325e-10, 8.517e-10,
02311
            9.151e-10, 9.351e-10, 9.677e-10, 1.071e-9, 1.126e-9, 1.219e-9,
02313
            1.297e-9, 1.408e-9, 1.476e-9, 1.517e-9, 1.6e-9, 1.649e-9,
            1.678e-9, 1.746e-9, 1.742e-9, 1.728e-9, 1.699e-9, 1.655e-9,
02314
02315
            1.561e-9, 1.48e-9, 1.451e-9, 1.411e-9, 1.171e-9, 1.106e-9,
            9.714e-10, 8.523e-10, 7.346e-10, 6.241e-10, 5.371e-10, 4.704e-10,
02316
            4.144e-10, 3.683e-10, 3.292e-10, 2.942e-10, 2.62e-10, 2.341e-10, 2.104e-10, 1.884e-10, 1.7e-10, 1.546e-10, 1.394e-10, 1.265e-10,
02317
02318
            1.14e-10, 1.019e-10, 9.279e-11, 8.283e-11, 7.458e-11, 6.668e-11,
            5.976e-11, 5.33e-11, 4.794e-11, 4.289e-11, 3.841e-11, 3.467e-11,
02320
02321
            3.13e-11, 2.832e-11, 2.582e-11, 2.356e-11, 2.152e-11, 1.97e-11,
02322
            1.808e-11, 1.664e-11, 1.539e-11, 1.434e-11, 1.344e-11, 1.269e-11,
02323
            1.209e-11, 1.162e-11, 1.129e-11, 1.108e-11, 1.099e-11, 1.103e-11,
            1.119e-11, 1.148e-11, 1.193e-11, 1.252e-11, 1.329e-11, 1.421e-11,
02324
02325
            1.555e-11, 1.685e-11, 1.839e-11, 2.054e-11, 2.317e-11, 2.571e-11,
            2.839e-11, 3.171e-11, 3.49e-11, 3.886e-11, 4.287e-11, 4.645e-11,
02326
02327
            5.047e-11, 5.592e-11, 6.109e-11, 6.628e-11, 7.381e-11, 8.088e-11,
02328
            8.966e-11, 1.045e-10, 1.12e-10, 1.287e-10, 1.486e-10, 1.662e-10,
            1.866e-10, 2.133e-10, 2.524e-10, 2.776e-10, 3.204e-10, 3.559e-10,
02329
            4.028e-10, 4.448e-10, 4.882e-10, 5.244e-10, 5.605e-10, 6.018e-10,
02330
            6.328e-10, 6.579e-10, 6.541e-10, 7.024e-10, 7.074e-10, 7.068e-10,
02332
            7.009e-10, 6.698e-10, 6.545e-10, 6.209e-10, 5.834e-10, 5.412e-10,
            5.001e-10, 4.231e-10, 3.727e-10, 3.211e-10, 2.833e-10, 2.447e-10,
02333
02334
            2.097e-10, 1.843e-10, 1.639e-10, 1.449e-10, 1.27e-10, 1.161e-10,
            1.033e-10, 9.282e-11, 8.407e-11, 7.639e-11, 7.023e-11, 6.474e-11, 6.142e-11, 5.76e-11, 5.568e-11, 5.472e-11, 5.39e-11, 5.455e-11, 5.54e-11, 5.587e-11, 6.23e-11, 6.49e-11, 6.868e-11, 7.382e-11,
02335
02336
02337
            8.022e-11, 8.372e-11, 9.243e-11, 1.004e-10, 1.062e-10, 1.13e-10, 1.176e-10, 1.244e-10, 1.279e-10, 1.298e-10, 1.302e-10, 1.312e-10,
02338
02339
02340
            1.295e-10, 1.244e-10, 1.211e-10, 1.167e-10, 1.098e-10, 9.927e-11,
02341
            8.854e-11, 8.011e-11, 7.182e-11, 5.923e-11, 5.212e-11, 4.453e-11,
            3.832e-11, 3.371e-11, 2.987e-11, 2.651e-11, 2.354e-11, 2.093e-11,
02342
            1.863e-11, 1.662e-11, 1.486e-11, 1.331e-11, 1.193e-11, 1.071e-11,
02343
            9.628e-12, 8.66e-12, 7.801e-12, 7.031e-12, 6.347e-12, 5.733e-12,
            5.182e-12, 4.695e-12, 4.26e-12, 3.874e-12, 3.533e-12, 3.235e-12,
02345
02346
            2.979e-12, 2.76e-12, 2.579e-12, 2.432e-12, 2.321e-12, 2.246e-12,
02347
            2.205e-12, 2.196e-12, 2.223e-12, 2.288e-12, 2.387e-12, 2.525e-12,
            2.704e-12, 2.925e-12, 3.191e-12, 3.508e-12, 3.876e-12, 4.303e-12,
02348
            4.793e-12, 5.347e-12, 5.978e-12, 6.682e-12, 7.467e-12, 8.34e-12,
02349
            9.293e-12, 1.035e-11, 1.152e-11, 1.285e-11, 1.428e-11, 1.586e-11,
            1.764e-11, 1.972e-11, 2.214e-11, 2.478e-11, 2.776e-11, 3.151e-11,
02351
            3.591e-11, 4.103e-11, 4.66e-11, 5.395e-11, 6.306e-11, 7.172e-11, 8.358e-11, 9.67e-11, 1.11e-10, 1.325e-10, 1.494e-10, 1.736e-10,
02352
02353
02354
            2.007e-10, 2.296e-10, 2.608e-10, 3.004e-10, 3.361e-10, 3.727e-10,
02355
            4.373e-10, 4.838e-10, 5.483e-10, 6.006e-10, 6.535e-10, 6.899e-10,
            7.687e-10, 8.444e-10, 8.798e-10, 9.135e-10, 9.532e-10, 9.757e-10,
            9.968e-10, 1.006e-9, 9.949e-10, 9.789e-10, 9.564e-10, 9.215e-10, 8.51e-10, 8.394e-10, 7.707e-10, 7.152e-10, 6.274e-10, 5.598e-10,
02357
02358
02359
            5.028e-10, 4.3e-10, 3.71e-10, 3.245e-10, 2.809e-10, 2.461e-10,
02360
            2.154e-10, 1.91e-10, 1.685e-10, 1.487e-10, 1.313e-10, 1.163e-10,
            1.031e-10, 9.172e-11, 8.221e-11, 7.382e-11, 6.693e-11, 6.079e-11,
02361
            5.581e-11, 5.167e-11, 4.81e-11, 4.506e-11, 4.255e-11, 4.083e-11, 3.949e-11, 3.881e-11, 3.861e-11, 3.858e-11, 3.951e-11, 4.045e-11,
02362
            4.24e-11, 4.487e-11, 4.806e-11, 5.133e-11, 5.518e-11, 5.919e-11,
02364
02365
            6.533e-11, 7.031e-11, 7.762e-11, 8.305e-11, 9.252e-11, 9.727e-11,
            1.045e-10, 1.117e-10, 1.2e-10, 1.275e-10, 1.341e-10, 1.362e-10, 1.438e-10, 1.45e-10, 1.455e-10, 1.455e-10, 1.434e-10, 1.381e-10,
02366
02367
02368
            1.301e-10, 1.276e-10, 1.163e-10, 1.089e-10, 9.911e-11, 8.943e-11,
            7.618e-11, 6.424e-11, 5.717e-11, 4.866e-11, 4.257e-11, 3.773e-11,
02369
            3.331e-11, 2.958e-11, 2.629e-11, 2.316e-11, 2.073e-11, 1.841e-11,
02370
02371
            1.635e-11, 1.464e-11, 1.31e-11, 1.16e-11, 1.047e-11, 9.408e-12,
            8.414e-12, 7.521e-12, 6.705e-12, 5.993e-12, 5.371e-12, 4.815e-12, 4.338e-12, 3.921e-12, 3.567e-12, 3.265e-12, 3.01e-12, 2.795e-12,
02372
02373
02374
            2.613e-12, 2.464e-12, 2.346e-12, 2.256e-12, 2.195e-12, 2.165e-12,
            2.166e-12, 2.198e-12, 2.262e-12, 2.364e-12, 2.502e-12, 2.682e-12,
            2.908e-12, 3.187e-12, 3.533e-12, 3.946e-12, 4.418e-12,
02376
                                                                                5.013e-12,
            5.708e-12, 6.379e-12, 7.43e-12, 8.39e-12, 9.51e-12, 1.078e-11,
02377
            1.259e-11, 1.438e-11, 1.63e-11, 1.814e-11, 2.055e-11, 2.348e-11, 2.664e-11, 2.956e-11, 3.3e-11, 3.677e-11, 4.032e-11, 4.494e-11, 4.951e-11, 5.452e-11, 6.014e-11, 6.5e-11, 6.915e-11, 7.45e-11,
02378
02379
02380
```

```
7.971e-11, 8.468e-11, 8.726e-11, 8.995e-11, 9.182e-11, 9.509e-11,
            9.333e-11, 9.386e-11, 9.457e-11, 9.21e-11, 9.019e-11, 8.68e-11, 8.298e-11, 7.947e-11, 7.46e-11, 7.082e-11, 6.132e-11, 5.855e-11,
02382
02383
02384
            5.073e-11, 4.464e-11, 3.825e-11, 3.375e-11, 2.911e-11, 2.535e-11,
02385
            2.16e-11, 1.907e-11, 1.665e-11, 1.463e-11, 1.291e-11, 1.133e-11, 9.997e-12, 8.836e-12, 7.839e-12, 6.943e-12, 6.254e-12, 5.6e-12,
02386
             5.029e-12, 4.529e-12, 4.102e-12, 3.737e-12, 3.428e-12, 3.169e-12,
             2.959e-12, 2.798e-12, 2.675e-12, 2.582e-12, 2.644e-12, 2.557e-12,
02388
02389
             2.614e-12, 2.717e-12, 2.874e-12, 3.056e-12, 3.187e-12, 3.631e-12,
            3.979e-12, 4.248e-12, 4.817e-12, 5.266e-12, 5.836e-12, 6.365e-12, 6.807e-12, 7.47e-12, 7.951e-12, 8.636e-12, 8.972e-12, 9.314e-12,
02390
02391
            9.445e-12, 1.003e-11, 1.013e-11, 9.937e-12, 9.729e-12, 9.064e-12,
02392
02393
             9.119e-12, 9.124e-12, 8.704e-12, 8.078e-12, 7.47e-12, 6.329e-12,
             5.674e-12, 4.808e-12, 4.119e-12, 3.554e-12, 3.103e-12, 2.731e-12,
02394
02395
            2.415e-12, 2.15e-12, 1.926e-12, 1.737e-12, 1.578e-12, 1.447e-12,
02396
            1.34e-12, 1.255e-12, 1.191e-12, 1.146e-12, 1.121e-12, 1.114e-12,
            1.126e-12, 1.156e-12, 1.207e-12, 1.278e-12, 1.372e-12, 1.49e-12, 1.633e-12, 1.805e-12, 2.01e-12, 2.249e-12, 2.528e-12, 2.852e-12,
02397
02398
             3.228e-12, 3.658e-12, 4.153e-12, 4.728e-12, 5.394e-12, 6.176e-12,
             7.126e-12, 8.188e-12, 9.328e-12, 1.103e-11, 1.276e-11, 1.417e-11,
02400
             1.615e-11, 1.84e-11, 2.155e-11, 2.429e-11, 2.826e-11, 3.222e-11,
02401
02402
            3.664e-11, 4.14e-11, 4.906e-11, 5.536e-11, 6.327e-11, 7.088e-11,
            8.316e-11, 9.242e-11, 1.07e-10, 1.223e-10, 1.341e-10, 1.553e-10, 1.703e-10, 1.9e-10, 2.022e-10, 2.233e-10, 2.345e-10, 2.438e-10,
02403
02404
02405
             2.546e-10, 2.599e-10, 2.661e-10, 2.703e-10, 2.686e-10, 2.662e-10,
             2.56e-10, 2.552e-10, 2.378e-10, 2.252e-10, 2.146e-10, 1.885e-10,
02406
02407
             1.668e-10, 1.441e-10, 1.295e-10, 1.119e-10, 9.893e-11, 8.687e-11,
02408
             7.678e-11, 6.685e-11, 5.879e-11, 5.127e-11, 4.505e-11, 3.997e-11,
02409
            3.511e-11
02410
02411
02412
          static double h2ofrn[2001] = { .01095, .01126, .01205, .01322,
                                                                                           .0143.
02413
            .01506, .01548, .01534, .01486, .01373, .01262, .01134, .01001,
02414
             .008702, .007475, .006481, .00548, .0046, .003833, .00311,
            .002543, .002049, .00168, .001374, .001046, 8.193e-4, 6.267e-4, 4.968e-4, 3.924e-4, 2.983e-4, 2.477e-4, 1.997e-4, 1.596e-4, 1.331e-4, 1.061e-4, 8.942e-5, 7.168e-5, 5.887e-5, 4.848e-5,
02415
02416
02417
            3.817e-5, 3.17e-5, 2.579e-5, 2.162e-5, 1.768e-5, 1.49e-5,
02419
             1.231e-5, 1.013e-5, 8.555e-6, 7.328e-6, 6.148e-6, 5.207e-6,
             4.387e-6, 3.741e-6, 3.22e-6, 2.753e-6, 2.346e-6, 1.985e-6,
02420
            1.716e-6, 1.475e-6, 1.286e-6, 1.122e-6, 9.661e-7, 8.284e-7, 7.057e-7, 6.119e-7, 5.29e-7, 4.571e-7, 3.948e-7, 3.432e-7, 2.983e-7, 2.589e-7, 2.265e-7, 1.976e-7, 1.704e-7, 1.456e-7, 1.26e-7, 1.101e-7, 9.648e-8, 8.415e-8, 7.34e-8, 6.441e-8,
02421
02422
02423
02424
             5.643e-8, 4.94e-8, 4.276e-8, 3.703e-8, 3.227e-8, 2.825e-8
02425
02426
            2.478e-8, 2.174e-8, 1.898e-8, 1.664e-8, 1.458e-8, 1.278e-8,
02427
            1.126e-8, 9.891e-9, 8.709e-9, 7.652e-9, 6.759e-9, 5.975e-9,
            5.31e-9, 4.728e-9, 4.214e-9, 3.792e-9, 3.463e-9, 3.226e-9, 2.992e-9, 2.813e-9, 2.749e-9, 2.809e-9, 2.913e-9, 3.037e-9, 3.413e-9, 3.738e-9, 4.189e-9, 4.808e-9, 5.978e-9, 7.088e-9,
02428
02429
02430
             8.071e-9, 9.61e-9, 1.21e-8, 1.5e-8, 1.764e-8, 2.221e-8, 2.898e-8,
            3.948e-8, 5.068e-8, 6.227e-8, 7.898e-8, 1.033e-7, 1.437e-7, 1.889e-7, 2.589e-7, 3.59e-7, 4.971e-7, 7.156e-7, 9.983e-7,
02432
02433
02434
             1.381e-6, 1.929e-6, 2.591e-6, 3.453e-6, 4.57e-6, 5.93e-6,
             7.552e-6, 9.556e-6, 1.183e-5, 1.425e-5, 1.681e-5, 1.978e-5,
02435
             2.335e-5, 2.668e-5, 3.022e-5, 3.371e-5, 3.715e-5, 3.967e-5,
02436
             4.06e-5, 4.01e-5, 3.809e-5, 3.491e-5, 3.155e-5, 2.848e-5,
             2.678e-5, 2.66e-5, 2.811e-5, 3.071e-5, 3.294e-5, 3.459e-5,
02438
02439
             3.569e-5, 3.56e-5, 3.434e-5, 3.186e-5, 2.916e-5, 2.622e-5
02440
            2.275e-5, 1.918e-5, 1.62e-5, 1.373e-5, 1.182e-5, 1.006e-5,
            8.556e-6, 7.26e-6, 6.107e-6, 5.034e-6, 4.211e-6, 3.426e-6,
02441
            2.865e-6, 2.446e-6, 1.998e-6, 1.628e-6, 1.242e-6, 1.005e-6,
02442
02443
             7.853e-7, 6.21e-7, 5.071e-7, 4.156e-7, 3.548e-7, 2.825e-7,
            2.261e-7, 1.916e-7, 1.51e-7, 1.279e-7, 1.059e-7, 9.14e-8, 7.707e-8, 6.17e-8, 5.311e-8, 4.263e-8, 3.518e-8, 2.961e-8
02444
02445
02446
            2.457e-8, 2.119e-8, 1.712e-8, 1.439e-8, 1.201e-8, 1.003e-8,
            8.564e-9, 7.199e-9, 6.184e-9, 5.206e-9, 4.376e-9, 3.708e-9, 3.157e-9, 2.725e-9, 2.361e-9, 2.074e-9, 1.797e-9, 1.562e-9,
02447
02448
             1.364e-9, 1.196e-9, 1.042e-9, 8.862e-10, 7.648e-10, 6.544e-10,
02449
             5.609e-10, 4.791e-10, 4.108e-10, 3.531e-10, 3.038e-10, 2.618e-10,
02451
             2.268e-10, 1.969e-10, 1.715e-10, 1.496e-10, 1.308e-10, 1.147e-10,
02452
            1.008e-10, 8.894e-11, 7.885e-11, 7.031e-11, 6.355e-11, 5.854e-11,
            1. 3.636 17, 3.636 17, 3.636 17, 3.636 17, 3.636 17, 3.636 17, 5.534e-11, 5.466e-11, 5.725e-11, 6.447e-11, 7.943e-11, 1.038e-10, 1.437e-10, 2.04e-10, 2.901e-10, 4.051e-10, 5.556e-10, 7.314e-10, 9.291e-10, 1.134e-9, 1.321e-9, 1.482e-9, 1.596e-9, 1.669e-9, 1.715e-9, 1.762e-9, 1.817e-9, 1.828e-9, 1.848e-9, 1.873e-9,
02453
02454
02455
02456
             1.902e-9, 1.894e-9, 1.864e-9, 1.841e-9, 1.797e-9, 1.704e-9,
02457
02458
             1.559e-9, 1.382e-9, 1.187e-9, 1.001e-9, 8.468e-10, 7.265e-10,
             6.521e-10, 6.381e-10, 6.66e-10, 7.637e-10, 9.705e-10, 1.368e-9,
02459
            1.856e-9, 2.656e-9, 3.954e-9, 5.96e-9, 8.72e-9, 1.247e-8,
02460
             1.781e-8, 2.491e-8, 3.311e-8, 4.272e-8, 5.205e-8, 6.268e-8
02461
             7.337e-8, 8.277e-8, 9.185e-8, 1.004e-7, 1.091e-7, 1.159e-7,
             1.188e-7, 1.175e-7, 1.124e-7, 1.033e-7, 9.381e-8, 8.501e-8,
02463
02464
             7.956e-8, 7.894e-8, 8.331e-8, 9.102e-8, 9.836e-8, 1.035e-7,
02465
            1.064e-7, 1.06e-7, 1.032e-7, 9.808e-8, 9.139e-8, 8.442e-8,
            7.64le-8, 6.88le-8, 6.16le-8, 5.404e-8, 4.804e-8, 4.446e-8, 4.328e-8, 4.259e-8, 4.42le-8, 4.673e-8, 4.985e-8, 5.335e-8,
02466
02467
```

```
5.796e-8, 6.542e-8, 7.714e-8, 8.827e-8, 1.04e-7, 1.238e-7,
             1.499e-7, 1.829e-7, 2.222e-7, 2.689e-7, 3.303e-7, 3.981e-7, 4.84e-7, 5.91e-7, 7.363e-7, 9.087e-7, 1.139e-6, 1.455e-6,
02469
02470
02471
             1.866e-6, 2.44e-6, 3.115e-6, 3.941e-6, 4.891e-6, 5.992e-6,
             7.111e-6, 8.296e-6, 9.21e-6, 9.987e-6, 1.044e-5, 1.073e-5, 1.092e-5, 1.106e-5, 1.138e-5, 1.171e-5, 1.186e-5, 1.186e-5,
02472
02473
             1.179e-5, 1.166e-5, 1.151e-5, 1.16e-5, 1.197e-5, 1.241e-5,
              1.268e-5, 1.26e-5, 1.184e-5, 1.063e-5, 9.204e-6, 7.584e-6,
02475
02476
              6.053e-6, 4.482e-6, 3.252e-6, 2.337e-6, 1.662e-6, 1.18e-6,
             8.15e-7, 5.95e-7, 4.354e-7, 3.302e-7, 2.494e-7, 1.93e-7, 1.545e-7, 1.25e-7, 1.039e-7, 8.602e-8, 7.127e-8, 5.897e-8, 4.838e-8, 4.018e-8, 3.28e-8, 2.72e-8, 2.307e-8, 1.972e-8,
02477
02478
02479
              1.654e-8, 1.421e-8, 1.174e-8, 1.004e-8, 8.739e-9, 7.358e-9,
02480
              6.242e-9, 5.303e-9, 4.567e-9, 3.94e-9, 3.375e-9, 2.864e-9,
02481
02482
             2.422e-9, 2.057e-9, 1.75e-9, 1.505e-9, 1.294e-9, 1.101e-9,
02483
              9.401e-10, 8.018e-10, 6.903e-10, 5.965e-10, 5.087e-10, 4.364e-10,
             3.759e-10, 3.247e-10, 2.809e-10, 2.438e-10, 2.123e-10, 1.853e-10,
02484
             1.622e-10, 1.426e-10, 1.26e-10, 1.125e-10, 1.022e-10, 9.582e-11, 9.388e-11, 9.801e-11, 1.08e-10, 1.276e-10, 1.551e-10, 1.903e-10,
02485
             2.291e-10, 2.724e-10, 3.117e-10, 3.4e-10, 3.562e-10, 3.625e-10,
              3.619e-10, 3.429e-10, 3.221e-10, 2.943e-10, 2.645e-10, 2.338e-10,
02488
02489
             2.062e-10, 1.901e-10, 1.814e-10, 1.827e-10, 1.906e-10, 1.984e-10,
              2.04 e^{-10}, \ 2.068 e^{-10}, \ 2.075 e^{-10}, \ 2.018 e^{-10}, \ 1.959 e^{-10}, \ 1.897 e^{-10}, \\
02490
             1.852e-10, 1.791e-10, 1.696e-10, 1.634e-10, 1.598e-10, 1.561e-10, 1.518e-10, 1.443e-10, 1.377e-10, 1.346e-10, 1.342e-10, 1.375e-10,
02491
02492
             1.525e-10, 1.767e-10, 2.108e-10, 2.524e-10, 2.981e-10, 3.477e-10,
              4.262e-10, 5.326e-10, 6.646e-10, 8.321e-10, 1.069e-9,
02494
02495
             1.743e-9, 2.216e-9, 2.808e-9, 3.585e-9, 4.552e-9, 5.907e-9,
             7.611e-9, 9.774e-9, 1.255e-8, 1.666e-8, 2.279e-8, 3.221e-8, 4.531e-8, 6.4e-8, 9.187e-8, 1.295e-7, 1.825e-7, 2.431e-7, 3.181e-7, 4.009e-7, 4.941e-7, 5.88e-7, 6.623e-7, 7.155e-7, 7.451e-7, 7.594e-7, 7.541e-7, 7.467e-7, 7.527e-7, 7.935e-7, 8.461e-7, 8.954e-7, 9.364e-7, 9.843e-7, 1.024e-6, 1.05e-6,
02496
02497
02498
02499
02500
02501
             1.059e-6, 1.074e-6, 1.072e-6, 1.043e-6, 9.789e-7, 8.803e-7
             7.662e-7, 6.378e-7, 5.133e-7, 3.958e-7, 2.914e-7, 2.144e-7, 1.57e-7, 1.14e-7, 8.47e-8, 6.2e-8, 4.657e-8, 3.559e-8, 2.813e-8,
02502
02503
             2.222e-8, 1.769e-8, 1.391e-8, 1.125e-8, 9.186e-9, 7.704e-9, 6.447e-9, 5.381e-9, 4.442e-9, 3.669e-9, 3.057e-9, 2.564e-9,
02504
02506
              2.153e-9, 1.784e-9, 1.499e-9, 1.281e-9, 1.082e-9, 9.304e-10,
              8.169e-10, 6.856e-10, 5.866e-10, 5.043e-10, 4.336e-10, 3.731e-10,
02507
             3.175e-10, 2.745e-10, 2.374e-10, 2.007e-10, 1.737e-10, 1.508e-10, 1.302e-10, 1.13e-10, 9.672e-11, 8.375e-11, 7.265e-11, 6.244e-11,
02508
02509
             5.343e-11, 4.654e-11, 3.975e-11, 3.488e-11, 3.097e-11, 2.834e-11, 2.649e-11, 2.519e-11, 2.462e-11, 2.443e-11, 2.44e-11, 2.398e-11,
02510
02512
             2.306e-11, 2.183e-11, 2.021e-11, 1.821e-11, 1.599e-11, 1.403e-11,
02513
             1.196e-11, 1.023e-11, 8.728e-12, 7.606e-12, 6.941e-12, 6.545e-12,
02514
              6.484e-12, 6.6e-12, 6.718e-12, 6.785e-12, 6.746e-12, 6.724e-12,
             6.764e-12, 6.995e-12, 7.144e-12, 7.32e-12, 7.33e-12, 7.208e-12, 6.789e-12, 6.09e-12, 5.337e-12, 4.62e-12, 4.037e-12, 3.574e-12, 3.311e-12, 3.346e-12, 3.566e-12, 3.836e-12, 4.076e-12, 4.351e-12, 4.691e-12, 5.114e-12, 5.427e-12, 6.167e-12, 7.436e-12, 8.842e-12,
02515
02516
02517
             1.038e-11, 1.249e-11, 1.54e-11, 1.915e-11, 2.48e-11, 3.256e-11,
02519
             4.339e-11, 5.611e-11, 7.519e-11, 1.037e-10, 1.409e-10, 1.883e-10, 2.503e-10, 3.38e-10, 4.468e-10, 5.801e-10, 7.335e-10, 8.98e-10, 1.11e-9, 1.363e-9, 1.677e-9, 2.104e-9, 2.681e-9, 3.531e-9,
02520
02521
02522
              4.621e-9, 6.106e-9, 8.154e-9, 1.046e-8, 1.312e-8, 1.607e-8,
02523
              1.948e-8, 2.266e-8, 2.495e-8, 2.655e-8, 2.739e-8, 2.739e-8,
             2.662e-8, 2.589e-8, 2.59e-8, 2.664e-8, 2.833e-8, 3.023e-8,
02525
02526
             3.305e-8, 3.558e-8, 3.793e-8, 3.961e-8, 4.056e-8, 4.102e-8,
02527
             4.025e-8, 3.917e-8, 3.706e-8, 3.493e-8, 3.249e-8, 3.096e-8,
             3.011e-8, 3.111e-8, 3.395e-8, 3.958e-8, 4.875e-8, 6.066e-8,
02528
             7.915e-8, 1.011e-7, 1.3e-7, 1.622e-7, 2.003e-7, 2.448e-7, 2.863e-7, 3.317e-7, 3.655e-7, 3.96e-7, 4.098e-7, 4.168e-7, 4.198e-7, 4.207e-7, 4.289e-7, 4.384e-7, 4.471e-7, 4.524e-7, 4.574e-7, 4.633e-7, 4.785e-7, 5.028e-7, 5.371e-7, 5.727e-7,
02529
02531
02532
02533
             5.955e-7, 5.998e-7, 5.669e-7, 5.082e-7, 4.397e-7, 3.596e-7,
             2.814e-7, 2.074e-7, 1.486e-7, 1.057e-7, 7.25e-8, 4.946e-8, 3.43e-8, 2.447e-8, 1.793e-8, 1.375e-8, 1.096e-8, 9.091e-9,
02534
02535
02536
              7.709e-9, 6.631e-9, 5.714e-9, 4.886e-9, 4.205e-9, 3.575e-9,
             3.07e-9, 2.631e-9, 2.284e-9, 2.002e-9, 1.745e-9, 1.509e-9,
             1.284e-9, 1.084e-9, 9.163e-10, 7.663e-10, 6.346e-10, 5.283e-10,
02538
02539
              4.354e-10, 3.59e-10, 2.982e-10, 2.455e-10, 2.033e-10, 1.696e-10,
02540
             1.432 e^{-10}, \ 1.211 e^{-10}, \ 1.02 e^{-10}, \ 8.702 e^{-11}, \ 7.38 e^{-11}, \ 6.293 e^{-11},
02541
             5.343e-11, 4.532e-11, 3.907e-11, 3.365e-11, 2.945e-11, 2.558e-11,
             2.192e-11, 1.895e-11, 1.636e-11, 1.42e-11, 1.228e-11, 1.063e-11, 9.348e-12, 8.2e-12, 7.231e-12, 6.43e-12, 5.702e-12, 5.052e-12,
02542
02543
             4.469e-12, 4e-12, 3.679e-12, 3.387e-12, 3.197e-12, 3.158e-12,
02544
02545
             3.327e-12, 3.675e-12, 4.292e-12, 5.437e-12, 7.197e-12, 1.008e-11,
02546
             1.437e-11, 2.035e-11, 2.905e-11, 4.062e-11, 5.528e-11, 7.177e-11,
             9.064e-11, 1.109e-10, 1.297e-10, 1.473e-10, 1.652e-10, 1.851e-10,
02547
             2.079e-10, 2.313e-10, 2.619e-10, 2.958e-10, 3.352e-10, 3.796e-10, 4.295e-10, 4.923e-10, 5.49e-10, 5.998e-10, 6.388e-10, 6.645e-10,
02548
02550
              6.712e-10, 6.549e-10, 6.38e-10, 6.255e-10, 6.253e-10, 6.459e-10,
02551
              6.977e-10, 7.59e-10, 8.242e-10, 8.92e-10, 9.403e-10, 9.701e-10,
02552
             9.483e-10, 9.135e-10, 8.617e-10, 7.921e-10, 7.168e-10, 6.382e-10,
             5.677e-10, 5.045e-10, 4.572e-10, 4.312e-10, 4.145e-10, 4.192e-10, 4.541e-10, 5.368e-10, 6.771e-10, 8.962e-10, 1.21e-9, 1.659e-9,
02553
02554
```

```
2.33e-9, 3.249e-9, 4.495e-9, 5.923e-9, 7.642e-9, 9.607e-9,
            1.178e-8, 1.399e-8, 1.584e-8, 1.73e-8, 1.816e-8, 1.87e-8, 1.868e-8, 1.87e-8, 1.884e-8, 1.99e-8, 2.15e-8, 2.258e-8,
02556
02557
             2.364e-8, 2.473e-8, 2.602e-8, 2.689e-8, 2.731e-8, 2.816e-8,
02558
02559
            2.859e-8, 2.839e-8, 2.703e-8, 2.451e-8, 2.149e-8, 1.787e-8,
             1.449e-8, 1.111e-8, 8.282e-9, 6.121e-9, 4.494e-9, 3.367e-9,
02560
             2.487e-9, 1.885e-9, 1.503e-9, 1.249e-9, 1.074e-9, 9.427e-10,
             8.439e-10, 7.563e-10, 6.772e-10, 6.002e-10, 5.254e-10, 4.588e-10,
02562
02563
            3.977e-10, 3.449e-10, 3.003e-10, 2.624e-10, 2.335e-10, 2.04e-10,
02564
            1.771e-10, 1.534e-10, 1.296e-10, 1.097e-10, 9.173e-11, 7.73e-11,
             6.547e-11, 5.191e-11, 4.198e-11, 3.361e-11, 2.732e-11, 2.244e-11,
02565
02566
            1.791e-11, 1.509e-11, 1.243e-11, 1.035e-11, 8.969e-12, 7.394e-12,
             6.323e-12, 5.282e-12, 4.543e-12, 3.752e-12, 3.14e-12, 2.6e-12,
02567
02568
             2.194e-12, 1.825e-12, 1.511e-12, 1.245e-12, 1.024e-12, 8.539e-13,
02569
             7.227e-13, 6.102e-13, 5.189e-13, 4.43e-13, 3.774e-13, 3.236e-13,
            2.8e-13, 2.444e-13, 2.156e-13, 1.932e-13, 1.775e-13, 1.695e-13, 1.672e-13, 1.704e-13, 1.825e-13, 2.087e-13, 2.614e-13, 3.377e-13,
02570
02571
            4.817e-13, 6.989e-13, 1.062e-12, 1.562e-12, 2.288e-12, 3.295e-12, 4.55e-12, 5.965e-12, 7.546e-12, 9.395e-12, 1.103e-11, 1.228e-11, 1.318e-11, 1.38e-11, 1.39e-11, 1.358e-11, 1.336e-11,
02572
02574
             1.342e-11, 1.356e-11, 1.424e-11, 1.552e-11, 1.73e-11, 1.951e-11
02575
02576
            2.128e-11, 2.249e-11, 2.277e-11, 2.226e-11, 2.111e-11, 1.922e-11,
            1.775e-11, 1.661e-11, 1.547e-11, 1.446e-11, 1.323e-11, 1.21e-11, 1.054e-11, 9.283e-12, 8.671e-12, 8.67e-12, 9.429e-12, 1.062e-11, 1.255e-11, 1.506e-11, 1.818e-11, 2.26e-11, 2.831e-11, 3.723e-11,
02577
02578
             5.092e-11, 6.968e-11, 9.826e-11, 1.349e-10, 1.87e-10, 2.58e-10,
02580
             3.43e-10, 4.424e-10, 5.521e-10, 6.812e-10, 8.064e-10,
02581
                                                                                   9.109e-10.
02582
            9.839e-10, 1.028e-9, 1.044e-9, 1.029e-9, 1.005e-9, 1.002e-9,
            1.038e-9, 1.122e-9, 1.233e-9, 1.372e-9, 1.524e-9, 1.665e-9, 1.804e-9, 1.908e-9, 2.015e-9, 2.117e-9, 2.219e-9, 2.336e-9,
02583
02584
            2.531e-9, 2.805e-9, 3.189e-9, 3.617e-9, 4.208e-9, 4.911e-9, 5.619e-9, 6.469e-9, 7.188e-9, 7.957e-9, 8.503e-9, 9.028e-9,
02585
             9.571e-9, 9.99e-9, 1.055e-8, 1.102e-8, 1.132e-8, 1.141e-8,
02587
02588
            1.145e-8, 1.145e-8, 1.176e-8, 1.224e-8, 1.304e-8, 1.388e-8,
            1.445e-8, 1.453e-8, 1.368e-8, 1.22e-8, 1.042e-8, 8.404e-9, 6.403e-9, 4.643e-9, 3.325e-9, 2.335e-9, 1.638e-9, 1.19e-9,
02589
02590
            9.161e-10, 7.412e-10, 6.226e-10, 5.516e-10, 5.068e-10, 4.831e-10,
4.856e-10, 5.162e-10, 5.785e-10, 6.539e-10, 7.485e-10, 8.565e-10,
02591
02593
             9.534e-10, 1.052e-9, 1.115e-9, 1.173e-9, 1.203e-9, 1.224e-9,
02594
             1.243e-9, 1.248e-9, 1.261e-9, 1.265e-9, 1.25e-9, 1.217e-9,
02595
            1.176e-9, 1.145e-9, 1.153e-9, 1.199e-9, 1.278e-9, 1.366e-9,
            1.426e-9, 1.444e-9, 1.365e-9, 1.224e-9, 1.051e-9, 8.539e-10, 6.564e-10, 4.751e-10, 3.404e-10, 2.377e-10, 1.631e-10, 1.114e-10, 7.87e-11, 5.793e-11, 4.284e-11, 3.3e-11, 2.62e-11, 2.152e-11,
02596
02597
            1.777e-11, 1.496e-11, 1.242e-11, 1.037e-11, 8.725e-12, 7.004e-12, 5.718e-12, 4.769e-12, 3.952e-12, 3.336e-12, 2.712e-12, 2.213e-12,
02599
02600
02601
            1.803e-12, 1.492e-12, 1.236e-12, 1.006e-12, 8.384e-13, 7.063e-13,
            5.879e-13, 4.93e-13, 4.171e-13, 3.569e-13, 3.083e-13, 2.688e-13, 2.333e-13, 2.035e-13, 1.82e-13, 1.682e-13, 1.635e-13, 1.674e-13,
02602
02603
02604
             1.769e-13, 2.022e-13, 2.485e-13, 3.127e-13, 4.25e-13, 5.928e-13,
             8.514e-13, 1.236e-12, 1.701e-12, 2.392e-12, 3.231e-12, 4.35e-12,
            5.559e-12, 6.915e-12, 8.519e-12, 1.013e-11, 1.146e-11, 1.24e-11,
02606
02607
            1.305e-11, 1.333e-11, 1.318e-11, 1.263e-11, 1.238e-11, 1.244e-11,
02608
            1.305 e-11, \ 1.432 e-11, \ 1.623 e-11, \ 1.846 e-11, \ 2.09 e-11, \ 2.328 e-11,
            2.526e-11, 2.637e-11, 2.702e-11, 2.794e-11, 2.889e-11, 2.989e-11,
02609
            3.231e-11, 3.68e-11, 4.375e-11, 5.504e-11, 7.159e-11, 9.502e-11, 1.279e-10, 1.645e-10, 2.098e-10, 2.618e-10, 3.189e-10, 3.79e-10,
02610
             4.303e-10, 4.753e-10, 5.027e-10, 5.221e-10, 5.293e-10, 5.346e-10,
02612
02613
             5.467e-10, 5.796e-10, 6.2e-10, 6.454e-10, 6.705e-10, 6.925e-10,
             7.233e-10, 7.35e-10, 7.538e-10, 7.861e-10, 8.077e-10, 8.132e-10,
02614
             7.749e-10, 7.036e-10, 6.143e-10, 5.093e-10, 4.089e-10, 3.092e-10,
02615
            2.299e-10, 1.705e-10, 1.277e-10, 9.723e-11, 7.533e-11, 6.126e-11, 5.154e-11, 4.428e-11, 3.913e-11, 3.521e-11, 3.297e-11, 3.275e-11,
02616
            3.46e-11, 3.798e-11, 4.251e-11, 4.745e-11, 5.232e-11, 5.606e-11, 5.82e-11, 5.88e-11, 5.79e-11, 5.661e-11, 5.491e-11, 5.366e-11,
02618
02619
02620
            5.341e-11, 5.353e-11, 5.336e-11, 5.293e-11, 5.248e-11, 5.235e-11,
02621
            5.208 e^{-11}, \ 5.322 e^{-11}, \ 5.521 e^{-11}, \ 5.725 e^{-11}, \ 5.827 e^{-11}, \ 5.685 e^{-11},
            5.245e-11, 4.612e-11, 3.884e-11, 3.129e-11, 2.404e-11, 1.732e-11,
02622
            1.223e-11, 8.574e-12, 5.888e-12, 3.986e-12, 2.732e-12, 1.948e-12,
02623
             1.414e-12, 1.061e-12, 8.298e-13, 6.612e-13, 5.413e-13, 4.472e-13,
             3.772e-13, 3.181e-13, 2.645e-13, 2.171e-13, 1.778e-13,
                                                                                    1.464e-13,
02625
02626
            1.183e-13, 9.637e-14, 7.991e-14, 6.668e-14, 5.57e-14, 4.663e-14,
            3.848e-14, 3.233e-14, 2.706e-14, 2.284e-14, 1.944e-14, 1.664e-14, 1.43e-14, 1.233e-14, 1.066e-14, 9.234e-15, 8.023e-15, 6.993e-15,
02627
02628
             6.119e-15, 5.384e-15, 4.774e-15, 4.283e-15, 3.916e-15, 3.695e-15,
02629
             3.682e-15, 4.004e-15, 4.912e-15, 6.853e-15, 1.056e-14, 1.712e-14,
02630
            2.804e-14, 4.516e-14, 7.113e-14, 1.084e-13, 1.426e-13, 1.734e-13,
02631
02632
             1.978e-13, 2.194e-13, 2.388e-13, 2.489e-13, 2.626e-13, 2.865e-13,
            3.105e-13, 3.387e-13, 3.652e-13, 3.984e-13, 4.398e-13, 4.906e-13, 5.55e-13, 6.517e-13, 7.813e-13, 9.272e-13, 1.164e-12, 1.434e-12,
02633
02634
            1.849e-12, 2.524e-12, 3.328e-12, 4.523e-12, 6.108e-12, 8.207e-12,
02635
             1.122e-11, 1.477e-11, 1.9e-11, 2.412e-11, 2.984e-11, 3.68e-11,
             4.353e-11, 4.963e-11, 5.478e-11, 5.903e-11, 6.233e-11, 6.483e-11,
02637
02638
             6.904e-11, 7.569e-11, 8.719e-11, 1.048e-10, 1.278e-10, 1.557e-10,
02639
            1.869 e^{-10},\ 2.218 e^{-10},\ 2.61 e^{-10},\ 2.975 e^{-10},\ 3.371 e^{-10},\ 3.746 e^{-10},
            4.065e-10, 4.336e-10, 4.503e-10, 4.701e-10, 4.8e-10, 4.917e-10, 5.038e-10, 5.128e-10, 5.143e-10, 5.071e-10, 5.019e-10, 5.025e-10,
02640
02641
```

```
5.183e-10, 5.496e-10, 5.877e-10, 6.235e-10, 6.42e-10, 6.234e-10,
            5.698e-10, 4.916e-10, 4.022e-10, 3.126e-10, 2.282e-10, 1.639e-10,
02643
02644
            1.142e-10, 7.919e-11, 5.69e-11, 4.313e-11, 3.413e-11, 2.807e-11,
02645
            2.41e-11, 2.166e-11, 2.024e-11, 1.946e-11, 1.929e-11, 1.963e-11,
02646
            2.035e-11, 2.162e-11, 2.305e-11, 2.493e-11, 2.748e-11, 3.048e-11, 3.413e-11, 3.754e-11, 4.155e-11, 4.635e-11, 5.11e-11, 5.734e-11,
02647
            6.338e-11, 6.99e-11, 7.611e-11, 8.125e-11, 8.654e-11, 8.951e-11,
            9.182e-11, 9.31e-11, 9.273e-11, 9.094e-11, 8.849e-11, 8.662e-11,
02649
02650
            8.67e-11, 8.972e-11, 9.566e-11, 1.025e-10, 1.083e-10, 1.111e-10,
            1.074e-10, 9.771e-11, 8.468e-11, 6.958e-11, 5.47e-11, 4.04e-11, 2.94e-11, 2.075e-11, 1.442e-11, 1.01e-11, 7.281e-12, 5.409e-12, 4.138e-12, 3.304e-12, 2.784e-12, 2.473e-12, 2.273e-12, 2.186e-12,
02651
02652
02653
            2.118e-12, 2.066e-12, 1.958e-12, 1.818e-12, 1.675e-12, 1.509e-12,
02654
            1.349e-12, 1.171e-12, 9.838e-13, 8.213e-13, 6.765e-13, 5.378e-13,
02655
02656
            4.161e-13, 3.119e-13, 2.279e-13, 1.637e-13, 1.152e-13, 8.112e-14,
            5.919e-14, 4.47e-14, 3.492e-14, 2.811e-14, 2.319e-14, 1.948e-14, 1.66e-14, 1.432e-14, 1.251e-14, 1.109e-14, 1.006e-14, 9.45e-15,
02657
02658
            9.384e-15, 1.012e-14, 1.216e-14, 1.636e-14, 2.305e-14, 3.488e-14,
02659
            5.572e-14, 8.479e-14, 1.265e-13, 1.905e-13, 2.73e-13, 3.809e-13,
02660
            4.955e-13, 6.303e-13, 7.861e-13, 9.427e-13, 1.097e-12, 1.212e-12,
            1.328e-12, 1.415e-12, 1.463e-12, 1.495e-12, 1.571e-12, 1.731e-12,
02662
02663
            1.981e-12, 2.387e-12, 2.93e-12, 3.642e-12, 4.584e-12, 5.822e-12,
02664
            7.278e-12, 9.193e-12, 1.135e-11, 1.382e-11, 1.662e-11, 1.958e-11,
            2.286e-11, 2.559e-11, 2.805e-11, 2.988e-11, 3.106e-11, 3.182e-11,
02665
            2.200e-11, 2.358e-11, 2.308e-11, 2.308e-11, 3.100e-11, 3.102e-11, 3.258e-11, 3.368e-11, 3.58e-11, 3.58e-11, 3.688e-11, 3.8e-11, 3.929e-11, 4.062e-11, 4.186e-11, 4.293e-11, 4.48e-11, 4.643e-11, 4.704e-11, 4.571e-11, 4.206e-11, 3.715e-11, 3.131e-11, 2.541e-11,
02666
02668
02669
            1.978e-11, 1.508e-11, 1.146e-11, 8.7e-12, 6.603e-12, 5.162e-12,
02670
            4.157e-12, 3.408e-12, 2.829e-12, 2.405e-12, 2.071e-12, 1.826e-12,
02671
            1.648e-12, 1.542e-12, 1.489e-12, 1.485e-12, 1.493e-12, 1.545e-12,
            1.637e-12, 1.814e-12, 2.061e-12, 2.312e-12, 2.651e-12, 3.03e-12,
02672
            3.46e-12, 3.901e-12, 4.306e-12, 4.721e-12, 5.008e-12, 5.281e-12,
            5.541e-12, 5.791e-12, 6.115e-12, 6.442e-12, 6.68e-12, 6.791e-12,
02674
02675
            6.831e-12, 6.839e-12, 6.946e-12, 7.128e-12, 7.537e-12, 8.036e-12,
            8.392e-12, 8.526e-12, 8.11e-12, 7.325e-12, 6.329e-12, 5.183e-12, 4.081e-12, 2.985e-12, 2.141e-12, 1.492e-12, 1.015e-12, 6.684e-13,
02676
02677
02678
            4.414e-13, 2.987e-13, 2.038e-13, 1.391e-13, 9.86e-14, 7.24e-14,
            5.493e-14, 4.288e-14, 3.427e-14, 2.787e-14, 2.296e-14, 1.909e-14,
02680
            1.598e-14, 1.344e-14, 1.135e-14, 9.616e-15, 8.169e-15, 6.957e-15,
            5.938e-15, 5.08e-15, 4.353e-15, 3.738e-15, 3.217e-15, 2.773e-15,
02681
            2.397e-15, 2.077e-15, 1.805e-15, 1.575e-15, 1.382e-15, 1.221e-15, 1.09e-15, 9.855e-16, 9.068e-16, 8.537e-16, 8.27e-16, 8.29e-16,
02682
02683
            8.634e-16, 9.359e-16, 1.055e-15, 1.233e-15, 1.486e-15, 1.839e-15, 2.326e-15, 2.998e-15, 3.934e-15, 5.256e-15, 7.164e-15, 9.984e-15,
02684
02685
            1.427e-14, 2.099e-14, 3.196e-14, 5.121e-14, 7.908e-14, 1.131e-13,
02687
            1.602e-13, 2.239e-13, 3.075e-13, 4.134e-13, 5.749e-13,
02688
            1.071e-12, 1.464e-12, 2.032e-12, 2.8e-12, 3.732e-12, 4.996e-12,
            6.483e-12, 8.143e-12, 1.006e-11, 1.238e-11, 1.484e-11, 1.744e-11, 2.02e-11, 2.274e-11, 2.562e-11, 2.848e-11, 3.191e-11, 3.617e-11, 4.081e-11, 4.577e-11, 4.937e-11, 5.204e-11, 5.401e-11, 5.462e-11, 5.507e-11, 5.51e-11, 5.605e-11, 5.686e-11, 5.739e-11, 5.766e-11,
02689
02690
02691
            5.74e-11, 5.754e-11, 5.761e-11, 5.777e-11, 5.712e-11, 5.51e-11,
02693
02694
            5.088e-11, 4.438e-11, 3.728e-11, 2.994e-11, 2.305e-11, 1.715e-11,
02695
            1.256 e^{-11}, \ 9.208 e^{-12}, \ 6.745 e^{-12}, \ 5.014 e^{-12}, \ 3.785 e^{-12}, \ 2.9 e^{-12},
            2.239e-12, 1.757e-12, 1.414e-12, 1.142e-12, 9.482e-13, 8.01e-13,
02696
02697
            6.961e-13, 6.253e-13, 5.735e-13, 5.433e-13, 5.352e-13, 5.493e-13,
            5.706e-13, 6.068e-13, 6.531e-13, 7.109e-13, 7.767e-13, 8.59e-13,
            9.792e-13, 1.142e-12, 1.371e-12, 1.65e-12, 1.957e-12, 2.302e-12,
02699
02700
            2.705e-12, 3.145e-12, 3.608e-12, 4.071e-12, 4.602e-12, 5.133e-12,
02701
            5.572e-12, 5.987e-12, 6.248e-12, 6.533e-12, 6.757e-12, 6.935e-12,
            7.224e-12, 7.422e-12, 7.538e-12, 7.547e-12, 7.495e-12, 7.543e-12, 7.725e-12, 8.139e-12, 8.627e-12, 9.146e-12, 9.443e-12, 9.318e-12,
02702
02703
            8.649e-12, 7.512e-12, 6.261e-12, 4.915e-12, 3.647e-12, 2.597e-12,
            1.785e-12, 1.242e-12, 8.66e-13, 6.207e-13, 4.61e-13, 3.444e-13, 2.634e-13, 2.1e-13, 1.725e-13, 1.455e-13, 1.237e-13, 1.085e-13,
02705
02706
02707
            9.513e-14, 7.978e-14, 6.603e-14, 5.288e-14, 4.084e-14, 2.952e-14,
02708
            2.157e-14, 1.593e-14, 1.199e-14, 9.267e-15, 7.365e-15, 6.004e-15,
02709
            4.995e-15, 4.218e-15, 3.601e-15, 3.101e-15, 2.692e-15, 2.36e-15,
02710
            2.094e-15, 1.891e-15, 1.755e-15, 1.699e-15, 1.755e-15, 1.987e-15,
            2.506e-15, 3.506e-15, 5.289e-15, 8.311e-15, 1.325e-14, 2.129e-14,
02712
            3.237e-14, 4.595e-14, 6.441e-14, 8.433e-14, 1.074e-13,
                                                                                  1.383e-13.
02713
            1.762e-13, 2.281e-13, 2.831e-13, 3.523e-13, 4.38e-13, 5.304e-13,
            6.29e-13, 7.142e-13, 8.032e-13, 8.934e-13, 9.888e-13, 1.109e-12, 1.261e-12, 1.462e-12, 1.74e-12, 2.099e-12, 2.535e-12, 3.008e-12,
02714
02715
02716
            3.462e-12, 3.856e-12, 4.098e-12, 4.239e-12, 4.234e-12, 4.132e-12,
            3.986e-12, 3.866e-12, 3.829e-12, 3.742e-12, 3.705e-12, 3.694e-12,
02717
            3.765e-12, 3.849e-12, 3.929e-12, 4.056e-12, 4.092e-12, 4.047e-12,
02718
02719
            3.792e-12, 3.407e-12, 2.953e-12, 2.429e-12, 1.931e-12, 1.46e-12,
            1.099e-12, 8.199e-13, 6.077e-13, 4.449e-13, 3.359e-13, 2.524e-13, 1.881e-13, 1.391e-13, 1.02e-13, 7.544e-14, 5.555e-14, 4.22e-14,
02720
02721
            3.321e-14, 2.686e-14, 2.212e-14, 1.78e-14, 1.369e-14, 1.094e-14, 9.13e-15, 8.101e-15, 7.828e-15, 8.393e-15, 1.012e-14, 1.259e-14,
02722
             1.538e-14, 1.961e-14, 2.619e-14, 3.679e-14, 5.049e-14, 6.917e-14,
02724
02725
            8.88e-14, 1.115e-13, 1.373e-13, 1.619e-13, 1.878e-13, 2.111e-13,
02726
            2.33e-13,\ 2.503e-13,\ 2.613e-13,\ 2.743e-13,\ 2.826e-13,\ 2.976e-13,
            3.162e-13, 3.36e-13, 3.491e-13, 3.541e-13, 3.595e-13, 3.608e-13, 3.709e-13, 3.869e-13, 4.12e-13, 4.366e-13, 4.504e-13, 4.379e-13,
02727
02728
```

```
3.955e-13, 3.385e-13, 2.741e-13, 2.089e-13, 1.427e-13, 9.294e-14,
            5.775e-14, 3.565e-14, 2.21e-14, 1.398e-14, 9.194e-15, 6.363e-15, 4.644e-15, 3.55e-15, 2.808e-15, 2.274e-15, 1.871e-15, 1.557e-15,
02730
02731
02732
            1.308e-15, 1.108e-15, 9.488e-16, 8.222e-16, 7.238e-16, 6.506e-16,
            6.008e-16, 5.742e-16, 5.724e-16, 5.991e-16, 6.625e-16, 7.775e-16, 9.734e-16, 1.306e-15, 1.88e-15, 2.879e-15, 4.616e-15, 7.579e-15,
02733
02734
            1.248e-14, 2.03e-14, 3.244e-14, 5.171e-14, 7.394e-14, 9.676e-14,
            1.199e-13, 1.467e-13, 1.737e-13, 2.02e-13, 2.425e-13, 3.016e-13,
02736
02737
            3.7e-13, 4.617e-13, 5.949e-13, 7.473e-13, 9.378e-13, 1.191e-12,
            1.481e-12, 1.813e-12, 2.232e-12, 2.722e-12, 3.254e-12, 3.845e-12, 4.458e-12, 5.048e-12, 5.511e-12, 5.898e-12, 6.204e-12, 6.293e-12, 6.386e-12, 6.467e-12, 6.507e-12, 6.466e-12, 6.443e-12, 6.598e-12,
02738
02739
02740
            6.873e-12, 7.3e-12, 7.816e-12, 8.368e-12, 8.643e-12, 8.466e-12, 7.871e-12, 6.853e-12, 5.714e-12, 4.482e-12, 3.392e-12, 2.613e-12,
02741
02742
02743
            2.008e-12, 1.562e-12, 1.228e-12, 9.888e-13, 7.646e-13, 5.769e-13,
02744
            4.368e-13, 3.324e-13, 2.508e-13, 1.916e-13
02745
02746
          static double xfcrev[15] =
            { 1.003, 1.009, 1.015, 1.023, 1.029, 1.033, 1.037,
02748
02749
            1.039, 1.04, 1.046, 1.036, 1.027, 1.01, 1.002, 1.
02750
02751
02752
         double a1, a2, a3, dw, ew, dx, xw, xx, vf2, vf6, cw260, cw296,
02753
            sfac, fscal, cwfrn, ctmpth, ctwfrn, ctwslf;
02754
02755
         int iw, ix;
02756
02757
         /* Get H2O continuum absorption... */
02758
         xw = nu / 10 + 1;

if (xw >= 1 \&\& xw < 2001) {
02759
02760
            iw = (int) xw;
            dw = xw - iw;

ew = 1 - dw;
02761
02762
            cw296 = ew * h2o296[iw - 1] + dw * h2o296[iw];
cw260 = ew * h2o260[iw - 1] + dw * h2o260[iw];
cwfrn = ew * h2ofrn[iw - 1] + dw * h2ofrn[iw];
02763
02764
02765
02766
            if (nu <= 820 || nu >= 960) {
02767
              sfac = 1;
02768
            } else {
             xx = (nu - 820) / 10;
02769
              ix = (int) xx;
02770
               dx = xx - ix;
02771
02772
              sfac = (1 - dx) * xfcrev[ix] + dx * xfcrev[ix + 1];
02773
02774
            ctwslf = sfac * cw296 * pow(cw260 / cw296, (296 - t) / (296 - 260));
            vf2 = POW2(nu - 370);
vf6 = POW3(vf2);
02775
02776
            fscal = 36100 / (vf2 + vf6 * 1e-8 + 36100) * -.25 + 1;
ctwfrn = cwfrn * fscal;
02777
02778
02779
            a1 = nu * u * tanh(.7193876 / t * nu);
02780
            a2 = 296 / t;
02781
            a3 = p / P0 * (q * ctwslf + (1 - q) * ctwfrn) * 1e-20;
02782
            ctmpth = a1 * a2 * a3;
02783
          } else
02784
            ctmpth = 0;
02785
          return ctmpth;
02786 }
```

5.3.2.8 double ctmn2 (double nu, double p, double t)

Compute nitrogen continuum (absorption coefficient).

Definition at line 2790 of file jurassic.c.

```
02793
                      {
          static double ba[98] = { 0., 4.45e-8, 5.22e-8, 6.46e-8, 7.75e-8, 9.03e-8,
02795
02796
          1.06e-7, 1.21e-7, 1.37e-7, 1.57e-7, 1.75e-7, 2.01e-7, 2.3e-7, 2.59e-7, 2.95e-7, 3.26e-7, 3.66e-7, 4.05e-7, 4.47e-7, 4.92e-7,
02797
02798
            5.34e-7, 5.84e-7, 6.24e-7, 6.67e-7, 7.14e-7, 7.26e-7, 7.54e-7,
02799
            7.84e-7, 8.09e-7, 8.42e-7, 8.62e-7, 8.87e-7, 9.11e-7, 9.36e-7,
02800
            9.76e-7, 1.03e-6, 1.11e-6, 1.23e-6, 1.39e-6, 1.61e-6, 1.76e-6,
            1.94e-6, 1.97e-6, 1.87e-6, 1.75e-6, 1.56e-6, 1.42e-6, 1.35e-6,
02801
02802
            1.32e-6, 1.29e-6, 1.29e-6, 1.3e-6, 1.32e-6, 1.33e-6,
            1.34e-6, 1.35e-6, 1.33e-6, 1.31e-6, 1.29e-6, 1.24e-6, 1.2e-6, 1.16e-6, 1.1e-6, 1.04e-6, 9.96e-7, 9.38e-7, 8.63e-7, 7.98e-7, 7.26e-7, 6.55e-7, 5.94e-7, 5.35e-7, 4.74e-7, 4.24e-7, 3.77e-7,
02803
02804
02805
02806
            3.33e-7, 2.96e-7, 2.63e-7, 2.34e-7, 2.08e-7, 1.85e-7, 1.67e-7,
            1.47e-7, 1.32e-7, 1.2e-7, 1.09e-7, 9.85e-8, 9.08e-8, 8.18e-8,
```

```
7.56e-8, 6.85e-8, 6.14e-8, 5.83e-8, 5.77e-8, 5e-8, 4.32e-8, 0.
02809
02810
                        static double betaa[98] = { 802., 802., 761., 722., 679., 646., 609., 562.,
511., 472., 436., 406., 377., 355., 338., 319., 299., 278., 255.,
233., 208., 184., 149., 107., 66., 25., -13., -49., -82., -104.,
-119., -130., -139., -144., -146., -146., -147., -148., -150.,
-153., -160., -169., -181., -189., -195., -200., -205., -209.,
02811
02812
02813
02815
                              133., 130., 160., 161., 163., 153., 260., 263., 263., 161., -161., -161., -161., -161., -161., -161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 161., 
02816
02817
02818
02819
02820
02821
02822
                        static double nua[98] = { 2120., 2125., 2130., 2135., 2140., 2145., 2150.,
2155., 2160., 2165., 2170., 2175., 2180., 2185., 2190., 2195.,
2200., 2205., 2210., 2215., 2220., 2225., 2230., 2235., 2240.,
2245., 2250., 2255., 2260., 2265., 2270., 2275., 2280., 2285.,
02823
02824
02825
02826
02827
                               2290., 2295., 2300., 2305., 2310., 2315., 2320., 2325., 2330.,
02828
                               2335., 2340., 2345., 2350., 2355., 2360., 2365., 2370., 2375.,
02829
                               2380., 2385., 2390., 2395., 2400., 2405., 2410., 2415., 2420.,
02830
                               2425., 2430., 2435., 2440., 2445., 2450., 2455., 2460., 2465.,
                              2470., 2475., 2480., 2485., 2490., 2495., 2500., 2505., 2510., 2515., 2520., 2525., 2530., 2535., 2540., 2545., 2550., 2555., 2560., 2565., 2570., 2575., 2580., 2585., 2590., 2595., 2600., 2605.
02831
02832
02833
02834
02835
                        double b, beta, q_n2 = 0.79, t0 = 273, tr = 296;
02836
02837
02838
                        int idx:
02839
02840
                         /* Check wavenumber range...
02841
                         if (nu < nua[0] || nu > nua[97])
                              return 0;
02842
02843
02844
                        /* Interpolate B and beta... */
02845
                       idx = locate_reg(nua, 98, nu);
02846
                        b = LIN(nua[idx], ba[idx], nua[idx + 1], ba[idx + 1], nu);
02847
                        beta = LIN(nua[idx], betaa[idx], nua[idx + 1], betaa[idx + 1], nu);
02848
                        /* Compute absorption coefficient... */
return 0.1 * POW2(p / P0 * t0 / t) * exp(beta * (1 / tr - 1 / t))
  * q_n2 * b * (q_n2 + (1 - q_n2) * (1.294 - 0.4545 * t / tr));
02849
02850
02851
02852 }
```



5.3.2.9 double ctmo2 (double nu, double p, double t)

Compute oxygen continuum (absorption coefficient).

Definition at line 2856 of file jurassic.c.

```
02859 {
02860
02861 static double ba[90] = { 0., .061, .074, .084, .096, .12, .162, .208, .246, 
02862 .285, .314, .38, .444, .5, .571, .673, .768, .853, .966, 1.097, 
02863 1.214, 1.333, 1.466, 1.591, 1.693, 1.796, 1.922, 2.037, 2.154, 
02864 2.264, 2.375, 2.508, 2.671, 2.847, 3.066, 3.417, 3.828, 4.204, 
02865 4.453, 4.599, 4.528, 4.284, 3.955, 3.678, 3.477, 3.346, 3.29, 
02866 3.251, 3.231, 3.226, 3.212, 3.192, 3.108, 3.033, 2.911, 2.798,
```

```
2.646, 2.508, 2.322, 2.13, 1.928, 1.757, 1.588, 1.417, 1.253,
                1.109, .99, .888, .791, .678, .587, .524, .464, .403, .357, .32
.29, .267, .242, .215, .182, .16, .146, .128, .103, .087, .081,
02868
02869
02870
                 .071, .064, 0.
02871
02872
             static double betaa[90] = { 467., 467., 400., 315., 379., 368., 475., 521., 531., 512., 442., 444., 430., 381., 335., 324., 296., 248., 215.,
02874
                193., 158., 127., 101., 71., 31., -6., -26., -47., -63., -79., -88., -88., -87., -90., -98., -99., -109., -134., -160., -167., -164., -158., -153., -151., -156., -166., -168., -173., -170., -161., -145., -126., -108., -84., -59., -29., 4., 41., 73., 97., 123., 159., 198., 220., 242., 256., 281., 311., 334., 319., 313., 321., 323., 310., 315., 320., 335., 361., 378., 373., 338., 319., 346., 322., 291., 290., 350., 371., 504., 504.
02875
02876
02877
02878
02879
02880
02881
02882
02883
             static double nua[90] = { 1360., 1365., 1370., 1375., 1380., 1385., 1390.,
1395., 1400., 1405., 1410., 1415., 1420., 1425., 1430., 1435.,
1440., 1445., 1450., 1455., 1460., 1465., 1470., 1475., 1480.,
02884
02885
02886
02887
                 1485., 1490., 1495., 1500., 1505., 1510., 1515., 1520., 1525.,
02888
                 1530., 1535., 1540., 1545., 1550., 1555., 1560., 1565., 1570.,
                 1575., 1580., 1585., 1590., 1595., 1600., 1605., 1610., 1615.,
02889
                 1620., 1625., 1630., 1635., 1640., 1645., 1650., 1655., 1660., 1665., 1670., 1675., 1680., 1685., 1690., 1695., 1700., 1705., 1710., 1715., 1720., 1725., 1730., 1735., 1740., 1745., 1750., 1755., 1760., 1765., 1770., 1775., 1780., 1785., 1790., 1795.,
02890
02891
02893
02894
                1800., 1805.
02895
02896
02897
             double b, beta, q_02 = 0.21, t0 = 273, tr = 296;
02898
02899
             int idx;
02900
02901
              /* Check wavenumber range...
             if (nu < nua[0] || nu > nua[89])
02902
02903
               return 0;
02905
              /* Interpolate B and beta... */
02906
            idx = locate_reg(nua, 90, nu);
02907
             b = LIN(nua[idx], ba[idx], nua[idx + 1], ba[idx + 1], nu);
02908
             beta = LIN(nua[idx], betaa[idx], nua[idx + 1], betaa[idx + 1], nu);
02909
             /* Compute absorption coefficient... */ return 0.1 * POW2(p / P0 * t0 / t) * exp(beta * (1 / tr - 1 / t)) * q_o2 *
02910
02911
02912
02913 }
```

Here is the call graph for this function:



5.3.2.10 void copy_atm ($ctl_t * ctl$, $atm_t * atm_dest$, $atm_t * atm_src$, int init)

Copy and initialize atmospheric data.

Definition at line 2917 of file jurassic.c.

```
02921 {
02922
02923 int ig, ip, iw;
02924
02925 size_t s;
02926
```

```
/* Data size... */
02928
       s = (size_t) atm_src->np * sizeof(double);
02929
       /* Copy data... */
02930
02931
       atm_dest->np = atm_src->np;
02932
       memcpv(atm dest->time, atm src->time, s);
       memcpy(atm_dest->z, atm_src->z, s);
02934
       memcpy(atm_dest->lon, atm_src->lon, s);
02935
       memcpy(atm_dest->lat, atm_src->lat, s);
02936
       memcpy(atm_dest->p, atm_src->p, s);
       memcpy(atm_dest->t, atm_src->t, s);
02937
       for (ig = 0; ig < ctl->ng; ig++)
02938
02939
         memcpy(atm_dest->q[ig], atm_src->q[ig], s);
02940
       for (iw = 0; iw < ctl->nw; iw++)
02941
         memcpy(atm\_dest->k[iw], atm\_src->k[iw], s);
02942
02943
       /* Initialize... */
       if (init)
02944
        for (ip = 0; ip < atm_dest->np; ip++) {
02945
           atm_dest->p[ip] = 0;
02947
           atm_dest->t[ip] = 0;
          for (ig = 0; ig < ctl->ng; ig++)
02948
02949
             atm_dest->q[ig][ip] = 0;
02950
           for (iw = 0; iw < ctl->nw; iw++)
02951
             atm_dest->k[iw][ip] = 0;
02952
02953 }
```

5.3.2.11 void copy_obs ($ctl_t * ctl$, obs_t * obs_dest, obs_t * obs_src, int init)

Copy and initialize observation data.

Definition at line 2957 of file jurassic.c.

```
02962
02963
        int id, ir;
02964
02965
        size t s:
02966
02967
        /* Data size... */
02968
        s = (size_t) obs_src->nr * sizeof(double);
02969
02970
        /* Copy data... */
obs_dest->nr = obs_src->nr;
02971
02972
        memcpy(obs_dest->time, obs_src->time, s);
02973
        memcpy(obs_dest->obsz, obs_src->obsz, s);
02974
        memcpy(obs_dest->obslon, obs_src->obslon, s);
02975
        memcpy(obs_dest->obslat, obs_src->obslat, s);
02976
        memcpy(obs_dest->vpz, obs_src->vpz, s);
        memcpy(obs_dest->vplon, obs_src->vplon, s);
memcpy(obs_dest->vplat, obs_src->vplat, s);
02977
02978
        memcpy(obs_dest->tpz, obs_src->tpz, s);
02980
        memcpy(obs_dest->tplon, obs_src->tplon, s);
02981
         memcpy(obs_dest->tplat, obs_src->tplat, s);
02982
        for (id = 0; id < ctl->nd; id++)
02983
        memcpy(obs_dest->rad[id], obs_src->rad[id], s);
for (id = 0; id < ctl->nd; id++)
02984
02985
          memcpy(obs_dest->tau[id], obs_src->tau[id], s);
02986
02987
        /* Initialize... */
        if (init)
02988
        for (id = 0; id < ctl->nd; id++)
02989
           for (ir = 0; ir < obs_dest->nr; ir++)
02990
              if (gsl_finite(obs_dest->rad[id][ir])) {
02991
02992
                obs_dest->rad[id][ir] = 0;
02993
                 obs_dest->tau[id][ir] = 0;
02994
02995 1
```

5.3.2.12 int find_emitter ($ctl_t * ctl$, const char * emitter)

Find index of an emitter.

Definition at line 2999 of file jurassic.c.

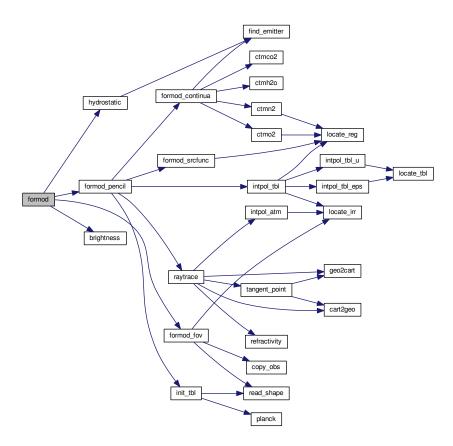
```
03001
03002
03003 int ig;
03004
03005 for (ig = 0; ig < ctl->ng; ig++)
03006 if (strcasecmp(ctl->emitter[ig], emitter) == 0)
03007 return ig;
03008
03009 return -1;
03010 }
```

```
5.3.2.13 void formod ( ctl_t * ctl, atm_t * atm, obs_t * obs )
```

Determine ray paths and compute radiative transfer.

Definition at line 3014 of file jurassic.c.

```
03017
03018
03019
         int id, ir, *mask;
03020
03021
         /* Allocate... */
         ALLOC(mask, int,
ND * NR);
03022
03023
03024
03025
         /* Save observation mask... */
         for (id = 0; id < ctl->nd; id+)
  for (ir = 0; ir < obs->nr; ir++)
    mask[id * NR + ir] = !gsl_finite(obs->rad[id][ir]);
03026
03027
03028
03029
03030
          /* Hydrostatic equilibrium... */
03031
         hydrostatic(ctl, atm);
03032
         /* Calculate pencil beams... */
for (ir = 0; ir < obs->nr; ir++)
03033
03034
03035
            formod_pencil(ctl, atm, obs, ir);
03036
03037
         /* Apply field-of-view convolution... */
03038
         formod_fov(ctl, obs);
03039
03040
         /* Convert radiance to brightness temperature... */
03041
         if (ctl->write_bbt)
03042
          for (id = 0; id < ctl->nd; id++)
03043
              for (ir = 0; ir < obs->nr; ir++)
                 obs->rad[id][ir] = brightness(obs->rad[id][ir], ctl->nu[id]);
03044
03045
         /* Apply observation mask... */
for (id = 0; id < ctl->nd; id++)
   for (ir = 0; ir < obs->nr; ir++)
03046
03047
03048
              if (mask[id * NR + ir])
  obs->rad[id][ir] = GSL_NAN;
03049
03050
03051
03052
         /* Free... */
03053
         free(mask);
03054 }
```



5.3.2.14 void formod_continua ($ctl_t * ctl$, $los_t * los$, int ip, double * beta)

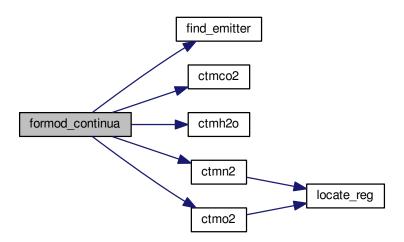
Compute absorption coefficient of continua.

Definition at line 3058 of file jurassic.c.

```
03062
                         {
03063
03064
        static int ig_co2 = -999, ig_h2o = -999;
03065
03066
        int id:
03067
        /* Extinction... */
for (id = 0; id < ctl->nd; id++)
  beta[id] = los->k[ctl->window[id]][ip];
03068
03069
03070
03071
03072
        /* CO2 continuum... */
03073
        if (ctl->ctm_co2) {
         if (ig_co2 == -999)
   ig_co2 = find_emitter(ct1, "CO2");
03074
03075
           if (ig_co2 >= 0)
  for (id = 0; id < ctl->nd; id++)
03076
03077
               03078
03079
03080
03081
03082
        /* H2O continuum... */
        if (ctl->ctm_h2o) {
   if (ig_h2o == -999)
     ig_h2o = find_emitter(ctl, "H2O");
03083
03084
03085
           if (ig_h2o >= 0)
03086
03087
             for (id = 0; id < ctl->nd; id++)
```

```
beta[id] += ctmh2o(ctl->nu[id], los->p[ip], los->t[ip],
                                       los->q[ig_h2o][ip],
los->u[ig_h2o][ip]) / los->ds[ip];
03089
03090
03091
         }
03092
         /* N2 continuum... */
03093
         if (ctl->ctm_n2)
03095
          for (id = 0; id < ctl->nd; id++)
03096
             beta[id] += ctmn2(ctl->nu[id], los->p[ip], los->t[ip]);
03097
03098
        /* 02 continuum... */
        if (ctl->ctm_o2)
  for (id = 0; id < ctl->nd; id++)
    beta[id] += ctmo2(ctl->nu[id], los->p[ip], los->t[ip]);
03099
03100
03101
03102 }
```

Here is the call graph for this function:



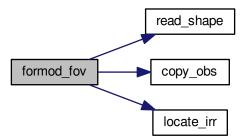
```
5.3.2.15 void formod_fov ( ctl_t * ctl, obs_t * obs )
```

Apply field of view convolution.

Definition at line 3106 of file jurassic.c.

```
03108
03109
03110
        static double dz[NSHAPE], w[NSHAPE];
03111
03112
        static int init = 0, n;
03113
03114
        obs_t *obs2;
03115
03116
        double rad[ND][NR], tau[ND][NR], wsum, z[NR], zfov;
03117
        int i, id, idx, ir, ir2, nz;
03118
03119
        /* Do not take into account FOV... */
if (ctl->fov[0] == '-')
03120
03121
03122
03123
        /* Initialize FOV data... */
if (!init) {
03124
03125
        init = 1;
03126
03127
          read_shape(ctl->fov, dz, w, &n);
03128
```

```
03130
        /* Allocate... */
03131
        ALLOC(obs2, obs_t, 1);
03132
03133
        /* Copy observation data... */
03134
       copy_obs(ctl, obs2, obs, 0);
03135
03136
        /* Loop over ray paths... */
03137
        for (ir = 0; ir < obs->nr; ir++) {
03138
03139
          /* Get radiance and transmittance profiles... */
03140
          nz = 0;
          for (ir2 = GSL_MAX(ir - NFOV, 0); ir2 < GSL_MIN(ir + 1 + NFOV, obs->nr);
03141
03142
03143
            if (obs->time[ir2] == obs->time[ir]) {
03144
             z[nz] = obs2->vpz[ir2];
              for (id = 0; id < ctl->nd; id++) {
03145
                rad[id][nz] = obs2->rad[id][ir2];
tau[id][nz] = obs2->tau[id][ir2];
03146
03147
03148
03149
              nz++;
          }
if (nz < 2)
03150
03151
            ERRMSG("Cannot apply FOV convolution!");
03152
03153
03154
          /\star Convolute profiles with FOV... \star/
03155
03156
          for (id = 0; id < ctl->nd; id++) {
03157
           obs->rad[id][ir] = 0;
03158
            obs->tau[id][ir] = 0;
03159
03160
          for (i = 0; i < n; i++) {
03161
           zfov = obs->vpz[ir] + dz[i];
03162
             idx = locate_irr(z, nz, zfov);
            for (id = 0; id < ctl->nd; id++) {
03163
              obs->rad[id][ir] += w[i]
03164
              * LIN(z[idx], rad[id][idx], z[idx + 1], rad[id][idx + 1], zfov);
obs->tau[id][ir] += w[i]
03165
03166
03167
                * LIN(z[idx], tau[id][idx], z[idx + 1], tau[id][idx + 1], zfov);
03168
03169
            wsum += w[i];
0.3170
          for (id = 0; id < ctl->nd; id++) {
0.3171
            obs->rad[id][ir] /= wsum;
03172
03173
            obs->tau[id][ir] /= wsum;
03174
03175
03176
03177
        /* Free... */
03178
        free (obs2):
03179 }
```

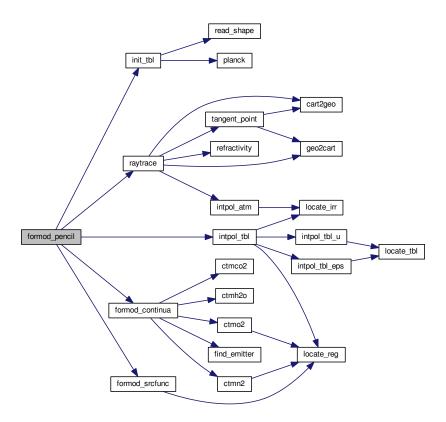


5.3.2.16 void formod_pencil ($ctl_t * ctl$, $atm_t * atm$, $obs_t * obs$, int ir)

Compute radiative transfer for a pencil beam.

Definition at line 3183 of file jurassic.c.

```
03187
                {
03188
03189
        static tbl t *tbl;
03190
03191
        static int init = 0;
03192
03193
        los_t *los;
03194
03195
        double beta_ctm[ND], eps, src_planck[ND], tau_path[NG][ND], tau_gas[ND];
03196
03197
        int id, ip;
03198
03199
        /\star Initialize look-up tables... \star/
03200
        if (!init) {
03201
         init = 1;
03202
          ALLOC(tbl, tbl_t, 1);
03203
          init tbl(ctl, tbl);
03204
03205
03206
        /* Allocate... */
03207
        ALLOC(los, los_t, 1);
03208
03209
        /* Initialize... */
        for (id = 0; id < ctl->nd; id++) {
03210
        obs->rad[id][ir] = 0;
03211
03212
         obs->tau[id][ir] = 1;
03213
03214
        /* Raytracing... */
03215
03216
        raytrace(ctl, atm, obs, los, ir);
03217
03218
        /* Loop over LOS points... */
03219
        for (ip = 0; ip < los->np; ip++) {
03220
03221
          /* Get trace gas transmittance... */
          intpol_tbl(ctl, tbl, los, ip, tau_path, tau_gas);
03222
03223
03224
          /* Get continuum absorption... */
03225
          formod_continua(ctl, los, ip, beta_ctm);
03226
03227
          /* Compute Planck function... */
03228
          formod_srcfunc(ctl, tbl, los->t[ip], src_planck);
03229
03230
          /* Loop over channels... */
03231
          for (id = 0; id < ctl->nd; id++)
03232
            if (tau_gas[id] > 0) {
03233
03234
              /* Get segment emissivity... */
03235
              eps = 1 - tau_gas[id] * exp(-beta_ctm[id] * los->ds[ip]);
03236
03237
              /* Compute radiance... */
03238
              obs->rad[id][ir] += src_planck[id] * eps * obs->tau[id][ir];
03239
03240
              /\star Compute path transmittance... \star/
03241
              obs->tau[id][ir] *= (1 - eps);
03242
03243
03244
03245
        /* Add surface... */
        if (los->tsurf > 0) {
  formod_srcfunc(ctl, tbl, los->tsurf, src_planck);
03246
03247
03248
         for (id = 0; id < ctl->nd; id++)
03249
            obs->rad[id][ir] += src_planck[id] * obs->tau[id][ir];
03250
03251
03252
        /* Free... */
03253
        free(los);
03254 }
```



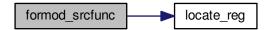
5.3.2.17 void formod_srcfunc (ctl_t * ctl, tbl_t * tbl, double *, double * src)

Compute Planck source function.

Definition at line 3258 of file jurassic.c.

```
03262
03263
03264
       int id, it;
03265
      /* Determine index in temperature array... */
it = locate_reg(tbl->st, TBLNS, t);
03266
03267
03268
03269
       /* Interpolate Planck function value... */
       03270
03271
03272
03273 }
```

Here is the call graph for this function:



5.3.2.18 void geo2cart (double z, double lon, double lat, double *x)

Convert geolocation to Cartesian coordinates.

Definition at line 3277 of file jurassic.c.

```
03281 {
03282
03283 double radius;
03284
03285 radius = z + RE;
03286 x[0] = radius * cos(lat / 180 * M_PI) * cos(lon / 180 * M_PI);
03287 x[1] = radius * cos(lat / 180 * M_PI) * sin(lon / 180 * M_PI);
03288 x[2] = radius * sin(lat / 180 * M_PI);
```

5.3.2.19 void hydrostatic (ctl t * ctl, atm t * atm)

Set hydrostatic equilibrium.

Definition at line 3293 of file jurassic.c.

```
03295
03296
03297
        static int ig_h2o = -999;
03298
03299
        double dzmin = 1e99, e = 0, mean, mmair = 28.96456e-3, mmh2o = 18.0153e-3;
03300
        int i, ip, ipref = 0, ipts = 20;
03302
03303
        /\star Check reference height... \star/
03304
        if (ctl->hydz < 0)
03305
          return:
03306
03307
        /* Determine emitter index of H2O... */
        if (ig_h2o == -999)
03308
03309
          ig_h2o = find_emitter(ctl, "H2O");
03310
03311
        /* Find air parcel next to reference height... */
        for (ip = 0; ip < atm->np; ip++)
  if (fabs(atm->z[ip] - ctl->hydz) < dzmin) {</pre>
03312
03313
            dzmin = fabs(atm->z[ip] - ctl->hydz);
ipref = ip;
03314
03315
03316
03317
03318
        /* Upper part of profile... */
        for (ip = ipref + 1; ip < atm->np; ip++) {
03319
03320
          mean = 0;
03321
          for (i = 0; i < ipts; i++) {
03322
            if (ig_h2o >= 0)
            03323
03324
03325
03326
              * G0 / RI
03327
               / LIN(0.0, atm->t[ip - 1], ipts - 1.0, atm->t[ip], (double) i) / ipts;
03328
03329
03330
          /* Compute p(z,T) \dots */
03331
          atm->p[ip] =
03332
            \exp(\log(\text{atm->p[ip - 1]}) - \text{mean} * 1000 * (\text{atm->z[ip] - atm->z[ip - 1]}));
03333
03334
        /* Lower part of profile... */
for (ip = ipref - 1; ip >= 0; ip--) {
03335
03336
03337
          mean = 0;
          for (i = 0; i < ipts; i++) {</pre>
03338
            if (ig_h2o >= 0)
03339
03340
              e = LIN(0.0, atm->q[ig_h2o][ip + 1],
            ipts - 1.0, atm->q[ig_h2o][ip], (double) i);
mean += (e * mmh2o + (1 - e) * mmair)
  * G0 / RI
03341
03342
03343
03344
               / LIN(0.0, atm->t[ip + 1], ipts - 1.0, atm->t[ip], (double) i) / ipts;
03345
03346
03347
          /* Compute p(z,T) \dots */
03348
          atm->p[ip]
03349
             \exp(\log(atm - p[ip + 1]) - mean * 1000 * (atm - z[ip] - atm - z[ip + 1]));
03350
03351 }
```



```
5.3.2.20 void idx2name ( ctl_t * ctl, int idx, char * quantity )
```

Determine name of state vector quantity for given index.

Definition at line 3355 of file jurassic.c.

```
03358
                         {
03359
03360
        int ig, iw;
03362
        if (idx == IDXP)
03363
         sprintf(quantity, "PRESSURE");
03364
03365
        if (idx == IDXT)
03366
         sprintf(quantity, "TEMPERATURE");
03367
03368
        for (ig = 0; ig < ctl->ng; ig++)
03369
        if (idx == IDXQ(ig))
            sprintf(quantity, "%s", ctl->emitter[ig]);
03370
03371
        for (iw = 0; iw < ctl->nw; iw++)
  if (idx == IDXK(iw))
03372
03373
03374
            sprintf(quantity, "EXTINCT_WINDOW%d", iw);
03375 }
```

```
5.3.2.21 void init_tbl ( ctl_t * ctl, tbl_t * tbl )
```

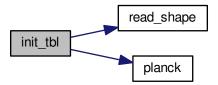
Initialize look-up tables.

Definition at line 3379 of file jurassic.c.

```
03381
03382
03383
        FILE *in;
03384
03385
        char filename[2 * LEN], line[LEN];
03386
03387
        double eps, eps_old, press, press_old, temp, temp_old, u, u_old,
03388
          f[NSHAPE], fsum, nu[NSHAPE];
03389
03390
        int i, id, ig, ip, it, n;
03391
03392
        /* Loop over trace gases and channels... */
03393 for (ig = 0; ig < ctl->ng; ig++)
03394 #pragma omp parallel for default (none) shared(ctl,tbl,ig) private(in,filename,line,eps,eps_old,press,
03396
03397
             /* Initialize... */
03398
            tbl->np[ig][id] = -1;
03399
            eps_old = -999;
            press_old = -999;
temp_old = -999;
u_old = -999;
03400
03401
03402
03403
03404
            /* Try to open file... */
```

```
sprintf(filename, "%s_%.4f_%s.tab",
                      ctl->tblbase, ctl->nu[id], ctl->emitter[ig]);
if (!(in = fopen(filename, "r"))) {
03406
03407
                          printf("Missing emissivity table: sn", filename);
03408
03409
                          continue;
03410
03411
                      printf("Read emissivity table: %s\n", filename);
03412
03413
                       /* Read data... */
03414
                       while (fgets(line, LEN, in)) {
03415
03416
                          /* Parse line... */
if (sscanf(line, "%lg %lg %lg %lg", &press, &temp, &u, &eps) != 4)
03417
03418
03419
03420
                           /* Determine pressure index... */
                          if (press != press_old) {
  press_old = press;
  if ((++tbl->np[ig][id]) >= TBLNP)
03421
03422
03423
                              ERRMSG("Too many pressure levels!");
tbl->nt[ig][id][tbl->np[ig][id]] = -1;
03424
03425
03426
03427
03428
                          /* Determine temperature index... */
03429
                          if (temp != temp_old) {
                             temp_old = temp;
03430
03431
                              if ((++tbl->nt[ig][id][tbl->np[ig][id]]) >= TBLNT)
03432
                                 ERRMSG("Too many temperatures!");
03433
                              tbl->nu[ig][id][tbl->np[ig][id]]
                                  [tbl->nt[ig][id][tbl->np[ig][id]]] = -1;
03434
03435
03436
03437
                          /* Determine column density index... */
03438
                          03439
                                  [tbl->nt[ig][id][tbl->np[ig][id]]] < 0) {
03440
                              eps_old = eps;
                              u_old = u;
03441
                              if ((++tbl->nu[ig][id][tbl->np[ig][id]]
03442
03443
                                        [tbl->nt[ig][id][tbl->np[ig][id]]]) >= TBLNU) {
03444
                                  tbl->nu[ig][id][tbl->np[ig][id]]
03445
                                     [tbl->nt[ig][id][tbl->np[ig][id]]]--;
03446
                                 continue:
03447
                             }
03448
                          }
03449
03450
                           /* Store data...
03451
                          tbl->p[ig][id][tbl->np[ig][id]] = press;
                          \label{tbl-hp[ig][id][tbl-hp[ig][id]][tbl-ht[ig][id][tbl-hp[ig][id]]]} tbl-ht[ig][id][tbl-hp[ig][id]]]
03452
03453
                              = temp;
03454
                          tbl->u[ig][id][tbl->np[ig][id]][tbl->nt[ig][id][tbl->np[ig][id]]]
                              [tbl->nu[ig][id][tbl->np[ig][id]]
03455
03456
                                [tbl->nt[ig][id][tbl->np[ig][id]]] = (float) u;
03457
                          \label{locality} $$ tbl->eps[ig][id][tbl->np[ig][id]][tbl->nt[ig][id][tbl->np[ig][id]]] $$ $$ $$ the sum of 
03458
                              [tbl->nu[ig][id][tbl->np[ig][id]]
                                [tbl->nt[ig][id][tbl->np[ig][id]]] = (float) eps;
03459
03460
03461
03462
                       /* Increment counters... */
03463
                       tbl->np[ig][id]++;
03464
                       for (ip = 0; ip < tbl->np[ig][id]; ip++) {
                          tbl->nt[ig][id][ip]++;
for (it = 0; it < tbl->nt[ig][id][ip]; it++)
03465
03466
03467
                              tbl->nu[ig][id][ip][it]++;
03468
03469
03470
                       /* Close file... */
03471
                      fclose(in);
03472
03473
03474
               /* Write info... */
03475
              printf("Initialize source function table...\n");
03476
03477  /* Loop over channels... */
03478  #pragma omp parallel for default(none) shared(ctl,tbl,ig) private(filename,it,i,n,f,fsum,nu)
03479  for (id = 0; id < ctl->nd; id++) {
03480
03481
                   /* Read filter function... */
                  sprintf(filename, "%s_%.4f.filt", ctl->tblbase, ctl->nu[id]);
03482
03483
                  read_shape(filename, nu, f, &n);
03484
03485
                   /* Compute source function table... */
                  for (it = 0; it < TBLNS; it++) {</pre>
03486
03487
03488
                       /* Set temperature... */
03489
                      tbl \rightarrow st[it] = LIN(0.0, TMIN, TBLNS - 1.0, TMAX, (double) it);
03490
03491
                       /* Integrate Planck function... */
```

Here is the call graph for this function:



```
5.3.2.22 void intpol_atm ( ctl_t * ctl, atm_t * atm, double z, double * p, double * t, double * q, double * k)
```

Interpolate atmospheric data.

Definition at line 3505 of file jurassic.c.

```
03512
03513
03514
        int ig, ip, iw;
03515
03516
        /* Get array index... */
03517
        ip = locate_irr(atm->z, atm->np, z);
03518
03519
       03520
03521
03522
03523
03524
            \label{eq:linear} \begin{split} &\text{LIN}(\text{atm->z[ip], atm->q[ig][ip], atm->z[ip+1], atm->q[ig][ip+1], z);} \end{split}
        for (iw = 0; iw < ctl->nw; iw++)
k[iw] =
03525
03526
03527
            \label{lin} LIN\,(atm->z\,[ip]\,,\ atm->z\,[ip+1]\,,\ atm->z\,[ip+1]\,,\ atm->k\,[iw]\,[ip+1]\,,\ z)\,;
03528 }
```

Here is the call graph for this function:



5.3.2.23 void intpol_tbl (ctl_t * ctl, tbl_t * tbl, los_t * los, int ip, double tau_path[NG][ND], double tau_seg[ND])

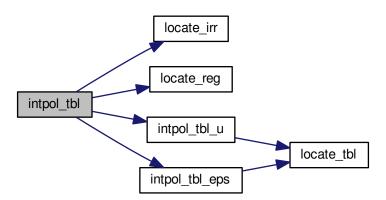
Get transmittance from look-up tables.

Definition at line 3532 of file jurassic.c.

```
03538
03539
03540
        double eps, eps00, eps01, eps10, eps11, u;
03541
03542
        int id, ig, ipr, it0, it1;
03543
03544
        /* Initialize... */
03545
        if (ip <= 0)</pre>
03546
         for (ig = 0; ig < ctl->ng; ig++)
03547
            for (id = 0; id < ctl->nd; id++)
03548
              tau_path[ig][id] = 1;
03549
03550
        /* Loop over channels... */
03551
       for (id = 0; id < ctl->nd; id++) {
03552
          /* Initialize... */
03553
03554
          tau_seg[id] = 1;
03555
03556
          /* Loop over emitters.... */
03557
          for (ig = 0; ig < ctl->ng; ig++) {
03558
03559
            /\star Check size of table (pressure)... \star/
03560
            if (tbl->np[ig][id] < 2)</pre>
03561
              eps = 0;
03562
            /* Check transmittance... */
03563
            else if (tau_path[ig][id] < 1e-9)</pre>
03565
             eps = 1;
03566
03567
            /* Interpolate... */
03568
            else (
03569
03570
              /\star Determine pressure and temperature indices... \star/
03571
              ipr = locate_irr(tbl->p[ig][id], tbl->np[ig][id], los->p[ip]);
03572
              it0 =
03573
                locate_irr(tbl->t[ig][id][ipr], tbl->nt[ig][id][ipr], los->
     t[ip]);
03574
              it1 =
               locate_reg(tbl->t[ig][id][ipr + 1], tbl->nt[ig][id][ipr + 1],
03576
                           los->t[ip]);
03577
03578
              /\star Check size of table (temperature and column density)... \star/
03579
              || tbl->nu[ig][id][ipr][it0] < 2
03580
03581
                  || tbl->nu[ig][id][ipr][it0 + 1] < 2
                  || tbl->nu[ig][id][ipr + 1][it1] < 2
|| tbl->nu[ig][id][ipr + 1][it1 + 1] < 2)
03582
03583
03584
                eps = 0;
03585
03586
              else {
03587
03588
                /\star Get emissivities of extended path... \star/
                u = intpol_tbl_u(tbl, ig, id, ipr, it0, 1 - tau_path[ig][id]);
eps00 = intpol_tbl_eps(tbl, ig, id, ipr, it0, u + los->u[ig][ip]);
03589
03590
03591
03592
                u = intpol_tbl_u(tbl, iq, id, ipr, it0 + 1, 1 - tau_path[iq][id]);
                eps01 =
03593
03594
                  intpol_tbl_eps(tbl, ig, id, ipr, it0 + 1, u + los->u[ig][ip]);
03595
03596
                u = intpol_tbl_u(tbl, ig, id, ipr + 1, it1, 1 - tau_path[ig][id]);
                eps10 =
03597
03598
                  intpol_tbl_eps(tbl, ig, id, ipr + 1, it1, u + los->u[ig][ip]);
03599
03600
                  intpol_tbl_u(tbl, ig, id, ipr + 1, it1 + 1, 1 - tau_path[ig][id]);
03601
03602
                  intpol_tbl_eps(tbl, ig, id, ipr + 1, it1 + 1, u + los->
03603
     u[iq][ip]);
03604
03605
                /* Interpolate with respect to temperature... */
03606
                eps00 = LIN(tbl->t[ig][id][ipr][it0], eps00,
03607
                            tbl->t[ig][id][ipr][it0 + 1], eps01, los->t[ip]);
                03608
03609
03610
03611
                /* Interpolate with respect to pressure... */
03612
                eps00 = LIN(tbl->p[ig][id][ipr], eps00,
```

```
tbl->p[ig][id][ipr + 1], eps11, los->p[ip]);
03614
03615
                 /* Check emssivity range... */
                 eps00 = GSL_MAX(GSL_MIN(eps00, 1), 0);
03616
03617
03618
                 /* Determine segment emissivity... */
03619
                 eps = 1 - (1 - eps00) / tau_path[ig][id];
03620
03621
03622
            /* Get transmittance of extended path... */ tau_path[ig][id] *= (1 - eps);
03623
03624
03625
03626
             /* Get segment transmittance... */
03627
             tau_seg[id] *= (1 - eps);
03628
        }
03629
03630 }
```

Here is the call graph for this function:



5.3.2.24 double intpol_tbl_eps ($tbl_t * tbl_t$, int ig_t , int

Interpolate emissivity from look-up tables.

Definition at line 3634 of file jurassic.c.

```
03640
                   {
03641
03642
        int idx;
03643
        /* Lower boundary... */
if (u < tbl->u[ig][id][ip][it][0])
03644
03645
03646
        return LIN(0, 0, tbl->u[ig][id][ip][it][0], tbl->eps[ig][id][ip][it][0],
03647
03648
03649
        /* Upper boundary... */
        else if (u > tbl->u[ig][id][ip][it][tbl->nu[ig][id][ip][it] - 1])
03650
         return LIN(tbl->u[ig][id][ip][it][tbl->nu[ig][id][ip][it] - 1],
03651
                     tbl->eps[ig][id][ip][it][tbl->nu[ig][id][ip][it] - 1],
03652
03653
                      1e30, 1, u);
03654
03655
        /* Interpolation... */
03656
        else {
03657
03658
          /* Get index... */
03659
          idx = locate_tbl(tbl->u[ig][id][ip][it], tbl->nu[ig][id][ip][it], u);
03660
```

Here is the call graph for this function:



5.3.2.25 double intpol_tbl_u ($tbl_t * tbl$, int ig, int ig, int ig, int if, double eps)

Interpolate column density from look-up tables.

Definition at line 3671 of file jurassic.c.

```
03677
03678
03679
         int idx;
03680
03681
         /* Lower boundary... */
         if (eps < tbl->eps[ig][id][ip][it][0])
03683
          return LIN(0, 0, tbl->eps[ig][id][ip][it][0], tbl->u[ig][id][ip][it][0],
03684
                        eps);
03685
         /* Upper boundary... */
else if (eps > tbl->eps[ig][id][ip][it][tbl->nu[ig][id][ip][it] - 1])
03686
03687
03688
          return LIN(tbl->eps[ig][id][ip][it][tbl->nu[ig][id][ip][it] - 1],
                        tbl->u[ig][id][ip][it][tbl->nu[ig][id][ip][it] - 1],
03689
03690
                        1, 1e30, eps);
03691
03692
         /* Interpolation... */
03693
         else {
03694
03695
03696
           idx = locate\_tbl(tbl->eps[ig][id][ip][it], \ tbl->nu[ig][id][ip][it], \ eps);
03697
           /* Interpolate... */
03698
03699
             LIN(tbl->eps[ig][id][ip][it][idx], tbl->u[ig][id][ip][it][idx], tbl->eps[ig][id][ip][it][idx + 1], tbl->u[ig][id][ip][it][idx + 1],
03700
03701
03702
03703
         }
03704 }
```

Here is the call graph for this function:



5.3.2.26 void jsec2time (double jsec, int * year, int * mon, int * day, int * hour, int * min, int * sec, double * remain)

Convert seconds to date.

Definition at line 3708 of file jurassic.c.

```
03716
03717
03718
        struct tm t0, *t1;
03720
        time_t jsec0;
03721
03722
        t0.tm\_year = 100;
        t0.tm_mon = 0;
03723
03724
        t0.tm_mday = 1;
03725
        t0.tm\_hour = 0;
        t0.tm_min = 0;
03726
        t0.tm\_sec = 0;
03727
03728
03729
        jsec0 = (time_t) jsec + timegm(&t0);
03730
       t1 = gmtime(&jsec0);
03731
03732
        *year = t1->tm_year + 1900;
03733
        *mon = t1->tm_mon + 1;
        *day = t1->tm_mday;
03734
03735
        *hour = t1->tm_hour;
03736
        *min = t1->tm_min;
        *sec = t1->tm_sec;
*remain = jsec - floor(jsec);
03737
03738
03739 }
```

5.3.2.27 void kernel (ctl_t * ctl, atm_t * atm, obs_t * obs, gsl_matrix * k)

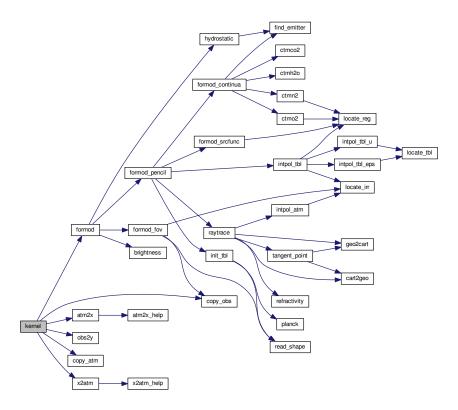
Compute Jacobians.

Definition at line 3743 of file jurassic.c.

```
03747
                                                                             {
03748
03749
                         atm_t *atm1;
03750
                        obs_t *obs1;
03751
03752
                        gsl_vector *x0, *x1, *yy0, *yy1;
03753
03754
                        int *iqa, j;
03755
03756
                        double h;
03757
03758
                        size_t i, n, m;
03759
03760
                        /* Get sizes... */
03761
                        m = k -> size1;
                        n = k - > size2;
03762
03763
03764
                         /* Allocate... */
03765
                        x0 = gsl\_vector\_alloc(n);
03766
                        yy0 = gsl_vector_alloc(m);
03767
                         ALLOC(iqa, int,
03768
                                            N);
03769
03770
                        /* Compute radiance for undisturbed atmospheric data... */
03771
                        formod(ctl, atm, obs);
03772
03773
                        /* Compose vectors... */
03774
                        atm2x(ctl, atm, x0, iqa, NULL);
03775
                        obs2y(ctl, obs, yy0, NULL, NULL);
03776
03777
                        /* Initialize kernel matrix... */
03778
                        gsl_matrix_set_zero(k);
03779
03780
                         /\star Loop over state vector elements... \star/
03781 \text{ \#pragma omp parallel for default(none) shared(ctl,atm,obs,k,x0,yy0,n,m,iqa) private(i, j, h, x1, yy1, atm1, h, x1, yy1, h, x1, 
                      obs1)
03782
                        for (j = 0; j < (int) n; j++) {
03783
03784
                               /* Allocate... */
```

```
03785
           x1 = gsl_vector_alloc(n);
03786
           yy1 = gsl_vector_alloc(m);
03787
           ALLOC(atm1, atm_t, 1);
03788
           ALLOC(obs1, obs_t, 1);
03789
03790
           /* Set perturbation size... */
03791
           if (iqa[j] == IDXP)
03792
             h = GSL_MAX(fabs(0.01 * gsl_vector_get(x0, (size_t) j)), 1e-7);
03793
           else if (iqa[j] == IDXT)
            h = 1;
03794
           else if (iqa[j] >= IDXQ(0) && iqa[j] < IDXQ(ctl->ng))
03795
           h = GSL_MAX(fabs(0.01 * gsl_vector_get(x0, (size_t) j)), le-15);
else if (iqa[j] >= IDXK(0) && iqa[j] < IDXK(ctl->nw))
03796
03797
03798
03799
           else
             ERRMSG("Cannot set perturbation size!");
03800
03801
03802
           /* Disturb state vector element... */
03803
           gsl_vector_memcpy(x1, x0);
03804
           gsl_vector_set(x1, (size_t) j, gsl_vector_get(x1, (size_t) j) + h);
           copy_atm(ctl, atml, atm, 0);
copy_obs(ctl, obs1, obs, 0);
03805
03806
03807
           x2atm(ctl, x1, atm1);
03808
03809
           /* Compute radiance for disturbed atmospheric data... */
03810
           formod(ctl, atml, obsl);
03811
03812
           /\star Compose measurement vector for disturbed radiance data... \star/
03813
           obs2y(ctl, obs1, yy1, NULL, NULL);
03814
03815
           /* Compute derivatives... */
03816
           for (i = 0; i < m; i++)
03817
             gsl_matrix_set(k, i, (size_t) j,
03818
                              (gsl_vector_get(yy1, i) - gsl_vector_get(yy0, i)) / h);
03819
           /* Free... */
gsl_vector_free(x1);
gsl_vector_free(yy1);
03820
03821
03822
03823
           free(atm1);
03824
           free (obs1);
03825
03826
        /* Free... */
gsl_vector_free(x0);
03827
03828
03829
         gsl_vector_free(yy0);
03830
         free(iqa);
03831 }
```

Here is the call graph for this function:



5.3.2.28 int locate_irr (double *xx, int n, double x)

Find array index for irregular grid.

Definition at line 3835 of file jurassic.c.

```
03838
03839
03840
          int i, ilo, ihi;
03841
         ilo = 0;
ihi = n - 1;
i = (ihi + ilo) >> 1;
03842
03843
03844
03845
          if (xx[i] < xx[i + 1])
  while (ihi > ilo + 1) {
   i = (ihi + ilo) >> 1;
03846
03847
03848
               <u>if</u> (xx[i] > x)
03849
03850
                 ihi = i;
               else
03851
03852
                 ilo = i;
03853
         } else
            while (ihi > ilo + 1) {
03854
             i = (ihi + ilo) >> 1;
if (xx[i] <= x)
03856
03857
                 ihi = i;
               else
03858
03859
                 ilo = i;
03860
03861
03862
         return ilo;
03863 }
```

```
5.3.2.29 int locate_reg ( double *xx, int n, double x )
```

Find array index for regular grid.

Definition at line 3867 of file jurassic.c.

```
03870
03871
03872
          int i;
03873
03874
         /* Calculate index... */
i = (int) ((x - xx[0]) / (xx[1] - xx[0]));
03875
03877
          /* Check range... */
          <u>if</u> (i < 0)
03878
         i = 0;
else if (i >= n - 2)
i = n - 2;
03879
03880
03881
03883
          return i;
03884 }
```

5.3.2.30 int locate_tbl (float *xx, int n, double x)

Find array index in float array.

Definition at line 3888 of file jurassic.c.

```
03891
                    {
03892
03893
        int i, ilo, ihi;
03894
        ilo = 0;
ihi = n - 1;
03895
03896
        i = (ihi + ilo) >> 1;
03897
03898
        while (ihi > ilo + 1) {
        i = (ihi + ilo) >> 1;
if (xx[i] > x)
03900
03901
            ihi = i;
03902
          else
03903
03904
             ilo = i;
03905
        }
03906
03907
        return ilo;
03908 }
```

5.3.2.31 size_t obs2y (ctl_t * ctl, obs_t * obs, gsl_vector * y, int * ida, int * ira)

Compose measurement vector.

Definition at line 3912 of file jurassic.c.

```
03917
                    {
03918
03919
         int id, ir;
03920
03921
        size_t m = 0;
03922
03923
        /* Determine measurement vector... */
03924
        for (ir = 0; ir < obs->nr; ir++)
03925
          for (id = 0; id < ctl->nd; id++)
03926
             if (gsl_finite(obs->rad[id][ir])) {
              if (y != NULL)
    gsl_vector_set(y, m, obs->rad[id][ir]);
if (ida != NULL)
    ida[m] = id;
03927
03928
03929
03930
03931
               if (ira != NULL)
03932
                 ira[m] = ir;
03933
               m++;
             }
03934
03935
03936
        return m:
03937 }
```

5.3.2.32 double planck (double t, double nu)

Compute Planck function.

Definition at line 3941 of file jurassic.c.

5.3.2.33 void raytrace ($ctl_t * ctl$, $atm_t * atm$, $obs_t * obs$, $los_t * los$, int ir)

Do ray-tracing to determine LOS.

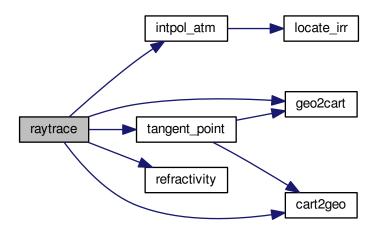
Definition at line 3950 of file jurassic.c.

```
03955
03956
03957
        double cosa, d, dmax, dmin = 0, ds, ex0[3], ex1[3], frac, h = 0.02, k[NW],
03958
          lat, lon, n, naux, ng[3], norm, p, q[NG], t, x[3], xh[3],
03959
          xobs[3], xvp[3], z = 1e99, zmax, zmin, zrefrac = 60;
03960
03961
        int i, ig, ip, iw, stop = 0;
03962
        /* Initialize... */
03964
        los->np = 0;
03965
        los \rightarrow tsurf = -999;
03966
        obs->tpz[ir] = obs->vpz[ir];
        obs->tplon[ir] = obs->vplon[ir];
03967
03968
        obs->tplat[ir] = obs->vplat[ir];
03969
03970
        /* Get altitude range of atmospheric data... */
03971
        gsl_stats_minmax(&zmin, &zmax, atm->z, 1, (size_t) atm->np);
03972
03973
        /* Check observer altitude... */
03974
        if (obs->obsz[ir] < zmin)</pre>
03975
          ERRMSG("Observer below surface!");
03976
03977
        /\star Check view point altitude... \star/
03978
        if (obs->vpz[ir] > zmax)
03979
          return;
03980
03981
        /* Determine Cartesian coordinates for observer and view point... */
03982
        geo2cart(obs->obsz[ir], obs->obslon[ir], obs->obslat[ir], xobs);
03983
        geo2cart(obs->vpz[ir], obs->vplon[ir], obs->vplat[ir], xvp);
03984
03985
        /\star Determine initial tangent vector... \star/
        for (i = 0; i < 3; i++)
  ex0[i] = xvp[i] - xobs[i];</pre>
03986
03987
03988
        norm = NORM(ex0);
03989
        for (i = 0; i < 3; i++)</pre>
03990
          ex0[i] /= norm;
03991
03992
        /* Observer within atmosphere... */
        for (i = 0; i < 3; i++)
03993
          x[i] = xobs[i];
03995
03996
        /\star Observer above atmosphere (search entry point)... \star/
03997
        if (obs->obsz[ir] > zmax) {
03998
          dmax = norm;
03999
          while (fabs(dmin - dmax) > 0.001) {
04000
            d = (dmax + dmin) / 2;
04001
            for (i = 0; i < 3; i++)
04002
              x[i] = xobs[i] + d * ex0[i];
04003
            cart2geo(x, &z, &lon, &lat);
            if (z <= zmax && z > zmax - 0.001)
04004
04005
              break;
            if (z < zmax - 0.0005)
04006
04007
              dmax = d;
04008
            else
04009
              dmin = d;
04010
04011
        }
04012
04013
        /* Ray-tracing... */
```

```
04014
        while (1) {
04015
04016
           /* Set step length... */
04017
           ds = ctl->rayds;
           if (ctl->raydz > 0) {
04018
04019
             norm = NORM(x);
             for (i = 0; i < 3; i++)
04020
04021
               xh[i] = x[i] / norm;
04022
             cosa = fabs(DOTP(ex0, xh));
04023
             if (cosa != 0)
               ds = GSL_MIN(ctl->rayds, ctl->raydz / cosa);
04024
04025
04026
04027
           /* Determine geolocation... */
04028
           cart2geo(x, &z, &lon, &lat);
04029
           /\star Check if LOS hits the ground or has left atmosphere... \star/
04030
04031
           if (z < zmin || z > zmax)
             stop = (z < zmin ? 2 : 1);
04032
04033
             frac =
               ((z <
04034
04035
                  zmin ? zmin : zmax) - los->z[los->np-1]) / (z - los->z[los->np-1])
04036
                                                                                  11);
             04037
04038
04039
             for (i = 0; i < 3; i++)
04040
               x[i] = xh[i] + frac * (x[i] - xh[i]);
             cart2geo(x, &z, &lon, &lat);
los->ds[los->np - 1] = ds * frac;
04041
04042
04043
             ds = 0:
04044
04045
04046
           /* Interpolate atmospheric data... */
04047
           intpol_atm(ctl, atm, z, &p, &t, q, k);
04048
04049
           /* Save data... */
           los->lon[los->np] = lon;
los->lat[los->np] = lat;
04050
04051
04052
           los \rightarrow z[los \rightarrow np] = z;
04053
           los \rightarrow p[los \rightarrow np] = p;
04054
           los \rightarrow t[los \rightarrow np] = t;
           for (ig = 0; ig < ctl->ng; ig++)
04055
           los->q[ig][los->np] = q[ig];
for (iw = 0; iw < ctl->nw; iw++)
los->k[iw][los->np] = k[iw];
04056
04057
04058
04059
           los->ds[los->np] = ds;
04060
04061
           /\star Increment and check number of LOS points... \star/
           if ((++los->np) > NLOS)
04062
             ERRMSG("Too many LOS points!");
04063
04064
04065
           /* Check stop flag... */
04066
           if (stop) {
04067
             los->tsurf = (stop == 2 ? t : -999);
04068
             break;
04069
           }
04070
04071
           /* Determine refractivity... */
04072
           if (ctl->refrac && z <= zrefrac)</pre>
04073
             n = 1 + refractivity(p, t);
04074
           else
04075
            n = 1;
04076
04077
           /* Construct new tangent vector (first term)... */
04078
           for (i = 0; i < 3; i++)
             ex1[i] = ex0[i] * n;
04079
04080
           /* Compute gradient of refractivity... */
04081
04082
           if (ctl->refrac && z <= zrefrac) {
             for (i = 0; i < 3; i++)
04084
               xh[i] = x[i] + 0.5 * ds * ex0[i];
             cart2geo(xh, &z, &lon, &lat);
04085
04086
             intpol_atm(ctl, atm, z, &p, &t, q, k);
             n = refractivity(p, t);
for (i = 0; i < 3; i++) {
   xh[i] += h;</pre>
04087
04088
04089
04090
               cart2geo(xh, &z, &lon, &lat);
04091
               intpol_atm(ctl, atm, z, &p, &t, q, k);
               naux = refractivity(p, t);
04092
               naux - rerractivity(p,
ng[i] = (naux - n) / h;
xh[i] -= h;
04093
04094
04095
04096
           } else
             for (i = 0; i < 3; i++)
04097
04098
               ng[i] = 0;
04099
04100
           /* Construct new tangent vector (second term) ... */
```

```
04101
             for (i = 0; i < 3; i++)
04102
               ex1[i] += ds * ng[i];
04103
04104
             /\star Normalize new tangent vector... \star/
             norm = NORM(ex1);
for (i = 0; i < 3; i++)
  ex1[i] /= norm;</pre>
04105
04106
04107
04108
04109
              /\star Determine next point of LOS... \star/
             for (i = 0; i < 3; i++)
  x[i] += 0.5 * ds * (ex0[i] + ex1[i]);</pre>
04110
04111
04112
            /* Copy tangent vector... */
for (i = 0; i < 3; i++)</pre>
04113
04114
04115
               ex0[i] = ex1[i];
04116
04117
          /\star Get tangent point (to be done before changing segment lengths!)... \star/
04118
          tangent_point(los, &obs->tpz[ir], &obs->tplon[ir], &obs->
04119
       tplat[ir]);
04120
04121
           /\star Change segment lengths according to trapezoid rule... \star/
          for (ip = los->np - 1; ip >= 1; ip--)
los->ds[ip] = 0.5 * (los->ds[ip - 1] + los->ds[ip]);
los->ds[0] *= 0.5;
04122
04123
04124
04125
04126
           /\star Compute column density... \star/
04127
          for (ip = 0; ip < los->np; ip++)
            for (ig = 0; ig < ctl->ng; ig++)
  los->u[ig][ip] = 10 * los->q[ig][ip] * los->p[ip]
  / (KB * los->t[ip]) * los->ds[ip];
04128
04129
04130
04131 }
```

Here is the call graph for this function:



5.3.2.34 void read_atm (const char * dirname, const char * filename, ctl_t * ctl, atm_t * atm)

Read atmospheric data.

Definition at line 4135 of file jurassic.c.

```
04139 {
04140
04141 FILE *in;
04142
04143 char file[LEN], line[LEN], *tok;
```

```
04144
04145
           int ig, iw;
04146
04147
            /* Init... */
           atm->np = 0;
04148
04149
04150
            /* Set filename... */
04151
            if (dirname != NULL)
04152
              sprintf(file, "%s/%s", dirname, filename);
04153
            else
              sprintf(file, "%s", filename);
04154
04155
04156
           /* Write info... */
04157
           printf("Read atmospheric data: %s\n", file);
04158
04159
            /* Open file... */
           if (!(in = fopen(file, "r")))
04160
              ERRMSG("Cannot open file!");
04161
04162
04163
           /* Read line... */
04164
           while (fgets(line, LEN, in)) {
04165
              /* Read data... */

TOK(line, tok, "%lg", atm->time[atm->np]);

TOK(NULL, tok, "%lg", atm->z[atm->np]);

TOK(NULL, tok, "%lg", atm->lon[atm->np]);

TOK(NULL, tok, "%lg", atm->lat[atm->np]);

TOK(NULL, tok, "%lg", atm->[atm->np]);

TOK(NULL, tok, "%lg", atm->t[atm->np]);

TOK(NULL, tok, "%lg", atm->p[atm->np]);

for (ig = 0; ig < ctl->ng; ig++)

TOK(NULL, tok, "%lg", atm->q[ig][atm->np]);

for (iw = 0; iw < ctl->nw; iw++)

TOK(NULL, tok, "%lg", atm->k[iw][atm->np]);
04166
04167
04168
04169
04170
04171
04172
04173
04174
04175
04176
04177
              /* Increment data point counter... */
if ((++atm->np) > NP)
04178
04179
                  ERRMSG("Too many data points!");
04180
04181
04182
04183
            /* Close file... */
04184
           fclose(in);
04185
04186
           /* Check number of points... */
04187
            if (atm->np < 1)</pre>
               ERRMSG("Could not read any data!");
04188
04189 }
```

5.3.2.35 void read_ctl (int argc, char * argv[], ctl_t * ctl)

Read forward model control parameters.

Definition at line 4193 of file jurassic.c.

```
04196
04197
04198
       int id, ig, iw;
04199
04200
       /* Write info... */
       04201
04202
04203
                argv[0], __DATE__, __TIME__);
04204
04205
        /* Emitters... */
       ctl->ng = (int) scan_ctl(argc, argv, "NG", -1, "0", NULL);
if (ctl->ng < 0 || ctl->ng > NG)
04206
04207
         ERRMSG("Set 0 <= NG <= MAX!");
04208
        for (ig = 0; ig < ctl->ng; ig++)
    scan_ctl(argc, argv, "EMITTER", ig, "", ctl->emitter[ig]);
04209
04210
04211
04212
        /* Radiance channels... */
        ctl->nd = (int) scan_ctl(argc, argv, "ND", -1, "0", NULL);
04213
        if (ctl->nd < 0 || ctl->nd > ND)
04214
          ERRMSG("Set 0 <= ND <= MAX!");</pre>
04216
        for (id = 0; id < ctl->nd; id++)
04217
          ctl->nu[id] = scan_ctl(argc, argv, "NU", id, "", NULL);
04218
04219
        /* Spectral windows... */
       ctl->nw = (int) scan_ctl(argc, argv, "NW", -1, "1", NULL);
if (ctl->nw < 0 || ctl->nw > NW)
04220
04221
          ERRMSG("Set 0 <= NW <= MAX!");</pre>
```

```
for (id = 0; id < ctl->nd; id++)
04224
              ctl->window[id] = (int) scan_ctl(argc, argv, "WINDOW", id, "0", NULL);
04225
            /* Emissivity look-up tables... */
scan_ctl(argc, argv, "TBLBASE", -1, "-", ctl->tblbase);
04226
04227
04228
04229
             /* Hydrostatic equilibrium... */
04230
            ctl->hydz = scan_ctl(argc, argv, "HYDZ", -1, "-999", NULL);
04231
04232
            /* Continua... */
            ctl->ctm_co2 = (int) scan_ctl(argc, argv, "CTM_CO2", -1, "1", NULL);
ctl->ctm_h2o = (int) scan_ctl(argc, argv, "CTM_H2O", -1, "1", NULL);
ctl->ctm_n2 = (int) scan_ctl(argc, argv, "CTM_N2", -1, "1", NULL);
ctl->ctm_o2 = (int) scan_ctl(argc, argv, "CTM_O2", -1, "1", NULL);
04233
04234
04235
04236
04237
04238
            ctl->refrac = (int) scan_ctl(argc, argv, "REFRAC", -1, "1", NULL);
ctl->rayds = scan_ctl(argc, argv, "RAYDS", -1, "10", NULL);
ctl->raydz = scan_ctl(argc, argv, "RAYDZ", -1, "0.5", NULL);
04239
04240
04241
04242
            /* Field of view... */
scan_ctl(argc, argv, "FOV", -1, "-", ctl->fov);
04243
04244
04245
04246
            /* Retrieval interface... */
            /* Retrieval interface... */
ctl->retp_zmin = scan_ctl(argc, argv, "RETP_ZMIN", -1, "-999", NULL);
ctl->retp_zmax = scan_ctl(argc, argv, "RETP_ZMAX", -1, "-999", NULL);
ctl->rett_zmin = scan_ctl(argc, argv, "RETT_ZMIN", -1, "-999", NULL);
ctl->rett_zmax = scan_ctl(argc, argv, "RETT_ZMAX", -1, "-999", NULL);
04247
04248
04249
04250
04251
            for (ig = 0; ig < ctl->ng; ig++) {
             ctl->retq_zmin[ig] = scan_ctl(argc, argv, "RETO_ZMIN", ig, "-999", NULL);
ctl->retq_zmax[ig] = scan_ctl(argc, argv, "RETO_ZMAX", ig, "-999", NULL);
04252
04253
04254
04255
            for (iw = 0; iw < ctl->nw; iw++) {
04256
            ctl->retk_zmin[iw] = scan_ctl(argc, argv, "RETK_ZMIN", iw, "-999", NULL);
              ctl->retk_zmax[iw] = scan_ctl(argc, argv, "RETK_ZMAX", iw, "-999", NULL);
04257
04258
04259
04260
            /* Output flags... */
04261
            ctl->write_bbt = (int) scan_ctl(argc, argv, "WRITE_BBT", -1, "0", NULL);
04262
            ctl->write_matrix =
                (int) scan_ctl(argc, argv, "WRITE_MATRIX", -1, "0", NULL);
04263
04264 }
```

Here is the call graph for this function:



5.3.2.36 void read_matrix (const char * dirname, const char * filename, gsl_matrix * matrix)

Read matrix.

Definition at line 4268 of file jurassic.c.

```
04271
                              {
04272
04273
       FILE *in;
04275
       char dum[LEN], file[LEN], line[LEN];
04276
04277
       double value;
04278
04279
       int i. i:
04280
04281
       /* Set filename... */
```

```
04282
        if (dirname != NULL)
04283
          sprintf(file, "%s/%s", dirname, filename);
04284
        else
04285
          sprintf(file, "%s", filename);
04286
04287
        /* Write info... */
        printf("Read matrix: %s\n", file);
04288
04289
04290
         /* Open file... */
        if (!(in = fopen(file, "r")))
04291
          ERRMSG("Cannot open file!");
04292
04293
04294
        /* Read data... */
04295
        gsl_matrix_set_zero(matrix);
04296
        while (fgets(line, LEN, in))
04297
         if (sscanf(line, "%d %s %s %s %s %d %s %s %s %s %s %lg",
04298
                      &i, dum, dum, dum, dum, dum,
            &j, dum, dum, dum, dum, dum, &value) == 13)
gsl_matrix_set(matrix, (size_t) i, (size_t) j, value);
04299
04301
04302
         /* Close file... */
04303
        fclose(in);
04304 }
```

5.3.2.37 void read_obs (const char * dirname, const char * filename, ctl_t * ctl, obs_t * obs)

Read observation data.

Definition at line 4308 of file jurassic.c.

```
04312
04313
04314
             FILE *in:
04315
04316
             char file[LEN], line[LEN], *tok;
04317
04318
04319
04320
             /* Init... */
04321
             obs->nr = 0;
04322
04323
             /* Set filename... */
04324
             if (dirname != NULL)
04325
                sprintf(file, "%s/%s", dirname, filename);
04326
                sprintf(file, "%s", filename);
04327
04328
04329
             /* Write info... */
04330
             printf("Read observation data: %s\n", file);
04331
04332
             /* Open file... */
             if (!(in = fopen(file, "r")))
04333
                ERRMSG("Cannot open file!");
04334
04335
04336
             /* Read line... */
04337
             while (fgets(line, LEN, in)) {
04338
                 /* Read data... */
TOK(line, tok, "%lg", obs->time[obs->nr]);
TOK(NULL, tok, "%lg", obs->obsz[obs->nr]);
04339
04340
04341
                TOK (NULL, tok, "%lg", obs->obsz[obs->nr]);
TOK (NULL, tok, "%lg", obs->obslon[obs->nr]);
TOK (NULL, tok, "%lg", obs->obslat[obs->nr]);
TOK (NULL, tok, "%lg", obs->vpz[obs->nr]);
TOK (NULL, tok, "%lg", obs->vplon[obs->nr]);
TOK (NULL, tok, "%lg", obs->vplon[obs->nr]);
TOK (NULL, tok, "%lg", obs->tpz[obs->nr]);
TOK (NULL, tok, "%lg", obs->tpz[obs->nr]);
TOK (NULL, tok, "%lg", obs->tplat[obs->nr]);
TOK (NULL, tok, "%lg", obs->tplat[obs->nr]);
for (id = 0; id < ctl->nd; id+)
    TOK (NULL, tok, "%lg", obs->rad[id][obs->nr]);
for (id = 0; id < ctl->nd; id+)
    TOK (NULL, tok, "%lg", obs->tplat[obs->nr]);
04342
04343
04344
04345
04346
04347
04348
04349
04350
04351
04352
04353
04354
04355
                 /* Increment counter... */
04356
                 if ((++obs->nr) > NR)
                    ERRMSG("Too many rays!");
04357
04358
04359
04360
             /* Close file... */
04361
             fclose(in);
```

```
04362

04363  /* Check number of points... */

04364  if (obs->nr < 1)

04365  ERRMSG("Could not read any data!");

04366 }
```

5.3.2.38 void read_shape (const char * filename, double * x, double * y, int * n)

Read shape function.

Definition at line 4370 of file jurassic.c.

```
04374
04375
04376
       FILE *in;
04377
04378
        char line[LEN];
04379
04380
       /* Write info... */
04381
       printf("Read shape function: %s\n", filename);
04382
04383
       /* Open file... */
       if (!(in = fopen(filename, "r")))
04384
          ERRMSG("Cannot open file!");
04386
04387
        /* Read data... */
04388
       while (fgets(line, LEN, in))
  if (sscanf(line, "%lg %lg", &x[*n], &y[*n]) == 2)
04389
04390
           if ((++(*n)) > NSHAPE)
04391
04392
             ERRMSG("Too many data points!");
04393
04394
        /* Check number of points... */
       if (*n < 1)
04395
         ERRMSG("Could not read any data!");
04396
04397
04398
       /* Close file... */
04399 fclose(in);
04400 }
```

5.3.2.39 double refractivity (double p, double t)

Compute refractivity (return value is n - 1).

Definition at line 4404 of file jurassic.c.

```
04406

04407

04408  /* Refractivity of air at 4 to 15 micron... */

04409  return 7.753e-05 * p / t;

04410 }
```

5.3.2.40 double scan_ctl (int argc, char * argv[], const char * varname, int arridx, const char * defvalue, char * value)

Search control parameter file for variable entry.

Definition at line 4414 of file jurassic.c.

```
04421
04422
        FILE *in = NULL;
04423
04424
         char dummy[LEN], fullname1[LEN], fullname2[LEN], line[LEN],
04425
          msg[2 * LEN], rvarname[LEN], rval[LEN];
04426
04427
04428
        /* Open file... */
if (argv[1][0] != '-')
04429
04430
         if (!(in = fopen(argv[1], "r")))
04431
             ERRMSG("Cannot open file!");
04432
04433
04434
         /\star Set full variable name... \star/
04435
        if (arridx >= 0) {
         sprintf(fullname1, "%s[%d]", varname, arridx);
sprintf(fullname2, "%s[*]", varname);
04436
04437
04438
         } else {
          sprintf(fullname1, "%s", varname);
sprintf(fullname2, "%s", varname);
04439
04440
04441
04442
04443
         /* Read data... */
04444
         if (in != NULL)
         while (fgets(line, LEN, in))
04446
             if (sscanf(line, "%s %s %s", rvarname, dummy, rval) == 3)
               if (strcasecmp(rvarname, fullname1) == 0 ||
04447
04448
                    strcasecmp(rvarname, fullname2) == 0) {
04449
                  contain = 1;
04450
                 break:
04451
               }
04452
         for (i = 1; i < argc - 1; i++)</pre>
04453
         if (strcasecmp(argv[i], fullname1) == 0 ||
             strcasecmp(argv[i], fullname2) == 0) {
sprintf(rval, "%s", argv[i + 1]);
04454
04455
04456
             contain = 1;
04457
             break;
04458
04459
04460
        /* Close file... */
        if (in != NULL)
04461
04462
          fclose(in);
04463
04464
        /* Check for missing variables... */
04465
         if (!contain) {
         if (strlen(defvalue) > 0)
   sprintf(rval, "%s", defvalue);
04466
04467
           else {
04468
04469
            sprintf(msg, "Missing variable %s!\n", fullname1);
             ERRMSG (msg);
04471
04472
04473
04474
        /* Write info... */
04475
        printf("%s = %s\n", fullname1, rval);
04477
        /* Return values... */
04478
        if (value != NULL)
          sprintf(value, "%s", rval);
04479
04480
        return atof(rval);
04481 }
```

5.3.2.41 void tangent_point (los t * los, double * tpz, double * tplon, double * tplat)

Find tangent point of a given LOS.

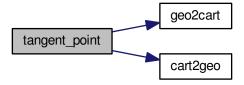
Definition at line 4485 of file jurassic.c.

```
04489
04490
04491
        double a, b, c, dummy, v[3], v0[3], v2[3], x, x1, x2, yy0, yy1, yy2;
04492
04493
       size_t i, ip;
04494
04495
        /\star Find minimum altitude... \star/
04496
       ip = gsl_stats_min_index(los->z, 1, (size_t) los->np);
04497
04498
       /* Nadir or zenith... */
04499
       if (ip <= 0 || ip >= (size_t) los->np - 1) {
```

```
*tpz = los -> z[los -> np - 1];
          *tplon = los->lon[los->np - 1];

*tplat = los->lat[los->np - 1];
04501
04502
04503
04504
04505
        /* Limb... */
04506
        else {
04507
04508
           /* Determine interpolating polynomial y=a*x^2+b*x+c...*/
04509
          yy0 = los -> z[ip - 1];
          yy1 = los \rightarrow z[ip];
04510
04511
          yy2 = los -> z[ip + 1];
          x1 = sqrt (POW2(los->ds[ip]) - POW2(yy1 - yy0));
04512
04513
          x2 = x1 + sqrt(POW2(los->ds[ip + 1]) - POW2(yy2 - yy1));
04514
           a = 1 / (x1 - x2) * (-(yy0 - yy1) / x1 + (yy0 - yy2) / x2);
          b = -(yy0 - yy1) / x1 - a * x1;
04515
          c = yy0;
04516
04517
04518
          /* Get tangent point location... */
04519
          x = -b / (2 * a);
04520
           *tpz = a * x * x + b * x + c;
04521
           geo2cart(los->z[ip - 1], los->lon[ip - 1], los->lat[ip - 1], v0);
           geo2cart(los->z[ip + 1], los->lon[ip + 1], los->lat[ip + 1], v2);
04522
          for (i = 0; i < 3; i++)
v[i] = LIN(0.0, v0[i], x2, v2[i], x);
04523
04524
04525
          cart2geo(v, &dummy, tplon, tplat);
04526
04527 }
```

Here is the call graph for this function:



5.3.2.42 void time2jsec (int year, int mon, int day, int hour, int min, int sec, double remain, double * jsec)

Convert date to seconds.

Definition at line 4531 of file jurassic.c.

```
04539
                      {
04540
04541
       struct tm t0, t1;
04542
04543
        t0.tm_year = 100;
04544
        t0.tm\_mon = 0;
        t0.tm_mday = 1;
04545
       t0.tm_hour = 0;
04546
        t0.tm_min = 0;
04547
04548
        t0.tm\_sec = 0;
04549
04550
        t1.tm_year = year - 1900;
04551
        t1.tm_mon = mon - 1;
04552
        t1.tm_mday = day;
        t1.tm_hour = hour;
04553
04554
        t1.tm_min = min;
       t1.tm_sec = sec;
04556
04557
        *jsec = (double) timegm(&t1) - (double) timegm(&t0) + remain;
04558 }
```

5.3.2.43 void timer (const char * name, const char * file, const char * func, int line, int mode)

Measure wall-clock time.

Definition at line 4562 of file jurassic.c.

```
04567
                   {
04568
04569
        static double w0[10];
04571
        static int 10[10], nt;
04572
04573
        /* Start new timer... */
04574
        if (mode == 1) {
         w0[nt] = omp_get_wtime();
10[nt] = line;
04575
             ((++nt) >= 10)
04577
         if
04578
            ERRMSG("Too many timers!");
04579
04580
04581
        /* Write elapsed time... */
04582
        else {
04583
04584
          /\star Check timer index... \star/
          if (nt - 1 < 0)</pre>
04585
            ERRMSG("Coding error!");
04586
04587
04588
          /* Write elapsed time... */
         printf("Timer '%s' (%s, %s, 1%d-%d): %.3f sec\n",
04590
                name, file, func, 10[nt - 1], line, omp_get_wtime() - w0[nt - 1]);
04591
04592
04593
        /* Stop timer... */
04594
        if (mode == 3)
04595
          nt--;
04596 }
```

5.3.2.44 void write_atm (const char * dirname, const char * filename, ctl_t * ctl, atm_t * atm)

Write atmospheric data.

Definition at line 4600 of file jurassic.c.

```
04604
04605
04606
        FILE *out;
04607
04608
        char file[LEN];
04609
04610
        int ig, ip, iw, n = 6;
04611
         /* Set filename... */
04612
04613
        if (dirname != NULL)
          sprintf(file, "%s/%s", dirname, filename);
04614
        else
04615
04616
          sprintf(file, "%s", filename);
04617
04618
        /\star Write info... \star/
04619
        printf("Write atmospheric data: %s\n", file);
04620
04621
        /* Create file... */
04622
        if (!(out = fopen(file, "w")))
          ERRMSG("Cannot create file!");
04623
04624
04625
        /* Write header... */
04626
        fprintf(out,
                 "# $1 = time (seconds since 2000-01-01T00:00Z) \n"
04627
                 "# $2 = altitude [km] \n"
04628
                 "# $3 = longitude [deg] \n"
04629
04630
                 "# $4 = latitude [deg] \n"
04631
                 "# $5 = pressure [hPa] \n" "# $6 = temperature [K] \n");
        for (ig = 0; ig < ctl->ng; ig++)
  fprintf(out, "# $%d = %s volume mixing ratio\n", ++n, ctl->emitter[ig]);
for (iw = 0; iw < ctl->nw; iw++)
04632
04633
04634
04635
          fprintf(out, "# \$%d = window %d: extinction [1/km]\n", ++n, iw);
04636
```

```
04637
          /* Write data... */
04638
          for (ip = 0; ip < atm->np; ip++) {
04639
             if (ip == 0 || atm->lat[ip] != atm->lat[ip - 1]
            || atm->lon[ip] != atm->lon[ip - 1])
fprintf(out, "\n");
fprintf(out, "%.2f %g %g %g %g", atm->time[ip], atm->z[ip],
04640
04641
04642
                      atm->lon[ip], atm->lat[ip], atm->p[ip], atm->t[ip]);
04643
             for (ig = 0; ig < ctl->ng; ig++)
  fprintf(out, " %g", atm->q[ig][ip]);
04644
04645
             for (iw = 0; iw < ctl->nw; iw+)
  fprintf(out, " %g", atm->k[iw][ip]);
fprintf(out, "\n");
04646
04647
04648
04649
04650
04651
          /* Close file... */
04652
         fclose(out);
04653 }
```

5.3.2.45 void write_matrix (const char * dirname, const char * filename, ctl_t * ctl, gsl_matrix * matrix, atm_t * atm, obs_t * obs, const char * rowspace, const char * colspace, const char * sort)

Write matrix.

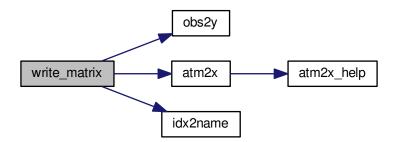
Definition at line 4657 of file jurassic.c.

```
04666
04667
04668
        FILE *out;
04669
        char file[LEN], quantity[LEN];
04671
04672
        int *cida, *ciqa, *cipa, *cira, *rida, *riqa, *ripa, *rira;
04673
        size t i, j, nc, nr;
04674
04675
04676
        /* Check output flag... */
04677
        if (!ctl->write_matrix)
04678
04679
        /* Allocate... */
04680
        ALLOC(cida, int, M);
04681
04682
        ALLOC(ciqa, int,
04683
              N);
04684
        ALLOC(cipa, int,
04685
              N);
        ALLOC(cira, int,
04686
04687
              M);
04688
        ALLOC(rida, int,
              M);
04690
        ALLOC(riqa, int,
04691
              N);
        ALLOC(ripa, int,
04692
04693
              N);
04694
        ALLOC(rira, int,
04695
             M);
04696
04697
        /* Set filename... */
        if (dirname != NULL)
04698
         sprintf(file, "%s/%s", dirname, filename);
04699
04700
        else
04701
          sprintf(file, "%s", filename);
04702
04703
        /* Write info... */
04704
        printf("Write matrix: %s\n", file);
04705
04706
        /* Create file... */
        if (!(out = fopen(file, "w")))
04707
04708
          ERRMSG("Cannot create file!");
04709
04710
        /* Write header (row space)... */
04711
        if (rowspace[0] == 'y') {
04712
          fprintf(out,
04714
                   "# $1 = Row: index (measurement space) \n"
04715
                   "# $2 = Row: channel wavenumber [cm^-1]\n"
04716
                   "# \$3 = \text{Row: time (seconds since 2000-01-01T00:00Z)} \n"
                   "# $4 = Row: view point altitude [km]\n"
"# $5 = Row: view point longitude [deg]\n"
04717
04718
04719
                   "# $6 = Row: view point latitude [deg]\n");
04720
```

```
04721
           /* Get number of rows...
04722
          nr = obs2y(ctl, obs, NULL, rida, rira);
04723
04724
        } else {
04725
04726
          fprintf(out,
04727
                    "# $1 = Row: index (state space)\n"
04728
                    "# $2 = Row: name of quantity n"
04729
                    "# \$3 = Row: time (seconds since 2000-01-01T00:00Z)\n"
04730
                    "# $4 = Row: altitude [km]\n"
                    "# $5 = Row: longitude [deg]\n" "# $6 = Row: latitude [deg]\n");
04731
04732
04733
           /* Get number of rows... */
04734
          nr = atm2x(ctl, atm, NULL, riqa, ripa);
04735
04736
04737
         /* Write header (column space)... */
04738
        if (colspace[0] == 'y') {
04740
           fprintf(out,
04741
                    "# \$7 = \text{Col: index (measurement space)} \n"
                    "# $8 = Col: channel wavenumber [cm^-1]\n"
04742
                    "# $9 = Col: time (seconds since 2000-01-01T00:00Z)\n"
04743
                    "# $10 = Col: view point altitude [km]\n" "# $11 = Col: view point longitude [deg]\n"
04744
04745
04746
                    "# $12 = Col: view point latitude [deg]\n");
04747
           /\star Get number of columns... \star/
04748
04749
          nc = obs2y(ctl, obs, NULL, cida, cira);
04750
04751
        } else {
04752
04753
           fprintf(out,
04754
                    "# $7 = Col: index (state space) \n"
                    "# $8 = Col: name of quantity\n"
04755
                    "# $9 = Col: time (seconds since 2000-01-01T00:00Z)\n"
04756
04757
                    "# $10 = Col: altitude [km] \n"
04758
                    "# $11 = Col: longitude [deg]\n" "# $12 = Col: latitude [deg]\n");
04759
04760
           /* Get number of columns... */
04761
          nc = atm2x(ctl, atm, NULL, ciqa, cipa);
04762
04763
        /* Write header entry... */
fprintf(out, "# $13 = Matrix element n'");
04764
04765
04766
04767
         /* Write matrix data... */
04768
        i = j = 0;
        while (i < nr && j < nc) {
04769
04770
04771
           /* Write info about the row... */
          if (rowspace[0] == 'y')
  fprintf(out, "%d %g %.2f %g %g %g",
04772
04773
04774
                      (int) i, ctl->nu[rida[i]],
04775
                      obs->time[rira[i]], obs->vpz[rira[i]],
04776
                      obs->vplon[rira[i]], obs->vplat[rira[i]]);
04777
             idx2name(ctl, riqa[i], quantity);
fprintf(out, "%d %s %.2f %g %g %g", (int) i, quantity,
04778
04779
04780
                      atm->time[ripa[i]], atm->z[ripa[i]],
04781
                      atm->lon[ripa[i]], atm->lat[ripa[i]]);
04782
           }
04783
04784
           /\star Write info about the column... \star/
           if (colspace[0] == 'y')
  fprintf(out, " %d %g %.2f %g %g %g",
04785
04786
04787
                      (int) j, ctl->nu[cida[j]],
                      obs->time[cira[j]], obs->vpz[cira[j]],
obs->vplon[cira[j]], obs->vplat[cira[j]]);
04788
04789
04790
           else {
             idx2name(ctl, ciqa[j], quantity);
fprintf(out, " %d %s %.2f %g %g %g", (int) j, quantity,
04791
04792
                      atm->time[cipa[j]], atm->z[cipa[j]],
04793
04794
                      atm->lon[cipa[j]], atm->lat[cipa[j]]);
04795
           }
04796
04797
           /* Write matrix entry... */
04798
           fprintf(out, " %g\n", gsl_matrix_get(matrix, i, j));
04799
04800
           /* Set matrix indices... */
           if (sort[0] == 'r') {
04801
04802
             j++;
04803
             if (j >= nc) {
04804
               j = 0;
04805
               i++:
               fprintf(out, "\n");
04806
04807
```

```
04808
          } else {
04809
            i++;
            if (i >= nr) {
  i = 0;
04810
04811
04812
              j++;
04813
              fprintf(out, "\n");
04814
            }
04815
04816
04817
        /* Close file... */
04818
04819
        fclose(out);
04820
04821
        /* Free... */
04822
        free(cida);
04823
        free(ciqa);
04824
        free(cipa);
04825
        free(cira);
04826
        free(rida);
        free(riqa);
04828
        free(ripa);
04829
        free(rira);
04830 }
```

Here is the call graph for this function:



5.3.2.46 void write_obs (const char * dirname, const char * filename, ctl_t * ctl, obs_t * obs)

Write observation data.

Definition at line 4834 of file jurassic.c.

```
04838
04839
04840
        FILE *out;
04841
04842
        char file[LEN];
04843
04844
        int id, ir, n = 10;
04845
04846
         /* Set filename...
        if (dirname != NULL)
   sprintf(file, "%s/%s", dirname, filename);
04847
04848
04849
        else
          sprintf(file, "%s", filename);
04850
04851
04852
        /* Write info... */
04853
        printf("Write observation data: %s\n", file);
04854
        /* Create file... */
if (!(out = fopen(file, "w")))
04855
04856
04857
          ERRMSG("Cannot create file!");
04858
```

```
04859
                         /* Write header... */
04860
                        fprintf(out,
04861
                                                 "# $1 = time (seconds since 2000-01-01T00:00Z) \n"
                                                "# $2 = observer altitude [km] \n"
04862
                                                 "# $3 = observer longitude [deg] \n"
04863
                                                "# $4 = observer latitude [deg]\n"
04864
                                                "# $5 = view point altitude [km]\n"
04865
04866
                                                "# $6 = view point longitude [deg]\n"
04867
                                                "# $7 = \text{view point latitude [deg]} \n"
                                                 "# $8 = tangent point altitude [km]\n"
04868
                                                "# $9 = tangent point longitude [deg]\n"
04869
                                                "# $10 = tangent point latitude [deg]\n");
04870
                        for (id = 0; id < ctl->nd; id++)
04871
04872
                         fprintf(out, "# $%d = channel %g: radiance [W/(m^2 sr cm^-1)]\n",
04873
                                                      ++n, ctl->nu[id]);
                      for (id = 0; id < ctl->nd; id++)
  fprintf(out, "# $%d = channel %g: transmittance\n", ++n, ctl->nu[id]);
04874
04875
04876
                        /* Write data... */
04877
04878
                        for (ir = 0; ir < obs->nr; ir++) {
                           cor (ir = 0; ir < obs->nr; ir++) {
   if (ir == 0 || obs->time[ir] != obs->time[ir - 1])
        fprintf(out, "\n");
   fprintf(out, "%.2f %g %g %g %g %g %g %g %g %g", obs->time[ir],
        obs->obsz[ir], obs->obslon[ir], obs->obslat[ir],
        obs->vpz[ir], obs->vplon[ir], obs->vplat[ir],
        obs->tpz[ir], obs->tplon[ir], obs->tplat[ir]);
        for (id = 0.1 ord | id | left | 
04879
04880
04881
04882
04883
04884
                              for (id = 0; id < ctl->nd; id++)
  fprintf(out, " %g", obs->rad[id][ir]);
04885
04886
                             for (id = 0; id < ctl->nd; id+)
fprintf(out, " %g", obs->tau[id][ir]);
fprintf(out, "\n");
04887
04888
04889
04890
04891
04892
                        /* Close file... */
04893
                       fclose(out);
04894 }
```

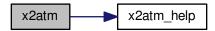
5.3.2.47 void x2atm (ctl_t * ctl, gsl_vector * x, atm_t * atm)

Decompose parameter vector or state vector.

Definition at line 4898 of file jurassic.c.

```
04902
04903
        int ig, iw;
04904
04905
       size_t n = 0;
04906
04907
        /* Set pressure... */
       x2atm_help(atm, ctl->retp_zmin, ctl->retp_zmax, atm->
04908
     p, x, &n);
04909
04910
        /* Set temperature... */
04911
       x2atm_help(atm, ctl->rett_zmin, ctl->rett_zmax, atm->
      t, x, &n);
04912
04913
        /\star Set volume mixing ratio... \star/
04914
        for (ig = 0; ig < ctl->ng; ig++)
04915
         x2atm_help(atm, ctl->retq_zmin[ig], ctl->retq_zmax[ig],
04916
                     atm->q[iq], x, &n);
04917
04918
       /* Set extinction... */
04919
        for (iw = 0; iw < ctl->nw; iw++)
04920
         x2atm_help(atm, ctl->retk_zmin[iw], ctl->retk_zmax[iw],
04921
                     atm->k[iw], x, &n);
04922 }
```

Here is the call graph for this function:



5.4 jurassic.c 93

5.3.2.48 void x2atm_help (atm_t * atm, double zmin, double zmax, double * value, gsl_vector * x, size_t * n)

Extract elements from state vector.

Definition at line 4926 of file jurassic.c.

```
04932
04933
04934
         int ip;
04935
04936
         /* Extract state vector elements... */
         for (ip = 0; ip < atm->np; ip++)
  if (atm->z[ip] >= zmin && atm->z[ip] <= zmax) {</pre>
04937
04939
             value[ip] = gsl_vector_get(x, *n);
04940
               (*n)++;
04941
            }
04942 }
```

5.3.2.49 void y2obs (ctl_t * ctl, gsl_vector * y, obs_t * obs)

Decompose measurement vector.

Definition at line 4946 of file jurassic.c.

```
04949
04950
04951
          int id, ir;
04952
04953
          size_t m = 0;
04955
           /* Decompose measurement vector... */
          for (ir = 0; ir < obs->nr; ir++)
  for (id = 0; id < ctl->nd; id++)
   if (gsl_finite(obs->rad[id][ir])) {
04956
04957
04958
04959
                 obs->rad[id][ir] = gsl_vector_get(y, m);
04960
                  m++;
04961
04962 }
```

5.4 jurassic.c

```
00001 /*
00002
       This file is part of JURASSIC.
00003
00004
        JURASSIC is free software: you can redistribute it and/or modify
00005
        it under the terms of the GNU General Public License as published by
00006
        the Free Software Foundation, either version 3 of the License, or
00007
        (at your option) any later version.
80000
00009
        {\tt JURASSIC} is distributed in the hope that it will be useful,
       but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
00010
00011
00012
       GNU General Public License for more details.
00013
00014
        You should have received a copy of the GNU General Public License
00015
       along with JURASSIC. If not, see <a href="http://www.gnu.org/licenses/">http://www.gnu.org/licenses/</a>.
00016
00017
        Copright (C) 2003-2015 Forschungszentrum Juelich GmbH
00018 */
00019
00025 #include "jurassic.h"
00026
00028
00029 size_t atm2x(
00030
      ctl_t * ctl,
00031
00032
        gsl_vector * x,
00033
        int *iqa,
00034
       int *ipa) {
00035
00036
       int ig, iw;
```

```
00037
00038
       size_t n = 0;
00039
00040
       /* Add pressure... */
00041
       atm2x_help(atm, ctl->retp_zmin, ctl->retp_zmax,
00042
                 atm->p, IDXP, x, iqa, ipa, &n);
00043
00044
       /* Add temperature... */
00045
       atm2x_help(atm, ctl->rett_zmin, ctl->rett_zmax,
00046
                  atm->t, IDXT, x, iqa, ipa, &n);
00047
       /* Add volume mixing ratios... */
00048
       for (ig = 0; ig < ctl->ng; ig++)
  atm2x_help(atm, ctl->retq_zmin[ig], ctl->retq_zmax[ig],
00049
00050
00051
                    atm->q[ig], IDXQ(ig), x, iqa, ipa, &n);
00052
00053
       /* Add extinction... */
       for (iw = 0; iw < ctl->nw; iw++)
00054
       atm2x_help(atm, ctl->retk_zmin[iw], ctl->retk_zmax[iw],
00055
00056
                   atm->k[iw], IDXK(iw), x, iqa, ipa, &n);
00057
00058
       return n;
00059 }
00060
00062
00063 void atm2x_help(
00064 atm_t * atm,
00065
       double zmin,
00066
       double zmax,
00067
       double *value,
00068
       int val_iqa,
00069
       gsl_vector * x,
00070
       int *iqa,
00071
       int *ipa,
00072
       size_t * n) {
00073
00074
       int ip;
00075
00076
       /* Add elements to state vector... */
       for (ip = 0; ip < atm->np; ip++)
  if (atm->z[ip] >= zmin && atm->z[ip] <= zmax) {
   if (x != NULL)</pre>
00077
00078
00079
08000
            gsl_vector_set(x, *n, value[ip]);
00081
           if (iqa != NULL)
00082
             iqa[*n] = val_iqa;
00083
           if (ipa != NULL)
            ipa[*n] = ip;
00084
00085
           (*n)++;
00086
00087 }
00088
00090
00091 double brightness (
00092
       double rad,
00093
       double nu) {
00094
00095
       return C2 * nu / gsl_log1p(C1 * POW3(nu) / rad);
00096 }
00097
00098
00100
00101 void cart2geo(
00102 double *x,
00103
       double *z,
       double *lon,
00104
00105
      double *lat) {
00106
00107
       double radius;
00108
       radius = NORM(x);
*lat = asin(x[2] / radius) * 180 / M_PI;
*lon = atan2(x[1], x[0]) * 180 / M_PI;
00109
00110
00111
00112
       *z = radius - RE;
00113 }
00114
00116
00117 void climatology(
00118
       ctl_t * ctl,
00119
00120
       static double z[121] = {
  0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19,
  20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37,
00121
00122
00123
```

5.4 jurassic.c 95

```
38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55,
            56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91,
00125
00126
            92, 93, 94, 95, 96, 97, 98, 99, 100, 101, 102, 103, 104, 105, 106, 107,
00127
00128
            108, 109, 110, 111, 112, 113, 114, 115, 116, 117, 118, 119, 120
00129
00130
00131
         static double pre[121] = {
00132
            1017, 901.083, 796.45, 702.227, 617.614, 541.644, 473.437, 412.288,
            357.603, 308.96, 265.994, 228.348, 195.619, 167.351, 143.039, 122.198, 104.369, 89.141, 76.1528, 65.0804, 55.641, 47.591, 40.7233, 34.8637,
00133
00134
            29.8633, 25.5956, 21.9534, 18.8445, 16.1909, 13.9258, 11.9913,
00135
            10.34, 8.92988, 7.72454, 6.6924, 5.80701, 5.04654, 4.39238, 3.82902,
00136
            3.34337, 2.92413, 2.56128, 2.2464, 1.97258, 1.73384, 1.52519, 1.34242,
00137
00138
            1.18197, 1.04086, 0.916546, 0.806832, 0.709875, 0.624101, 0.548176,
            0.480974, 0.421507, 0.368904, 0.322408, 0.281386, 0.245249, 0.213465, 0.185549, 0.161072, 0.139644, 0.120913, 0.104568, 0.0903249, 0.0779269,
00139
00140
            0.0671493, 0.0577962, 0.0496902, 0.0426736, 0.0366093, 0.0313743, 0.0268598, 0.0229699, 0.0196206, 0.0167399, 0.0142646, 0.0121397,
00141
            0.0103181, 0.00875775, 0.00742226, 0.00628076, 0.00530519, 0.00447183,
00143
            0.00376124, 0.00315632, 0.00264248, 0.00220738, 0.00184003, 0.00153095,
00144
00145
            0.00127204,\ 0.00105608,\ 0.000876652,\ 0.00072798,\ 0.00060492,
            0.000503201, 0.000419226, 0.000349896, 0.000292659, 0.000245421, 0.000206394, 0.000174125, 0.000147441, 0.000125333, 0.000106985,
00146
00147
00148
            9.173e-05, 7.90172e-05, 6.84172e-05, 5.95574e-05, 5.21183e-05,
            4.58348e-05, 4.05127e-05, 3.59987e-05, 3.21583e-05, 2.88718e-05, 2.60322e-05, 2.35687e-05, 2.14263e-05, 1.95489e-05
00150
00151
00152
00153
         static double tem[121] = {
           285.14, 279.34, 273.91, 268.3, 263.24, 256.55, 250.2, 242.82, 236.17, 229.87, 225.04, 221.19, 218.85, 217.19, 216.2, 215.68, 215.42, 215.55,
00154
00155
            215.92, 216.4, 216.93, 217.45, 218, 218.68, 219.39, 220.25, 221.3, 222.41, 223.88, 225.42, 227.2, 229.52, 231.89, 234.51, 236.85, 239.42,
00156
00157
            241.94, 244.57, 247.36, 250.32, 253.34, 255.82, 258.27, 260.39, 262.03, 263.45, 264.2, 264.78, 264.67, 264.38, 263.24, 262.03, 260.02,
00158
00159
            258.09, 255.63, 253.28, 250.43, 247.81, 245.26, 242.77, 240.38, 237.94, 235.79, 233.53, 231.5, 229.53, 227.6, 225.62, 223.77, 222.06,
00160
00162
            220.33, 218.69, 217.18, 215.64, 214.13, 212.52, 210.86, 209.25,
            207.49, 205.81, 204.11, 202.22, 200.32, 198.39, 195.92, 193.46,
00163
00164
            190.94, 188.31, 185.82, 183.57, 181.43, 179.74, 178.64, 178.1, 178.25,
           178.7, 179.41, 180.67, 182.31, 184.18, 186.6, 189.53, 192.66, 196.54, 201.13, 205.93, 211.73, 217.86, 225, 233.53, 242.57, 252.14, 261.48, 272.97, 285.26, 299.12, 312.2, 324.17, 338.34, 352.56, 365.28
00165
00166
00167
00169
00170
         static double c2h2[121] = {
           1.352e-09, 2.83e-10, 1.269e-10, 6.926e-11, 4.346e-11, 2.909e-11,
00171
            2.014e-11, 1.363e-11, 8.71e-12, 5.237e-12, 2.718e-12, 1.375e-12,
00172
            5.786e-13, 2.16e-13, 7.317e-14, 2.551e-14, 1.055e-14, 4.758e-15,
00173
00174
            2.056e-15, 7.703e-16, 2.82e-16, 1.035e-16, 4.382e-17, 1.946e-17,
00175
            9.638e-18, 5.2e-18, 2.811e-18, 1.494e-18, 7.925e-19, 4.213e-19,
00176
            1.998e-19, 8.78e-20, 3.877e-20, 1.728e-20, 7.743e-21, 3.536e-21,
            1.623e-21, 7.508e-22, 3.508e-22, 1.65e-22, 7.837e-23, 3.733e-23, 1.808e-23, 8.77e-24, 4.285e-24, 2.095e-24, 1.032e-24, 5.082e-25,
00177
00178
00179
            2.506e-25, 1.236e-25, 6.088e-26, 2.996e-26, 1.465e-26, 0, 0, 0,
            00181
            00182
00183
00184
00185
         static double c2h6[121] = {
            2.667e-09, 2.02e-09, 1.658e-09, 1.404e-09, 1.234e-09, 1.109e-09,
            1.012e-09, 9.262e-10, 8.472e-10, 7.71e-10, 6.932e-10, 6.216e-10, 5.503e-10, 4.87e-10, 4.342e-10, 3.861e-10, 3.347e-10, 2.772e-10,
00187
00188
00189
            2.209e-10, 1.672e-10, 1.197e-10, 8.536e-11, 5.783e-11, 3.846e-11,
00190
            2.495e-11, 1.592e-11, 1.017e-11, 6.327e-12, 3.895e-12, 2.403e-12, 1.416e-12, 8.101e-13, 4.649e-13, 2.686e-13, 1.557e-13, 9.14e-14,
00191
            5.386e-14, 3.19e-14, 1.903e-14, 1.14e-14, 6.875e-15, 4.154e-15,
00192
00193
            2.538e-15, 1.553e-15, 9.548e-16, 5.872e-16, 3.63e-16, 2.244e-16,
            1.388e-16, 8.587e-17, 5.308e-17, 3.279e-17, 2.017e-17, 1.238e-17,
00194
            7.542e-18, 4.585e-18, 2.776e-18, 1.671e-18, 9.985e-19, 5.937e-19, 3.518e-19, 2.07e-19, 1.215e-19, 7.06e-20, 4.097e-20, 2.37e-20, 1.363e-20, 7.802e-21, 4.441e-21, 2.523e-21, 1.424e-21, 8.015e-22,
00195
00196
00197
00198
            4.497e-22, 2.505e-22, 1.391e-22, 7.691e-23, 4.238e-23, 2.331e-23,
            1.274e-23, 6.929e-24, 3.752e-24, 2.02e-24, 1.083e-24, 5.774e-25,
00199
            3.041e-25, 1.593e-25, 8.308e-26, 4.299e-26, 2.195e-26, 1.112e-26,
00200
00201
            00202
            0, 0, 0, 0, 0, 0, 0, 0
00203
00204
         static double ccl4[121] = {
           1.075e-10, 1.075e-10, 1.075e-10, 1.075e-10, 1.075e-10, 1.075e-10,
00206
00207
            1.075e-10, 1.075e-10, 1.075e-10, 1.06e-10, 1.024e-10, 9.69e-11,
00208
            8.93e-11, 8.078e-11, 7.213e-11, 6.307e-11, 5.383e-11, 4.49e-11,
            3.609e-11, 2.705e-11, 1.935e-11, 1.385e-11, 8.35e-12, 5.485e-12, 3.853e-12, 2.22e-12, 5.875e-13, 3.445e-13, 1.015e-13, 6.075e-14,
00209
00210
```

```
4.383e-14, 2.692e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14,
                     le-14, le
00212
00213
00214
                     1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14,
00215
                     1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14,
00216
                     le-14, le
                     1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14,
                     le-14, le-14, le-14, le-14, le-14, le-14, le-14, le-14, le-14, le-14,
00218
00219
                     1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14,
00220
                     1e-14, 1e-14, 1e-14
00221
00222
00223
                static double ch4[121] = {
                  1.864e-06, 1.835e-06, 1.819e-06, 1.805e-06, 1.796e-06, 1.788e-06,
00224
00225
                     1.782e-06, 1.776e-06, 1.769e-06, 1.761e-06, 1.749e-06, 1.734e-06,
00226
                     1.716e-06, 1.692e-06, 1.654e-06, 1.61e-06, 1.567e-06, 1.502e-06,
00227
                     1.433e-06, 1.371e-06, 1.323e-06, 1.277e-06, 1.232e-06, 1.188e-06,
                     1.147e-06, 1.108e-06, 1.07e-06, 1.027e-06, 9.854e-07, 9.416e-07, 8.933e-07, 8.478e-07, 7.988e-07, 7.515e-07, 7.07e-07, 6.64e-07,
00228
                     6.239e-07, 5.864e-07, 5.512e-07, 5.184e-07, 4.87e-07, 4.571e-07,
00230
                    4.296e-07, 4.04e-07, 3.802e-07, 3.578e-07, 3.383e-07, 3.203e-07, 3.032e-07, 2.889e-07, 2.76e-07, 2.635e-07, 2.519e-07, 2.409e-07,
00231
00232
                     2.302e-07, 2.219e-07, 2.144e-07, 2.071e-07, 1.999e-07, 1.93e-07,
00233
                     1.862e-07, 1.795e-07, 1.731e-07, 1.668e-07, 1.607e-07, 1.548e-07,
00234
                     1.49e-07, 1.434e-07, 1.38e-07, 1.328e-07, 1.277e-07, 1.227e-07, 1.18e-07, 1.134e-07, 1.089e-07, 1.046e-07, 1.004e-07, 9.635e-08,
00235
00236
                     9.245e-08, 8.867e-08, 8.502e-08, 8.15e-08, 7.809e-08, 7.48e-08,
00237
00238
                     7.159e-08, 6.849e-08, 6.55e-08, 6.262e-08, 5.98e-08, 5.708e-08,
00239
                     5.448e-08, 5.194e-08, 4.951e-08, 4.72e-08, 4.5e-08, 4.291e-08,
00240
                     4.093e-08, 3.905e-08, 3.729e-08, 3.563e-08, 3.408e-08, 3.265e-08,
00241
                     3.128e-08, 2.996e-08, 2.87e-08, 2.76e-08, 2.657e-08, 2.558e-08,
00242
                     2.467e-08, 2.385e-08, 2.307e-08, 2.234e-08, 2.168e-08, 2.108e-08,
                     2.05e-08, 1.998e-08, 1.947e-08, 1.902e-08, 1.86e-08, 1.819e-08,
00243
                     1.782e-08
00244
00245
00246
00247
                static double clo[121] = {
                    7.419e-15, 1.061e-14, 1.518e-14, 2.195e-14, 3.175e-14, 4.666e-14,
00249
                     6.872e-14, 1.03e-13, 1.553e-13, 2.375e-13, 3.664e-13, 5.684e-13,
00250
                     8.915e-13, 1.402e-12, 2.269e-12, 4.125e-12, 7.501e-12, 1.257e-11,
00251
                     2.048e-11, 3.338e-11, 5.44e-11, 8.846e-11, 1.008e-10, 1.082e-10,
                     1.157e-10, 1.232e-10, 1.312e-10, 1.539e-10, 1.822e-10, 2.118e-10,
00252
                     2.387e-10, 2.687e-10, 2.875e-10, 3.031e-10, 3.23e-10, 3.648e-10, 4.117e-10, 4.477e-10, 4.633e-10, 4.794e-10, 4.95e-10, 5.104e-10,
00253
00254
                     5.259e-10, 5.062e-10, 4.742e-10, 4.443e-10, 4.051e-10, 3.659e-10,
00255
00256
                     3.305e-10, 2.911e-10, 2.54e-10, 2.215e-10, 1.927e-10, 1.675e-10,
00257
                    1.452e-10, 1.259e-10, 1.09e-10, 9.416e-11, 8.119e-11, 6.991e-11,
00258
                     6.015e-11,\ 5.163e-11,\ 4.43e-11,\ 3.789e-11,\ 3.24e-11,\ 2.769e-11,
                     2.361e-11, 2.011e-11, 1.71e-11, 1.453e-11, 1.233e-11, 1.045e-11,
00259
                     8.851e-12, 7.48e-12, 6.316e-12, 5.326e-12, 4.487e-12, 3.778e-12,
00260
                     3.176e-12, 2.665e-12, 2.234e-12, 1.87e-12, 1.563e-12, 1.304e-12,
                     1.085e-12, 9.007e-13, 7.468e-13, 6.179e-13, 5.092e-13, 4.188e-13,
00262
00263
                     3.442e-13, 2.816e-13, 2.304e-13, 1.885e-13, 1.542e-13, 1.263e-13,
                    1.035e-13, 8.5e-14, 7.004e-14, 5.783e-14, 4.795e-14, 4.007e-14, 3.345e-14, 2.792e-14, 2.33e-14, 1.978e-14, 1.686e-14, 1.438e-14, 1.234e-14, 1.07e-14, 9.312e-15, 8.131e-15, 7.164e-15, 6.367e-15, 5.67e-15, 5.088e-15, 4.565e-15, 4.138e-15, 3.769e-15, 3.432e-15,
00264
00265
00266
00268
                     3.148e-15
00269
00270
00271
                static double clono2[121] = {
                 1.011e-13, 1.515e-13, 2.272e-13, 3.446e-13, 5.231e-13, 8.085e-13,
00272
                     1.253e-12, 1.979e-12, 3.149e-12, 5.092e-12, 8.312e-12, 1.366e-11,
                    2.272e-11, 3.791e-11, 6.209e-11, 9.101e-11, 1.334e-10, 1.951e-10
2.853e-10, 3.94e-10, 4.771e-10, 5.771e-10, 6.675e-10, 7.665e-10,
00274
00275
00276
                    8.504e-10, 8.924e-10, 9.363e-10, 8.923e-10, 8.411e-10, 7.646e-10,
                     6.525e-10, 5.576e-10, 4.398e-10, 3.403e-10, 2.612e-10, 1.915e-10, 1.407e-10, 1.028e-10, 7.455e-11, 5.42e-11, 3.708e-11, 2.438e-11, 1.618e-11, 1.075e-11, 7.17e-12, 4.784e-12, 3.205e-12, 2.147e-12,
00277
00278
                     1.44e-12, 9.654e-13, 6.469e-13, 4.332e-13, 2.891e-13, 1.926e-13,
                     1.274e-13, 8.422e-14, 5.547e-14, 3.636e-14, 2.368e-14, 1.536e-14,
00281
00282
                     9.937e-15, 6.39e-15, 4.101e-15, 2.61e-15, 1.659e-15, 1.052e-15,
                     6.638e-16, 4.172e-16, 2.61e-16, 1.63e-16, 1.013e-16, 6.275e-17, 3.879e-17, 2.383e-17, 1.461e-17, 8.918e-18, 5.43e-18, 3.301e-18,
00283
00284
                     1.997e-18, 1.203e-18, 7.216e-19, 4.311e-19, 2.564e-19, 1.519e-19,
00285
                     8.911e-20, 5.203e-20, 3.026e-20, 1.748e-20, 9.99e-21, 5.673e-21,
00286
                     3.215e-21, 1.799e-21, 1.006e-21, 5.628e-22, 3.146e-22, 1.766e-22,
00287
00288
                     9.94e-23, 5.614e-23, 3.206e-23, 1.841e-23, 1.071e-23, 6.366e-24,
                     3.776e-24, 2.238e-24, 1.326e-24, 8.253e-25, 5.201e-25, 3.279e-25, 2.108e-25, 1.395e-25, 9.326e-26, 6.299e-26, 4.365e-26, 3.104e-26, 2.219e-26, 1.621e-26, 1.185e-26, 8.92e-27, 6.804e-27, 5.191e-27,
00289
00290
00291
00292
                     4.041e-27
00293
00294
                static double co[121] = {
1.907e-07, 1.553e-07, 1.362e-07, 1.216e-07, 1.114e-07, 1.036e-07,
9.737e-08, 9.152e-08, 8.559e-08, 7.966e-08, 7.277e-08, 6.615e-08,
00295
00296
00297
```

5.4 jurassic.c 97

```
5.884e-08, 5.22e-08, 4.699e-08, 4.284e-08, 3.776e-08, 3.274e-08,
            2.845e-08, 2.479e-08, 2.246e-08, 2.054e-08, 1.991e-08, 1.951e-08,
00299
00300
            1.94e-08, 2.009e-08, 2.1e-08, 2.201e-08, 2.322e-08, 2.45e-08,
            2.602e-08, 2.73e-08, 2.867e-08, 2.998e-08, 3.135e-08, 3.255e-08,
00301
00302
            3.352e-08, 3.426e-08, 3.484e-08, 3.53e-08, 3.593e-08, 3.671e-08, 3.759e-08, 3.945e-08, 4.192e-08, 4.49e-08, 5.03e-08, 5.703e-08,
00303
            6.538e-08, 7.878e-08, 9.644e-08, 1.196e-07, 1.498e-07, 1.904e-07,
            2.422e-07, 3.055e-07, 3.804e-07, 4.747e-07, 5.899e-07,
00305
00306
            8.91e-07, 1.071e-06, 1.296e-06, 1.546e-06, 1.823e-06, 2.135e-06,
00307
            2.44e-06, 2.714e-06, 2.967e-06, 3.189e-06, 3.391e-06, 3.58e-06,
            3.773e-06, 4.022e-06, 4.346e-06, 4.749e-06, 5.199e-06, 5.668e-06, 6.157e-06, 6.688e-06, 7.254e-06, 7.867e-06, 8.539e-06, 9.26e-06,
00308
00309
00310
            1.009e-05, 1.119e-05, 1.228e-05, 1.365e-05, 1.506e-05, 1.641e-05,
            1.784e-05, 1.952e-05, 2.132e-05, 2.323e-05, 2.531e-05, 2.754e-05,
00311
00312
            3.047e-05, 3.459e-05, 3.922e-05, 4.439e-05, 4.825e-05, 5.077e-05,
            5.34e-05, 5.618e-05, 5.909e-05, 6.207e-05, 6.519e-05, 6.845e-05, 6.819e-05, 6.726e-05, 6.622e-05, 6.512e-05, 6.671e-05, 6.862e-05, 7.048e-05, 7.264e-05, 7.3e-05, 7.3e-05, 7.3e-05, 7.3e-05, 7.3e-05
00313
00314
00315
00317
00318
         static double cof2[121] = +
00319
            7.5e-14, 1.055e-13, 1.485e-13, 2.111e-13, 3.001e-13, 4.333e-13,
            6.269e-13, 9.221e-13, 1.364e-12, 2.046e-12, 3.093e-12, 4.703e-12, 7.225e-12, 1.113e-11, 1.66e-11, 2.088e-11, 2.626e-11, 3.433e-11, 4.549e-11, 5.886e-11, 7.21e-11, 8.824e-11, 1.015e-10, 1.155e-10,
00320
00321
00322
            1.288e-10, 1.388e-10, 1.497e-10, 1.554e-10, 1.606e-10, 1.639e-10,
00324
            1.64e-10, 1.64e-10, 1.596e-10, 1.542e-10, 1.482e-10, 1.382e-10,
00325
            1.289e-10, 1.198e-10, 1.109e-10, 1.026e-10, 9.484e-11, 8.75e-11,
           8.086e-11, 7.49e-11, 6.948e-11, 6.46e-11, 5.961e-11, 5.505e-11, 5.085e-11, 4.586e-11, 4.1e-11, 3.665e-11, 3.235e-11, 2.842e-11, 2.491e-11, 2.11e-11, 1.769e-11, 1.479e-11, 1.197e-11, 9.631e-12, 7.74e-12, 6.201e-12, 4.963e-12, 3.956e-12, 3.151e-12, 2.507e-12,
00326
00327
00328
00329
            1.99e-12, 1.576e-12, 1.245e-12, 9.83e-13, 7.742e-13, 6.088e-13,
00330
00331
            4.782e-13, 3.745e-13, 2.929e-13, 2.286e-13, 1.782e-13, 1.388e-13,
00332
            1.079 e^{-13},\ 8.362 e^{-14},\ 6.471 e^{-14},\ 4.996 e^{-14},\ 3.85 e^{-14},\ 2.96 e^{-14},
            2.265e-14, 1.729e-14, 1.317e-14, 9.998e-15, 7.549e-15, 5.683e-15,
00333
            4.273e-15, 3.193e-15, 2.385e-15, 1.782e-15, 1.331e-15, 9.957e-16,
00334
            7.461e-16, 5.601e-16, 4.228e-16, 3.201e-16, 2.438e-16, 1.878e-16,
00336
            1.445e-16, 1.111e-16, 8.544e-17, 6.734e-17, 5.341e-17, 4.237e-17,
            3.394e-17, 2.759e-17, 2.254e-17, 1.851e-17, 1.54e-17, 1.297e-17,
00337
00338
            1.096e-17, 9.365e-18, 8e-18, 6.938e-18, 6.056e-18, 5.287e-18,
00339
            4.662e-18
00340
00341
00342
         static double f11[121] =
00343
            2.65e-10, 2.65e-10, 2.65e-10, 2.65e-10, 2.65e-10, 2.65e-10,
00344
            2.65e-10, 2.65e-10, 2.65e-10, 2.65e-10, 2.65e-10, 2.635e-10, 2.536e-10,
00345
            2.44 e^{-10},\ 2.348 e^{-10},\ 2.258 e^{-10},\ 2.153 e^{-10},\ 2.046 e^{-10},\ 1.929 e^{-10},
            1.782e-10, 1.648e-10, 1.463e-10, 1.291e-10, 1.1e-10, 8.874e-11,
00346
            7.165e-11, 5.201e-11, 3.744e-11, 2.577e-11, 1.64e-11, 1.048e-11,
00347
            5.993e-12, 3.345e-12, 1.839e-12, 9.264e-13, 4.688e-13, 2.329e-13,
            1.129e-13, 5.505e-14, 2.825e-14, 1.492e-14, 7.997e-15, 5.384e-15,
00349
00350
            3.988e-15, 2.955e-15, 2.196e-15, 1.632e-15, 1.214e-15, 9.025e-16,
           6.708e-16, 4.984e-16, 3.693e-16, 2.733e-16, 2.013e-16, 1.481e-16, 1.087e-16, 7.945e-17, 5.782e-17, 4.195e-17, 3.038e-17, 2.19e-17,
00351
00352
            1.577e-17, 1.128e-17, 8.063e-18, 5.753e-18, 4.09e-18, 2.899e-18,
00353
            2.048e-18, 1.444e-18, 1.015e-18, 7.12e-19, 4.985e-19, 3.474e-19,
            2.417e-19, 1.677e-19, 1.161e-19, 8.029e-20, 5.533e-20, 3.799e-20,
00355
           2.602e-20, 1.776e-20, 1.209e-20, 8.202e-21, 5.522e-21, 3.707e-21, 2.48e-21, 1.652e-21, 1.091e-21, 7.174e-22, 4.709e-22, 3.063e-22,
00356
00357
00358
            1.991e-22, 1.294e-22, 8.412e-23, 5.483e-23, 3.581e-23, 2.345e-23,
00359
            1.548e-23, 1.027e-23, 6.869e-24, 4.673e-24, 3.173e-24, 2.153e-24,
00360
            1.461e-24, 1.028e-24, 7.302e-25, 5.188e-25, 3.739e-25, 2.753e-25,
            2.043e-25, 1.528e-25, 1.164e-25, 9.041e-26, 7.051e-26, 5.587e-26,
00361
00362
            4.428e-26, 3.588e-26, 2.936e-26, 2.402e-26, 1.995e-26
00363
00364
         static double f12[121] = {
00365
00366
           5.45e-10, 5.45e-10, 5.45e-10, 5.45e-10, 5.45e-10, 5.45e-10, 5.45e-10,
            5.45e-10, 5.45e-10, 5.45e-10, 5.45e-10, 5.45e-10, 5.429e-10, 5.291e-10,
            5.155e-10, 5.022e-10, 4.893e-10, 4.772e-10, 4.655e-10, 4.497e-10,
00368
00369
            4.249e-10, 4.015e-10, 3.632e-10, 3.261e-10, 2.858e-10, 2.408e-10,
           2.03e-10, 1.685e-10, 1.4e-10, 1.163e-10, 9.65e-11, 8.02e-11, 6.705e-11, 5.624e-11, 4.764e-11, 4.249e-11, 3.792e-11, 3.315e-11, 2.819e-11,
00370
00371
            2.4e-11, 1.999e-11, 1.64e-11, 1.352e-11, 1.14e-11, 9.714e-12, 8.28e-12, 7.176e-12, 6.251e-12, 5.446e-12, 4.72e-12, 4.081e-1
00372
00374
            3.528e-12, 3.08e-12, 2.699e-12, 2.359e-12, 2.111e-12, 1.901e-12,
00375
            1.709e-12, 1.534e-12, 1.376e-12, 1.233e-12, 1.103e-12, 9.869e-13,
            8.808e-13, 7.859e-13, 7.008e-13, 6.241e-13, 5.553e-13, 4.935e-13,
00376
            4.383e-13, 3.889e-13, 3.447e-13, 3.054e-13, 2.702e-13, 2.389e-13,
00377
            2.11e-13, 1.862e-13, 1.643e-13, 1.448e-13, 1.274e-13, 1.121e-13,
00378
            9.844e-14, 8.638e-14, 7.572e-14, 6.62e-14, 5.782e-14, 5.045e-14,
            4.394e-14, 3.817e-14, 3.311e-14, 2.87e-14, 2.48e-14, 2.142e-14,
00380
00381
            1.851e-14, 1.599e-14, 1.383e-14, 1.196e-14, 1.036e-14, 9e-15,
00382
            7.828e-15, 6.829e-15, 5.992e-15, 5.254e-15, 4.606e-15, 4.037e-15,
            3.583e-15, 3.19e-15, 2.841e-15, 2.542e-15, 2.291e-15, 2.07e-15, 1.875e-15, 1.71e-15, 1.57e-15, 1.442e-15, 1.333e-15, 1.232e-15,
00383
00384
```

```
00385
               1.147e-15, 1.071e-15, 1.001e-15, 9.396e-16
00386
00387
00388
            static double f14[121] = {
               9e-11, 8.73e-11, 8.46e-11,
00389
00390
                8.19e-11, 7.92e-11, 7.74e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11,
                7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11,
00392
00393
                7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11,
                                                                                                7.65e-11,
                                                                                                                 7.65e-11,
00394
                7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11,
                7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11,
00395
                7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11,
00396
00397
                7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11,
                                                                                                                 7.65e-11.
                7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11,
00398
00399
                7.65e-11, 7.65e-11, 7.65e-11,
                                                                7.65e-11, 7.65e-11, 7.65e-11,
                                                                                                                 7.65e-11,
                7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-
                                                                                                                 7.65e-11,
00400
00401
                                                                                                                 7.65e-11.
00402
                7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11,
                7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11,
                7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11,
00404
00405
                7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11
00406
00407
00408
            static double f22[121] = {
00409
               1.4e-10, 1.4e-10, 1.4e-10, 1.4e-10, 1.4e-10, 1.4e-10,
                1.4e-10, 1.4e-10, 1.4e-10, 1.372e-10, 1.317e-10, 1.235e-10, 1.153e-10,
00411
                1.075e-10, 1.002e-10, 9.332e-11, 8.738e-11, 8.194e-11, 7.7e-11,
00412
               7.165e-11, 6.753e-11, 6.341e-11, 5.971e-11, 5.6e-11, 5.229e-11,
               4.859e-11, 4.488e-11, 4.118e-11, 3.83e-11, 3.568e-11, 3.308e-11, 3.047e-11, 2.82e-11, 2.594e-11, 2.409e-11, 2.237e-11, 2.065e-11, 1.894e-11, 1.771e-11, 1.647e-11, 1.532e-11, 1.416e-11, 1.332e-11,
00413
00414
00415
00416
                1.246e-11, 1.161e-11, 1.087e-11, 1.017e-11, 9.471e-12, 8.853e-12,
                8.235e-12, 7.741e-12, 7.247e-12, 6.836e-12, 6.506e-12,
00417
                                                                                                        6.176e-12,
00418
                5.913e-12, 5.65e-12, 5.419e-12, 5.221e-12, 5.024e-12, 4.859e-12,
00419
                4.694e-12, 4.546e-12, 4.414e-12, 4.282e-12, 4.15e-12, 4.019e-12,
                3.903e-12, 3.805e-12, 3.706e-12, 3.607e-12, 3.508e-12, 3.41e-12, 3.31e-12, 3.212e-12, 3.129e-12, 3.047e-12, 2.964e-12, 2.882e-12, 2.8e-12, 2.734e-12, 2.668e-12, 2.602e-12, 2.537e-12, 2.471e-12,
00420
00421
00423
                2.421e-12, 2.372e-12, 2.322e-12, 2.273e-12, 2.224e-12, 2.182e-12,
                2.141e-12, 2.1e-12, 2.059e-12, 2.018e-12, 1.977e-12, 1.935e-12,
00424
00425
                1.894e-12, 1.853e-12, 1.812e-12, 1.77e-12, 1.73e-12, 1.688e-12,
                1.647e-12, 1.606e-12, 1.565e-12, 1.524e-12, 1.483e-12, 1.441e-12,
00426
00427
                1.4e-12, 1.359e-12, 1.317e-12, 1.276e-12, 1.235e-12, 1.194e-12,
                1.153e-12, 1.112e-12, 1.071e-12, 1.029e-12, 9.883e-13
00428
00429
00430
00431
            static double h2o[121] = {
              0.01166, 0.008269, 0.005742, 0.003845, 0.00277, 0.001897, 0.001272, 0.000827, 0.000539, 0.0003469, 0.0001579, 3.134e-05, 1.341e-05,
00432
00433
                6.764e-06, 4.498e-06, 3.703e-06, 3.724e-06, 3.899e-06, 4.002e-06,
00434
                4.122e-06, 4.277e-06, 4.438e-06, 4.558e-06, 4.673e-06, 4.763e-06,
                4.809e-06, 4.856e-06, 4.936e-06, 5.021e-06, 5.114e-06, 5.222e-06,
00436
00437
                5.331e-06, 5.414e-06, 5.488e-06, 5.563e-06, 5.633e-06, 5.704e-06,
00438
                5.767e-06, 5.819e-06, 5.872e-06, 5.914e-06, 5.949e-06, 5.984e-06,
                6.015e-06, 6.044e-06, 6.073e-06, 6.104e-06, 6.136e-06, 6.167e-06,
00439
                6.189e-06, 6.208e-06, 6.226e-06, 6.212e-06, 6.185e-06, 6.158e-06,
00440
                6.114e-06, 6.066e-06, 6.018e-06, 5.877e-06, 5.728e-06, 5.582e-06,
                5.437e-06, 5.296e-06, 5.156e-06, 5.02e-06, 4.886e-06, 4.754e-06,
00442
00443
                4.625e-06, 4.498e-06, 4.374e-06, 4.242e-06, 4.096e-06, 3.955e-06,
               3.817e-06, 3.683e-06, 3.491e-06, 3.204e-06, 2.94e-06, 2.696e-06, 2.47e-06, 2.252e-06, 2.019e-06, 1.808e-06, 1.618e-06, 1.445e-06, 1.285e-06, 1.105e-06, 9.489e-07, 8.121e-07, 6.938e-07, 5.924e-07, 5.04e-07, 4.288e-07, 3.648e-07, 3.103e-07, 2.642e-07, 2.252e-07,
00444
00445
00446
00447
                1.921e-07, 1.643e-07, 1.408e-07, 1.211e-07, 1.048e-07, 9.063e-08,
00448
00449
                7.835e-08, 6.774e-08, 5.936e-08, 5.221e-08, 4.592e-08, 4.061e-08,
00450
                3.62e-08, 3.236e-08, 2.902e-08, 2.62e-08, 2.383e-08, 2.171e-08,
00451
               1.989e-08, 1.823e-08, 1.684e-08, 1.562e-08, 1.449e-08, 1.351e-08
00452
00453
            static double h2o2[121] =
              1.779e-10, 7.938e-10, 8.953e-10, 8.032e-10, 6.564e-10, 5.159e-10,
00455
00456
                4.003e-10, 3.026e-10, 2.222e-10, 1.58e-10, 1.044e-10, 6.605e-11,
00457
                3.413e-11, 1.453e-11, 1.062e-11, 1.009e-11, 9.597e-12, 1.175e-11,
               1.572e-11, 2.091e-11, 2.746e-11, 3.603e-11, 4.791e-11, 6.387e-11, 8.239e-11, 1.007e-10, 1.23e-10, 1.363e-10, 1.489e-10, 1.585e-10, 1.608e-10, 1.632e-10, 1.576e-10, 1.502e-10, 1.423e-10, 1.302e-10,
00458
00459
00460
                1.192e-10, 1.085e-10, 9.795e-11, 8.854e-11, 8.057e-11, 7.36e-11,
00461
00462
                6.736e-11, 6.362e-11, 6.087e-11, 5.825e-11, 5.623e-11,
                                                                                                        5.443e-11,
00463
                5.27 e-11, \ 5.098 e-11, \ 4.931 e-11, \ 4.769 e-11, \ 4.611 e-11, \ 4.458 e-11,
                4.308e-11, 4.102e-11, 3.887e-11, 3.682e-11, 3.521e-11, 3.369e-11,
00464
                3.224e-11, 3.082e-11, 2.946e-11, 2.814e-11, 2.687e-11, 2.566e-11, 2.449e-11, 2.336e-11, 2.227e-11, 2.123e-11, 2.023e-11, 1.927e-11,
00465
                1.835e-11, 1.746e-11, 1.661e-11, 1.58e-11, 1.502e-11, 1.428e-11,
00467
00468
                1.357e-11, 1.289e-11, 1.224e-11, 1.161e-11, 1.102e-11, 1.045e-11,
00469
                9.895e-12, 9.369e-12, 8.866e-12, 8.386e-12, 7.922e-12, 7.479e-12,
                7.06e-12, 6.656e-12, 6.274e-12, 5.914e-12, 5.575e-12, 5.257e-12, 4.959e-12, 4.679e-12, 4.42e-12, 4.178e-12, 3.954e-12, 3.75e-12,
00470
00471
```

5.4 jurassic.c 99

```
3.557e-12, 3.372e-12, 3.198e-12, 3.047e-12, 2.908e-12, 2.775e-12,
            2.653e-12, 2.544e-12, 2.442e-12, 2.346e-12, 2.26e-12, 2.183e-12,
00473
00474
            2.11e-12, 2.044e-12, 1.98e-12, 1.924e-12, 1.871e-12, 1.821e-12,
00475
           1.775e-12
00476
00477
00478
         static double hcn[121] = {
            5.5e-10, 5.5e-10, 5.5e-10, 5.5e-10, 5.5e-10, 5.5e-10,
00479
00480
            5.5e-10, 5.5e-10, 5.5e-10, 5.5e-10, 5.498e-10, 5.495e-10, 5.493e-10,
00481
            5.49e-10, 5.488e-10, 4.717e-10, 3.946e-10, 3.174e-10, 2.4e-10,
            1.626e-10, 1.619e-10, 1.612e-10, 1.602e-10, 1.593e-10, 1.582e-10,
00482
            1.572e-10, 1.56e-10, 1.549e-10, 1.539e-10, 1.53e-10, 1.519e-10,
00483
00484
            1.506e-10, 1.487e-10, 1.467e-10, 1.449e-10, 1.43e-10, 1.413e-10,
            1.397e-10, 1.382e-10, 1.368e-10, 1.354e-10, 1.337e-10, 1.315e-10,
00485
00486
            1.292e-10, 1.267e-10, 1.241e-10, 1.215e-10, 1.19e-10, 1.165e-10,
00487
            1.141e-10, 1.118e-10, 1.096e-10, 1.072e-10, 1.047e-10, 1.021e-10,
00488
            9.968e-11, 9.739e-11, 9.539e-11, 9.339e-11, 9.135e-11, 8.898e-11,
00489
            8.664e-11, 8.439e-11, 8.249e-11, 8.075e-11, 7.904e-11, 7.735e-11,
            7.565e-11, 7.399e-11, 7.245e-11, 7.109e-11, 6.982e-11, 6.863e-11,
00491
            6.755e-11, 6.657e-11, 6.587e-11, 6.527e-11, 6.476e-11, 6.428e-11,
            6.382e-11, 6.343e-11, 6.307e-11, 6.272e-11, 6.238e-11, 6.205e-11,
00492
00493
            6.17e-11, 6.137e-11, 6.102e-11, 6.072e-11, 6.046e-11, 6.03e-11,
            6.018e-11, 6.01e-11, 6.001e-11, 5.992e-11, 5.984e-11, 5.975e-11,
00494
           5.967e-11, 5.958e-11, 5.95e-11, 5.941e-11, 5.933e-11, 5.925e-11, 5.916e-11, 5.908e-11, 5.899e-11, 5.891e-11, 5.883e-11, 5.874e-11, 5.866e-11, 5.858e-11, 5.85e-11, 5.841e-11, 5.833e-11, 5.825e-11, 5.817e-11, 5.808e-11, 5.8e-11, 5.792e-11, 5.784e-11
00495
00496
00497
00498
00/99
00500
00501
         static double hno3[121] = {
           1.809e-10, 7.234e-10, 5.899e-10, 4.342e-10, 3.277e-10, 2.661e-10, 2.35e-10, 2.267e-10, 2.389e-10, 2.651e-10, 3.255e-10, 4.099e-10,
00502
00503
           5.42e-10, 6.978e-10, 8.807e-10, 1.112e-09, 1.405e-09, 2.04e-09, 3.111e-09, 4.5e-09, 5.762e-09, 7.37e-09, 7.852e-09, 8.109e-09,
00504
00505
           8.067e-09, 7.554e-09, 7.076e-09, 6.268e-09, 5.524e-09, 4.749e-09, 3.909e-09, 3.223e-09, 2.517e-09, 1.942e-09, 1.493e-09, 1.122e-09,
00506
00507
            8.449e-10, 6.361e-10, 4.787e-10, 3.611e-10, 2.804e-10, 2.215e-10,
00508
            1.758e-10, 1.441e-10, 1.197e-10, 9.953e-11, 8.505e-11, 7.334e-11,
00510
            6.325e-11, 5.625e-11, 5.058e-11, 4.548e-11, 4.122e-11, 3.748e-11,
00511
            3.402e-11, 3.088e-11, 2.8e-11, 2.536e-11, 2.293e-11, 2.072e-11,
00512
           1.871e-11, 1.687e-11, 1.52e-11, 1.368e-11, 1.23e-11, 1.105e-11,
            9.922e-12, 8.898e-12, 7.972e-12, 7.139e-12, 6.385e-12, 5.708e-12,
00513
00514
            5.099 e-12,\ 4.549 e-12,\ 4.056 e-12,\ 3.613 e-12,\ 3.216 e-12,\ 2.862 e-12,
            2.544e-12, 2.259e-12, 2.004e-12, 1.776e-12, 1.572e-12, 1.391e-12,
            1.227e-12, 1.082e-12, 9.528e-13, 8.379e-13, 7.349e-13, 6.436e-13,
00516
00517
            5.634e-13, 4.917e-13, 4.291e-13, 3.745e-13, 3.267e-13, 2.854e-13,
00518
           2.494e-13, 2.181e-13, 1.913e-13, 1.68e-13, 1.479e-13, 1.31e-13,
           1.159e-13, 1.025e-13, 9.067e-14, 8.113e-14, 7.281e-14, 6.535e-14, 5.892e-14, 5.348e-14, 4.867e-14, 4.439e-14, 4.073e-14, 3.76e-14, 3.476e-14, 3.229e-14, 3e-14, 2.807e-14, 2.635e-14, 2.473e-14,
00519
00520
00521
            2.332e-14
00523
00524
         static double hno4[121] = { 6.118e-12, 3.594e-12, 2.807e-12, 3.04e-12, 4.458e-12, 7.986e-12,
00525
00526
            1.509e-11, 2.661e-11, 3.738e-11, 4.652e-11, 4.429e-11, 3.992e-11,
00527
            3.347e-11, 3.005e-11, 3.173e-11, 4.055e-11, 5.812e-11, 8.489e-11,
            1.19e-10, 1.482e-10, 1.766e-10, 2.103e-10, 2.35e-10, 2.598e-10,
00529
00530
            2.801e-10, 2.899e-10, 3e-10, 2.817e-10, 2.617e-10, 2.332e-10,
00531
            1.933e-10, 1.605e-10, 1.232e-10, 9.285e-11, 6.941e-11, 4.951e-11,
           3.539e-11, 2.402e-11, 1.522e-11, 9.676e-12, 6.056e-12, 3.745e-12, 2.34e-12, 1.463e-12, 9.186e-13, 5.769e-13, 3.322e-13, 1.853e-13,
00532
00533
            1.035e-13, 7.173e-14, 5.382e-14, 4.036e-14, 3.401e-14, 2.997e-14,
            2.635e-14, 2.316e-14, 2.034e-14, 1.783e-14, 1.56e-14, 1.363e-14, 1.19e-14, 1.037e-14, 9.032e-15, 7.846e-15, 6.813e-15, 5.912e-15,
00535
00536
00537
            5.121e-15, 4.431e-15, 3.829e-15, 3.306e-15, 2.851e-15, 2.456e-15,
00538
            2.114e-15, 1.816e-15, 1.559e-15, 1.337e-15, 1.146e-15, 9.811e-16,
            8.389e-16, 7.162e-16, 6.109e-16, 5.203e-16, 4.425e-16, 3.76e-16,
00539
00540
           3.184e-16, 2.692e-16, 2.274e-16, 1.917e-16, 1.61e-16, 1.35e-16,
            1.131e-16, 9.437e-17, 7.874e-17, 6.57e-17, 5.481e-17, 4.579e-17,
00542
            3.828e-17, 3.204e-17, 2.691e-17, 2.264e-17, 1.912e-17, 1.626e-17,
00543
            1.382e-17, 1.174e-17, 9.972e-18, 8.603e-18, 7.45e-18, 6.453e-18,
           5.623e-18, 4.944e-18, 4.361e-18, 3.859e-18, 3.443e-18, 3.096e-18, 2.788e-18, 2.528e-18, 2.293e-18, 2.099e-18, 1.929e-18, 1.773e-18,
00544
00545
00546
            1.64e-18
00547
00548
00549
         static double hocl[121] = +
           1.056e-12, 1.194e-12, 1.35e-12, 1.531e-12, 1.737e-12, 1.982e-12,
00550
           2.263e-12, 2.599e-12, 2.991e-12, 3.459e-12, 4.012e-12, 4.662e-12, 5.438e-12, 6.35e-12, 7.425e-12, 8.686e-12, 1.016e-11, 1.188e-11,
00551
00552
            1.389e-11, 1.659e-11, 2.087e-11, 2.621e-11, 3.265e-11, 4.064e-11,
            4.859e-11, 5.441e-11, 6.09e-11, 6.373e-11, 6.611e-11, 6.94e-11, 7.44e-11, 7.97e-11, 8.775e-11, 9.722e-11, 1.064e-10, 1.089e-10,
00554
00555
00556
            1.114e-10, 1.106e-10, 1.053e-10, 1.004e-10, 9.006e-11, 7.778e-11,
            6.739e-11, 5.636e-11, 4.655e-11, 3.845e-11, 3.042e-11, 2.368e-11, 1.845e-11, 1.442e-11, 1.127e-11, 8.814e-12, 6.544e-12, 4.763e-12,
00557
00558
```

```
3.449e-12, 2.612e-12, 1.999e-12, 1.526e-12, 1.16e-12, 8.793e-13,
                   6.655e-13, 5.017e-13, 3.778e-13, 2.829e-13, 2.117e-13, 1.582e-13, 1.178e-13, 8.755e-14, 6.486e-14, 4.799e-14, 3.54e-14, 2.606e-14,
00560
00561
00562
                    1.916e-14, 1.403e-14, 1.026e-14, 7.48e-15, 5.446e-15, 3.961e-15,
                    2.872e-15, 2.076e-15, 1.498e-15, 1.077e-15, 7.726e-16, 5.528e-16,
00563
                    3.929e-16, 2.785e-16, 1.969e-16, 1.386e-16, 9.69e-17, 6.747e-17,
00564
                    4.692e-17, 3.236e-17, 2.232e-17, 1.539e-17, 1.061e-17, 7.332e-18,
                    5.076e-18, 3.522e-18, 2.461e-18, 1.726e-18, 1.22e-18, 8.75e-19,
00566
                    6.264e-19, 4.482e-19, 3.207e-19, 2.368e-19, 1.762e-19, 1.312e-19, 9.891e-20, 7.595e-20, 5.87e-20, 4.567e-20, 3.612e-20, 2.904e-20, 2.343e-20, 1.917e-20, 1.568e-20, 1.308e-20, 1.1e-20, 9.25e-21,
00567
00568
00569
00570
                    7.881e-21
00571
00572
00573
                static double n2o[121] = {
                  3.17e-07, 3.03e-07, 2.984e-07, 2.938e-07, 2.892e-07, 2.847e-07, 2.779e-07, 2.705e-07,
00574
00575
00576
                    2.631e-07, 2.557e-07, 2.484e-07, 2.345e-07, 2.201e-07, 2.01e-07,
00578
                    1.754e-07, 1.532e-07, 1.329e-07, 1.154e-07, 1.003e-07, 8.735e-08,
00579
                    7.617e-08, 6.512e-08, 5.547e-08, 4.709e-08, 3.915e-08, 3.259e-08,
00580
                    2.738e-08, 2.327e-08, 1.98e-08, 1.711e-08, 1.493e-08, 1.306e-08,
00581
                    1.165e-08, 1.049e-08, 9.439e-09, 8.375e-09, 7.391e-09, 6.525e-09,
                    5.759e-09, 5.083e-09, 4.485e-09, 3.953e-09, 3.601e-09, 3.27e-09, 2.975e-09, 2.757e-09, 2.556e-09, 2.37e-09, 2.195e-09, 2.032e-09,
00582
00583
                    1.912e-09, 1.79e-09, 1.679e-09, 1.572e-09, 1.482e-09, 1.402e-09,
                    1.326e-09, 1.254e-09, 1.187e-09, 1.127e-09, 1.071e-09, 1.02e-09,
00585
00586
                    9.673e-10, 9.193e-10, 8.752e-10, 8.379e-10, 8.017e-10, 7.66e-10,
00587
                    7.319e-10, 7.004e-10, 6.721e-10, 6.459e-10, 6.199e-10, 5.942e-10,
00588
                    5.703e-10, 5.488e-10, 5.283e-10, 5.082e-10, 4.877e-10, 4.696e-10,
00589
                    4.52e-10, 4.355e-10, 4.198e-10, 4.039e-10, 3.888e-10, 3.754e-10,
00590
                    3.624e-10, 3.499e-10, 3.381e-10, 3.267e-10, 3.163e-10, 3.058e-10,
00591
                    2.959e-10, 2.864e-10, 2.77e-10, 2.686e-10, 2.604e-10, 2.534e-10,
00592
                    2.462e-10, 2.386e-10, 2.318e-10, 2.247e-10, 2.189e-10, 2.133e-10,
00593
                   2.071e-10, 2.014e-10, 1.955e-10, 1.908e-10, 1.86e-10, 1.817e-10
00594
00595
               static double n2o5[121] = {
00597
                  1.231e-11, 3.035e-12, 1.702e-12, 9.877e-13, 8.081e-13, 9.039e-13,
00598
                    1.169e-12, 1.474e-12, 1.651e-12, 1.795e-12, 1.998e-12, 2.543e-12,
00599
                    4.398e-12, 7.698e-12, 1.28e-11, 2.131e-11, 3.548e-11, 5.894e-11,
00600
                    7.645 e^{-11}, \ 1.089 e^{-10}, \ 1.391 e^{-10}, \ 1.886 e^{-10}, \ 2.386 e^{-10}, \ 2.986 e^{-10},
00601
                    3.487e-10, 3.994e-10, 4.5e-10, 4.6e-10, 4.591e-10, 4.1e-10, 3.488e-10,
                    2.846e-10, 2.287e-10, 1.696e-10, 1.011e-10, 6.428e-11, 4.324e-11,
00602
                    2.225e-11, 6.214e-12, 3.608e-12, 8.793e-13, 4.491e-13, 1.04e-13,
00604
                    6.1e-14, 3.436e-14, 6.671e-15, 1.171e-15, 5.848e-16, 1.212e-16,
00605
                    le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16,
00606
                    1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16,
00607
                    le-16, le
00608
                    le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16,
                    1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16,
                    le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16,
00610
00611
                    1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16,
00612
                   1e-16, 1e-16
00613
00614
               static double nh3[121] = {
                   1e-10, 1e-10, 1e-10, 1e-10, 1e-10, 1e-10, 1e-10, 1e-10, 1e-10, 1e-10,
00616
00617
                    1e-10, 1e-10, 1e-10, 1e-10, 9.444e-11, 8.488e-11, 7.241e-11, 5.785e-11,
00618
                    4.178e-11, 3.018e-11, 2.18e-11, 1.574e-11, 1.137e-11, 8.211e-12,
00619
                   5.973e-12, 4.327e-12, 3.118e-12, 2.234e-12, 1.573e-12, 1.04e-12,
00620
                    6.762e-13, 4.202e-13, 2.406e-13, 1.335e-13, 6.938e-14, 3.105e-14,
00621
                    1.609e-14, 1.033e-14, 6.432e-15, 4.031e-15, 2.555e-15, 1.656e-15,
                    1.115e-15, 7.904e-16, 5.63e-16, 4.048e-16, 2.876e-16, 2.004e-16, 1.356e-16, 9.237e-17, 6.235e-17, 4.223e-17, 3.009e-17, 2.328e-17,
00622
00623
00624
                   2.002e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17,
00625
                    1.914e-17,\ 1.914e-17,\ 1.914e-17,\ 1.914e-17,\ 1.914e-17,\ 1.914e-17,
00626
                    1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17,
00627
                    1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17,
                    1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17,
                    1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17,
00629
00630
                    1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17,
                   1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17,
00631
00632
                    1.914e-17, 
00633
00634
00635
                    1.914e-17
00636
00637
00638
               static double no[1211 = {
                  2.586e-10, 4.143e-11, 1.566e-11, 9.591e-12, 8.088e-12, 8.462e-12,
00639
                    1.013e-11, 1.328e-11, 1.855e-11, 2.678e-11, 3.926e-11, 5.464e-11,
                    7.012e-11, 8.912e-11, 1.127e-10, 1.347e-10, 1.498e-10, 1.544e-10,
00641
00642
                   1.602e-10, 1.824e-10, 2.078e-10, 2.366e-10, 2.691e-10, 5.141e-10,
                   8.259e-10, 1.254e-09, 1.849e-09, 2.473e-09, 3.294e-09, 4.16e-09, 5.095e-09, 6.11e-09, 6.93e-09, 7.888e-09, 8.903e-09, 9.713e-09, 1.052e-08, 1.115e-08, 1.173e-08, 1.21e-08, 1.228e-08, 1.239e-08,
00643
00644
00645
```

5.4 jurassic.c 101

```
1.231e-08, 1.213e-08, 1.192e-08, 1.138e-08, 1.085e-08, 1.008e-08,
                9.224e-09, 8.389e-09, 7.262e-09, 6.278e-09, 5.335e-09, 4.388e-09,
00647
00648
                3.589e-09, 2.761e-09, 2.129e-09, 1.633e-09, 1.243e-09, 9.681e-10,
00649
                8.355e-10, 7.665e-10, 7.442e-10, 8.584e-10, 9.732e-10, 1.063e-09,
00650
                1.163e-09, 1.286e-09, 1.472e-09, 1.707e-09, 2.032e-09, 2.474e-09,
                2.977e-09, 3.506e-09, 4.102e-09, 5.013e-09, 6.493e-09, 8.414e-09,
00651
                1.077e-08, 1.367e-08, 1.777e-08, 2.625e-08, 3.926e-08, 5.545e-08,
                7.195e-08, 9.464e-08, 1.404e-07, 2.183e-07, 3.329e-07, 4.535e-07,
00653
               6.158e-07, 8.187e-07, 1.075e-06, 1.422e-06, 1.979e-06, 2.71e-06, 3.58e-06, 4.573e-06, 5.951e-06, 7.999e-06, 1.072e-05, 1.372e-05, 1.697e-05, 2.112e-05, 2.643e-05, 3.288e-05, 3.994e-05, 4.794e-05, 5.606e-05, 6.383e-05, 7.286e-05, 8.156e-05, 8.883e-05, 9.469e-05,
00654
00655
00656
00657
                9.848e-05, 0.0001023, 0.0001066, 0.0001115, 0.0001145, 0.0001142,
00658
00659
               0.0001133
00660
00661
00662
            static double no2[121] = {
                3.036e-09, 2.945e-10, 9.982e-11, 5.069e-11, 3.485e-11, 2.982e-11,
00663
                2.947e-11, 3.164e-11, 3.714e-11, 4.586e-11, 6.164e-11, 8.041e-11, 9.982e-11, 1.283e-10, 1.73e-10, 2.56e-10, 3.909e-10, 5.959e-10,
                9.081e-10, 1.384e-09, 1.788e-09, 2.189e-09, 2.686e-09, 3.091e-09,
00666
                3.49e-09, 3.796e-09, 4.2e-09, 5.103e-09, 6.005e-09, 6.3e-09, 6.706e-09, 7.07e-09, 7.434e-09, 7.663e-09, 7.788e-09, 7.8e-09, 7.597e-09,
00667
00668
                7.482e-09, 7.227e-09, 6.403e-09, 5.585e-09, 4.606e-09, 3.703e-09, 2.984e-09, 2.183e-09, 1.48e-09, 8.441e-10, 5.994e-10, 3.799e-10,
00669
00670
                2.751e-10, 1.927e-10, 1.507e-10, 1.102e-10, 6.971e-11, 5.839e-11,
                3.904e-11, 3.087e-11, 2.176e-11, 1.464e-11, 1.209e-11, 8.497e-12,
00672
00673
                6.477e-12, 4.371e-12, 2.914e-12, 2.424e-12, 1.753e-12, 1.35e-12,
00674
                9.417e-13, 6.622e-13, 5.148e-13, 3.841e-13, 3.446e-13, 3.01e-13,
00675
                2.551e-13, 2.151e-13, 1.829e-13, 1.64e-13, 1.475e-13, 1.352e-13,
00676
                1.155e-13, 9.963e-14, 9.771e-14, 9.577e-14, 9.384e-14, 9.186e-14,
00677
                 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14,
                9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14,
00678
00679
                9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14,
00680
                9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14
00681
00682
            static double o3[121] = {
00684
                2.218e-08, 3.394e-08, 3.869e-08, 4.219e-08, 4.501e-08, 4.778e-08,
                5.067e-08, 5.402e-08, 5.872e-08, 6.521e-08, 7.709e-08, 9.461e-08,
00685
00686
                1.269e-07, 1.853e-07, 2.723e-07, 3.964e-07, 5.773e-07, 8.2e-07,
                1.155e-06, 1.59e-06, 2.076e-06, 2.706e-06, 3.249e-06, 3.848e-06,
00687
                4.459e-06, 4.986e-06, 5.573e-06, 5.958e-06, 6.328e-06, 6.661e-06, 6.9e-06, 7.146e-06, 7.276e-06, 7.374e-06, 7.447e-06, 7.383e-06,
00688
00689
                7.321e-06, 7.161e-06, 6.879e-06, 6.611e-06, 6.216e-06, 5.765e-06,
00690
00691
                5.355e-06, 4.905e-06, 4.471e-06, 4.075e-06, 3.728e-06, 3.413e-06,
00692
                3.125e-06, 2.856e-06, 2.607e-06, 2.379e-06, 2.17e-06, 1.978e-06,
                1.8e-06, 1.646e-06, 1.506e-06, 1.376e-06, 1.233e-06, 1.102e-06, 9.839e-07, 8.771e-07, 7.814e-07, 6.947e-07, 6.102e-07, 5.228e-07, 4.509e-07, 3.922e-07, 3.501e-07, 3.183e-07, 2.909e-07, 2.686e-07,
00693
00694
00695
                2.476e-07, 2.284e-07, 2.109e-07, 2.003e-07, 2.013e-07, 2.022e-07,
                2.032e-07, 2.042e-07, 2.097e-07, 2.361e-07, 2.656e-07, 2.989e-07, 3.37e-07, 3.826e-07, 4.489e-07, 5.26e-07, 6.189e-07, 7.312e-07,
00697
00698
                8.496e-07, 8.444e-07, 8.392e-07, 8.339e-07, 8.286e-07, 8.234e-07, 8.181e-07, 8.129e-07, 8.077e-07, 8.026e-07, 6.918e-07, 5.176e-07,
00699
00700
00701
                3.865e-07, 2.885e-07, 2.156e-07, 1.619e-07, 1.219e-07, 9.161e-08,
                 6.972e-08, 5.399e-08, 3.498e-08, 2.111e-08, 1.322e-08, 8.482e-09,
00702
00703
                5.527e-09, 3.423e-09, 2.071e-09, 1.314e-09, 8.529e-10, 5.503e-10,
00704
                3.665e-10
00705
00706
00707
            static double ocs[121] = {
                6e-10, 6e-10, 6e-10, 6e-10, 6e-10, 6e-10, 6e-10, 6e-10, 5.997e-10,
                5.989e-10, 5.881e-10, 5.765e-10, 5.433e-10, 5.074e-10, 4.567e-10,
00709
00710
                4.067e-10, 3.601e-10, 3.093e-10, 2.619e-10, 2.232e-10,
                                                                                                           1.805e-10.
00711
               1.46e-10, 1.187e-10, 8.03e-11, 5.435e-11, 3.686e-11, 2.217e-11,
00712
                1.341e-11, 8.756e-12, 4.511e-12, 2.37e-12, 1.264e-12, 8.28e-13, 5.263e-13, 3.209e-13, 1.717e-13, 9.068e-14, 4.709e-14, 2.389e-14,
00713
00714
                1.236e-14, 1.127e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14,
                1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14,
00716
                1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14,
00717
                1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14,
00718
                1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14,
00719
                1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14,
00720
                1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14,
                1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14,
00721
                1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14,
00722
00723
                1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14,
00724
                1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14,
                1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 
00725
00726
00728
00729
00730
            static double sf6[121] = {
                4.103e-12, 4.03e-12, 4.087e-12, 4.064e-12, 4.023e-12,
00731
00732
```

```
3.988e-12, 3.941e-12, 3.884e-12, 3.755e-12, 3.622e-12, 3.484e-12,
               3.32e-12, 3.144e-12, 2.978e-12, 2.811e-12, 2.653e-12, 2.489e-12,
00734
00735
               2.332e-12, 2.199e-12, 2.089e-12, 2.013e-12, 1.953e-12, 1.898e-12,
00736
               1.859e-12, 1.826e-12, 1.798e-12, 1.776e-12, 1.757e-12, 1.742e-12,
00737
               1.728e-12, 1.717e-12, 1.707e-12, 1.698e-12, 1.691e-12, 1.685e-12,
00738
               1.679e-12, 1.675e-12, 1.671e-12, 1.668e-12, 1.665e-12, 1.663e-12,
               1.661e-12, 1.659e-12, 1.658e-12, 1.657e-12, 1.656e-12, 1.655e-12,
00740
               1.654e-12, 1.653e-12, 1.653e-12, 1.652e-12, 1.652e-12, 1.652e-12,
00741
               1.651e-12, 1.651e-12, 1.651e-12, 1.651e-12, 1.651e-12, 1.651e-12,
00742
               1.651e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12,
               1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12,
00743
00744
               1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12,
00745
               1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12,
00746
               1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12,
00747
               1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12,
00748
               1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12,
00749
               1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12
00750
00752
           static double so2[121] = {
00753
               le-10, le-10, le-10, le-10, le-10, le-10, le-10, le-10, le-10, le-10,
00754
               le-10, le-10, 9.867e-11, 9.537e-11, 9e-11, 8.404e-11, 7.799e-11,
               7.205e-11, 6.616e-11, 6.036e-11, 5.475e-11, 5.007e-11, 4.638e-11,
00755
               4.346e-11, 4.055e-11, 3.763e-11, 3.471e-11, 3.186e-11, 2.905e-11, 2.631e-11, 2.358e-11, 2.415e-11, 2.949e-11, 3.952e-11, 5.155e-11,
00756
00757
               6.76e-11, 8.741e-11, 1.099e-10, 1.278e-10, 1.414e-10, 1.512e-10,
00758
00759
               1.607e-10, 1.699e-10, 1.774e-10, 1.832e-10, 1.871e-10, 1.907e-10
00760
               1.943e-10, 1.974e-10, 1.993e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10,
00761
               2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10,
00762
               2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10,
               2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e
00763
00764
00765
               2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10,
00766
               2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10,
00767
               2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10
00768
00769
00770
           static int ig_co2 = -999;
00771
00772
           double co2, *q[NG] = {NULL};
00773
00774
           int ig, ip, iw, iz;
00775
00776
           /* Find emitter index of CO2... */
00777
           if (ig_co2 == -999)
00778
               ig_co2 = find_emitter(ct1, "CO2");
00779
00780
           /* Identify variable... */
00781
           for (ig = 0; ig < ctl->ng; ig++) {
              q[ig] = NULL;
00782
               if (strcasecmp(ctl->emitter[ig], "C2H2") == 0)
00784
                  q[ig] = c2h2;
00785
                   (strcasecmp(ctl->emitter[ig], "C2H6") == 0)
00786
                 q[ig] = c2h6;
               if (strcasecmp(ctl->emitter[ig], "CC14") == 0)
00787
00788
                 q[iq] = ccl4;
00789
                   (strcasecmp(ctl->emitter[ig], "CH4") == 0)
00790
                 q[ig] = ch4;
00791
                   (strcasecmp(ctl->emitter[ig], "ClO") == 0)
                  q[ig] = clo;
00792
00793
               if (strcasecmp(ctl->emitter[iq], "ClONO2") == 0)
00794
                 q[ig] = clono2;
00795
               if (strcasecmp(ctl->emitter[ig], "CO") == 0)
00796
                 q[ig] = co;
00797
                   (strcasecmp(ctl->emitter[ig], "COF2") == 0)
00798
                 q[ig] = cof2;
00799
               if (strcasecmp(ctl->emitter[ig], "F11") == 0)
00800
                 q[iq] = f11;
               if (strcasecmp(ctl->emitter[iq], "F12") == 0)
00801
                 q[ig] = f12;
00803
                    (strcasecmp(ctl->emitter[ig], "F14") == 0)
                 q[ig] = f14;
00804
00805
               if (strcasecmp(ctl->emitter[ig], "F22") == 0)
00806
                  q[ig] = f22;
                   (strcasecmp(ctl->emitter[ig], "H2O") == 0)
00807
               if
                  q[ig] = h2o;
00808
00809
                   (strcasecmp(ctl->emitter[ig], "H2O2") == 0)
00810
                  q[ig] = h2o2;
00811
               if (strcasecmp(ctl->emitter[ig], "HCN") == 0)
00812
                 q[iq] = hcn;
               if (strcasecmp(ctl->emitter[ig], "HNO3") == 0)
00813
                 q[ig] = hno3;
               if (strcasecmp(ctl->emitter[ig], "HNO4") == 0)
00815
                  q[ig] = hno4;
00816
00817
               if (strcasecmp(ctl->emitter[ig], "HOC1") == 0)
                 q[ig] = hocl;
00818
00819
               if (strcasecmp(ctl->emitter[iq], "N2O") == 0)
```

5.4 jurassic.c 103

```
q[iq] = n2o;
                        (strcasecmp(ctl->emitter[ig], "N2O5") == 0)
00821
                       q[ig] = n2o5;
00822
00823
                   if (strcasecmp(ctl->emitter[ig], "NH3") == 0)
00824
                      q[ig] = nh3;
00825
                   if (strcasecmp(ctl->emitter[ig], "NO") == 0)
                       q[ig] = no;
00827
                   if
                        (strcasecmp(ctl->emitter[ig], "NO2") == 0)
                      q[ig] = no2;
00828
                   if (strcasecmp(ctl->emitter[ig], "03") == 0)
00829
00830
                      q[ig] = o3;
                   if (strcasecmp(ctl->emitter[ig], "OCS") == 0)
00831
00832
                      q[ig] = ocs;
00833
                        (strcasecmp(ctl->emitter[ig], "SF6") == 0)
00834
                      q[ig] = sf6;
00835
                        (strcasecmp(ctl->emitter[ig], "SO2") == 0)
00836
                       q[ig] = so2;
00837
00838
00839
               /* Loop over atmospheric data points... */
00840
               for (ip = 0; ip < atm->np; ip++) {
00841
00842
                   /* Get altitude index... */
00843
                  iz = locate_reg(z, 121, atm->z[ip]);
00844
00845
                    /* Interpolate pressure... */
00846
                   atm \rightarrow p[ip] = EXP(z[iz], pre[iz], z[iz + 1], pre[iz + 1], atm \rightarrow z[ip]);
00847
                   /* Interpolate temperature... */
00848
00849
                   atm \rightarrow t[ip] = LIN(z[iz], tem[iz], z[iz + 1], tem[iz + 1], atm \rightarrow z[ip]);
00850
00851
                    /* Interpolate trace gases... */
00852
                   for (ig = 0; ig < ctl->ng; ig++)
00853
                       if (q[ig] != NULL)
00854
                           atm->q[ig][ip] =
                              \label{eq:linear} \mbox{LIN(z[iz], q[ig][iz], z[iz + 1], q[ig][iz + 1], atm->z[ip]);}
00855
00856
                       else
                          atm->q[iq][ip] = 0;
00858
00859
                    /* Set CO2... */
00860
                   if (ig_co2 >= 0) {
00861
                       co2 =
                          371.789948e-6 + 2.026214e-6 * (atm->time[ip] - 63158400.) / 31557600.
00862
00863
                       atm->q[ig\_co2][ip] = co2;
00864
00865
00866
                   /* Set extinction to zero... */
00867
                   for (iw = 0; iw < ctl->nw; iw++)
                       atm->k[iw][ip] = 0;
00868
00869
00870 }
00871
00873
00874 double ctmco2(
00875
               double nu,
00876
               double p,
00877
               double t.
00878
00879
               static double co2296[2001] = \{ 9.3388e-5, 9.7711e-5, 1.0224e-4, 1.0697e-4, 
00880
                 1.1193e-4, 1.1712e-4, 1.2255e-4, 1.2824e-4, 1.3419e-4, 1.4043e-4, 1.4695e-4, 1.5378e-4, 1.6094e-4, 1.6842e-4, 1.7626e-4, 1.8447e-4,
00881
00882
                   1.9307e-4, 2.0207e-4, 2.1149e-4, 2.2136e-4, 2.3169e-4, 2.4251e-4,
00883
                   2.5384e-4, 2.657e-4, 2.7813e-4, 2.9114e-4, 3.0477e-4, 3.1904e-4,
00884
00885
                   3.3399e-4, 3.4965e-4, 3.6604e-4, 3.8322e-4, 4.0121e-4, 4.2006e-4,
                   4.398e-4, 4.6047e-4, 4.8214e-4, 5.0483e-4, 5.286e-4, 5.535e-4, 5.7959e-4, 6.0693e-4, 6.3557e-4, 6.6558e-4, 6.9702e-4, 7.2996e-4, 7.6449e-4, 8.0066e-4, 8.3856e-4, 8.7829e-4, 9.1991e-4, 9.6354e-4,
00886
00887
00888
                   .0010093, .0010572, .0011074, .00116, .0012152, .001273,
00890
                   .0013336, .0013972, .0014638, .0015336, .0016068, .0016835,
00891
                   .001764, .0018483, .0019367, .0020295, .0021267, .0022286,
00892
                   .0023355, .0024476, .0025652, .0026885, .0028178, .0029534,
00893
                   .0030956, .0032448, .0034012, .0035654, .0037375, .0039181,
                   .0041076, .0043063, .0045148, .0047336, .0049632, .005204, .0054567, .0057219, .0060002, .0062923, .0065988, .0069204, .007258, .0076123, .0079842, .0083746, .0087844, .0092146,
00894
00895
00896
                   .0096663, .01014, .010638, .011161, .01171, .012286, .012891
.013527, .014194, .014895, .015631, .016404, .017217, .01807
00897
00898
                   .018966, .019908, .029897, .021936, .023028, .024176, .025382, .026649, .027981, .02938, .030851, .032397, .034023, .035732, .037528, .039416, .041402, .04349, .045685, .047994, .050422, .052975, .055661, .058486, .061458, .064584, .067873, .071334, .074975, .078807, .082839, .087082, .091549, .096249, .1012,
00899
00900
00901
00902
00903
00904
                    .10641, .11189, .11767, .12375, .13015, .13689, .14399, .15147,
                   .15935, .16765, .17639, .18561, .19531, .20554, .21632, .22769, .23967, .25229, .2656, .27964, .29443, .31004, .3265, .34386,
00905
00906
```

```
.36218, .3815, .40188, .42339, .44609, .47004, .49533, .52202,
                .5502, .57995, .61137, .64455, .6796, .71663, .75574, .79707, .84075, .88691, .9357, .98728, 1.0418, 1.0995, 1.1605, 1.225,
00908
00909
                1.2932, 1.3654, 1.4418, 1.5227, 1.6083, 1.6989, 1.7948, 1.8964, 2.004, 2.118, 2.2388, 2.3668, 2.5025, 2.6463, 2.7988, 2.9606, 3.1321, 3.314, 3.5071, 3.712, 3.9296, 4.1605, 4.4058, 4.6663,
00910
00911
00912
                4.9431, 5.2374, 5.5501, 5.8818, 6.2353, 6.6114, 7.0115, 7.4372
                7.8905, 8.3731, 8.8871, 9.4349, 10.019, 10.641, 11.305, 12.013,
00914
00915
                12.769, 13.576, 14.437, 15.358, 16.342, 17.39, 18.513, 19.716,
               21.003, 22.379, 23.854, 25.436, 27.126, 28.942, 30.89, 32.973, 35.219, 37.634, 40.224, 43.021, 46.037, 49.29, 52.803, 56.447, 60.418, 64.792, 69.526, 74.637, 80.182, 86.193, 92.713, 99.786, 107.47, 115.84, 124.94, 134.86, 145.69, 157.49, 170.3, 184.39, 199.83, 216.4, 234.55, 254.72, 276.82, 299.85, 326.16, 354.99,
00916
00917
00918
00919
00920
00921
                386.51, 416.68, 449.89, 490.12, 534.35, 578.25, 632.26, 692.61,
               756.43, 834.75, 924.11, 1016.9, 996.96, 1102.7, 1219.2, 1351.9, 1494.3, 1654.1, 1826.5, 2027.9, 2249., 2453.8, 2714.4, 2999.4, 3209.5, 3509., 3840.4, 3907.5, 4190.7, 4533.5, 4648.3, 5059.1, 5561.6, 6191.4, 6820.8, 7905.9, 9362.2, 2431.3, 2211.3, 2046.8,
00922
00923
00924
                2023.8, 1985.9, 1905.9, 1491.1, 1369.8, 1262.2, 1200.7, 887.74,
00926
                820.25, 885.23, 887.21, 816.73, 1126.9, 1216.2, 1272.4, 1579.5,
00927
               1634.2, 1656.3, 1657.9, 1789.5, 1670.8, 1509.5, 8474.6, 7489.2, 6793.6, 6117., 5574.1, 5141.2, 5084.6, 4745.1, 4413.2, 4102.8, 4024.7, 3715., 3398.6, 3100.8, 2900.4, 2629.2, 2374., 2144.7, 1955.8, 1760.8, 1591.2, 1435.2, 1296.2, 1174., 1065.1, 967.76,
00928
00929
00930
00931
                999.48, 897.45, 809.23, 732.77, 670.26, 611.93, 560.11, 518.77, 476.84, 438.8, 408.48, 380.21, 349.24, 322.71, 296.65, 272.85,
00933
00934
                251.96, 232.04, 213.88, 197.69, 182.41, 168.41, 155.79, 144.05,
                133.31, 123.48, 114.5, 106.21, 98.591, 91.612, 85.156, 79.204, 73.719, 68.666, 63.975, 59.637, 56.35, 52.545, 49.042, 45.788,
00935
00936
                42.78, 39.992, 37.441, 35.037, 32.8, 30.744, 28.801, 26.986,
00937
00938
                25.297, 23.731, 22.258, 20.883, 19.603, 18.403, 17.295, 16.249,
                15.271, 14.356, 13.501, 12.701, 11.954, 11.254, 10.6, 9.9864,
00939
00940
                9.4118, 8.8745, 8.3714, 7.8997, 7.4578, 7.0446, 6.6573, 6.2949,
                5.9577, 5.6395, 5.3419, 5.063, 4.8037, 4.5608, 4.3452, 4.1364, 3.9413, 3.7394, 3.562, 3.3932, 3.2325, 3.0789, 2.9318, 2.7898,
00941
00942
                2.6537, 2.5225, 2.3958, 2.2305, 2.1215, 2.0245, 1.9427, 1.8795,
00943
                1.8336, 1.7604, 1.7016, 1.6419, 1.5282, 1.4611, 1.3443, 1.27,
00945
                1.1675, 1.0824, 1.0534, .99833, .95854, .92981, .90887, .89346,
00946
                .88113, .87068, .86102, .85096, .88262, .86151, .83565, .80518,
00947
                .77045, .73736, .74744, .74954, .75773, .82267, .83493, .89402,
                .89725, .93426, .95564, .94045, .94174, .93404, .92035, .90456,
00948
                .88621, .86673, .78117, .7515, .72056, .68822, .65658, .62764, .55984, .55598, .57407, .60963, .63763, .66198, .61132, .60972,
00949
00950
                .52496, .50649, .41872, .3964, .32422, .27276, .24048, .23772, .2286, .22711, .23999, .32038, .34371, .36621, .38561, .39953,
00951
00952
00953
                .40636, .44913, .42716, .3919, .35477, .33935, .3351, .39746,
               .40936, .49313, .42716, .3913, .33474, .33933, .3331, .39746, .40993, .49998, .49956, .56157, .54742, .57295, .57386, .55417, .50745, .471, .43446, .39102, .34993, .31269, .27888, .24912, .22291, .19994, .17972, .16197, .14633, .13252, .12029, .10942, .099745, .091118, .083404, .076494, .070292, .064716, .059697,
00954
00955
00956
                .055173, .051093, .047411, .044089, .041092, .038392, .035965, .033789, .031846, .030122, .028607, .02729, .026169, .025209,
00958
00959
00960
                .024405, .023766, .023288, .022925, .022716, .022681, .022685,
                .022768, .023133, .023325, .023486, .024004, .024126, .024083,
00961
                .023785, .024023, .023029, .021649, .021108, .019454, .017809, .017292, .016635, .017037, .018068, .018977, .018756, .017847, .016557, .016142, .014459, .012869, .012381, .010875, .0098701
00962
00964
                .009285, .0091698, .0091701, .0096145, .010553, .01106, .012613, .014362, .015017, .016507, .017741, .01768, .017784, .0171,
00965
00966
                .016357, .016172, .017257, .018978, .020935, .021741, .023567,
00967
                .025183, .025589, .026732, .027648, .028278, .02815, .02856, .029015, .029062, .028851, .028497, .027825, .027801, .026523,
00968
                .02487, .022967, .022168, .020194, .018605, .017903, .018439, .019697, .020311, .020855, .020057, .018608, .016738, .015963,
00970
00971
00972
                .013844, .011801, .011134, .0097573, .0086007, .0086226,
                .0083721, .0090978, .0097616, .0098426, .011317, .012853, .014.
.014657, .015771, .016351, .016079, .014829, .013431, .013185,
.013207, .01448, .016176, .017971, .018265, .019526, .020455,
.019797, .019802, .0194, .018176, .017505, .016197, .015339,
00973
00974
                .014401, .013213, .012203, .011186, .010236, .0093288, .0084854,
00977
00978
                .0076837,\ .0069375,\ .0062614,\ .0056628,\ .0051153,\ .0046015,
                .0041501, .003752, .0033996, .0030865, .0028077, .0025586, .0023355, .0021353, .0019553, .0017931, .0016466, .0015141, .0013941, .0012852, .0011862, .0010962, .0010142, 9.3935e-4, 8.71e-4, 8.0851e-4, 7.5132e-4, 6.9894e-4, 6.5093e-4, 6.0689e-4,
00979
00980
00981
                5.6647e-4, 5.2935e-4, 4.9525e-4, 4.6391e-4, 4.3509e-4, 4.086e-4,
00983
00984
                3.8424e-4, 3.6185e-4, 3.4126e-4, 3.2235e-4, 3.0498e-4,
                                                                                                         2.8904e-4,
00985
                2.7444e-4, 2.6106e-4, 2.4883e-4, 2.3766e-4, 2.275e-4, 2.1827e-4,
                2.0992e-4, 2.0239e-4, 1.9563e-4, 1.896e-4, 1.8427e-4, 1.796e-4, 1.7555e-4, 1.7209e-4, 1.692e-4, 1.6687e-4, 1.6505e-4, 1.6375e-4,
00986
00987
                1.6294e-4, 1.6261e-4, 1.6274e-4, 1.6334e-4, 1.6438e-4, 1.6587e-4,
                1.678e-4, 1.7017e-4, 1.7297e-4, 1.762e-4, 1.7988e-4, 1.8399e-4,
00989
00990
                1.8855e-4, 1.9355e-4, 1.9902e-4, 2.0494e-4, 2.1134e-4, 2.1823e-4,
00991
                2.2561e-4, 2.335e-4, 2.4192e-4, 2.5088e-4, 2.604e-4, 2.705e-4,
                2.8119e-4, 2.9251e-4, 3.0447e-4, 3.171e-4, 3.3042e-4, 3.4447e-4, 3.5927e-4, 3.7486e-4, 3.9127e-4, 4.0854e-4, 4.267e-4, 4.4579e-4,
00992
00993
```

5.4 jurassic.c 105

```
4.6586e-4, 4.8696e-4, 5.0912e-4, 5.324e-4, 5.5685e-4, 5.8253e-4,
                        6.0949e-4, 6.378e-4, 6.6753e-4, 6.9873e-4, 7.3149e-4, 7.6588e-4,
00995
00996
                       8.0198e-4, 8.3987e-4, 8.7964e-4, 9.2139e-4, 9.6522e-4,
                                                                                                                                                        .0010112
                        .0010595, .0011102, .0011634, .0012193, .001278, .0013396,
00997
                       .0014043, .0014722, .0015436, .0016185, .0016972, .0017799, .0018668, .001958, .0020539, .0021547, .0022606, .0023719, .002489, .002612, .0027414, .0028775, .0030206, .0031712,
00998
00999
                        .0033295, .0034962, .0036716, .0038563, .0040506, .0042553,
01001
                       .0044709, .004698, .0049373, .0051894, .0054552, .0057354, .006031, .0063427, .0066717, .0070188, .0073854, .0077726,
01002
01003
                       .0081816, .0086138, .0090709, .0095543, .010066, .010607, .011181, .011789, .012433, .013116, .013842, .014613, .015432, .016304, .017233, .018224, .019281, .020394, .021574, .022836, .024181, .025594, .027088, .028707, .030401, .032245, .034219, .036262, .038539, .040987, .043578, .04641, .04949, .052726,
01004
01005
01007
01008
                      .056326, .0602, .064093, .068521, .073278, .077734, .083064, .088731, .093885, .1003, .1072, .11365, .12187, .13078, .13989, .15095, .16299, .17634, .19116, .20628, .22419, .24386, .26587, .28811, .31399, .34321, .36606, .39675, .42742, .44243, .47197, .49993, .49027, .51147, .52803, .48931, .49729, .5026, .43854,
01009
01010
01011
01013
                       .441, .44766, .43414, .46151, .50029, .55247, .43855, .32115, .32607, .3431, .36119, .38029, .41179, .43996, .47144, .51853
01014
01015
                       .55362, .59122, .66338, .69877, .74001, .82923, .86907, .90361
1.0025, 1.031, 1.0559, 1.104, 1.1178, 1.1341, 1.1547, 1.351,
1.4772, 1.4812, 1.4907, 1.512, 1.5442, 1.5853, 1.6358, 1.6963,
                                                                                                                                                           .90361.
01016
01017
                       1.7674, 1.8474, 1.9353, 2.0335, 2.143, 2.2592, 2.3853, 2.5217, 2.6686, 2.8273, 2.9998, 3.183, 3.3868, 3.6109, 3.8564, 4.1159,
01020
                      2.686, 2.82/3, 2.9998, 3.183, 3.3868, 3.6109, 3.8564, 4.1159, 4.4079, 4.7278, 5.0497, 5.3695, 5.758, 6.0834, 6.4976, 6.9312, 7.38, 7.5746, 7.9833, 8.3791, 8.3956, 8.7501, 9.1067, 9.072, 9.4649, 9.9112, 10.402, 10.829, 11.605, 12.54, 12.713, 10.443, 10.825, 11.375, 11.955, 12.623, 13.326, 14.101, 15.041, 15.547, 16.461, 17.439, 18.716, 19.84, 21.036, 22.642, 23.901, 25.244, 27.03, 28.411, 29.871, 31.403, 33.147, 34.744, 36.456, 39.239, 12.603, 45.163, 47.044, 45.163, 47.044, 45.163, 47.044, 45.163, 47.044, 45.163, 47.044, 45.163, 47.044, 45.163, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044, 47.044,
01021
01022
01023
01024
01026
01027
                       43.605, 45.162, 47.004, 49.093, 51.391, 53.946, 56.673, 59.629,
                       63.167, 66.576, 70.254, 74.222, 78.477, 83.034, 87.914, 93.18, 98.77, 104.74, 111.15, 117.95, 125.23, 133.01, 141.33, 150.21,
01028
01029
                       159.71, 169.89, 180.93, 192.54, 204.99, 218.34, 232.65, 248., 264.47, 282.14, 301.13, 321.53, 343.48, 367.08, 392.5, 419.88,
01030
01032
                        449.4, 481.26, 515.64, 552.79, 592.99, 636.48, 683.61, 734.65,
                       789.99, 850.02, 915.14, 985.81, 1062.5, 1147.1, 1237.8, 1336.4, 1443.2, 1558.9, 1684.2, 1819.2, 1965.2, 2122.6, 2291.7, 2470.8,
01033
01034
                       2665.7, 2874.9, 3099.4, 3337.9, 3541., 3813.3, 4111.9, 4439.3, 4798.9, 5196., 5639.2, 6087.5, 6657.7, 7306.7, 8040.7, 8845.5, 9702.2, 10670., 11739., 12842., 14141., 15498., 17068., 18729.,
01035
01036
                       20557., 22559., 25248., 27664., 30207., 32915., 35611., 38081.,
01039
                       40715., 43191., 41651., 42750., 43785., 44353., 44366., 44189.,
01040
                       43618., 42862., 41878., 35133., 35215., 36383., 39420., 44055.,
01041
                       44155., 45850., 46853., 39197., 38274., 29942., 28553., 21792.,
                       21228., 17106., 14955., 18181., 19557., 21427., 23728., 26301.,
01042
                       28584., 30775., 32536., 33867., 40089., 39204., 37329., 34452.,
01043
                        31373., 33921., 34800., 36043., 44415., 45162., 52181., 50895.,
                       54140., 50840., 50468., 48302., 44915., 40910., 36754., 32755.,
01045
01046
                       29093., 25860., 22962., 20448., 18247., 16326., 14645., 13165.,
                       11861., 10708., 9686.9, 8779.7, 7971.9, 7250.8, 6605.7, 6027.2, 5507.3, 5039.1, 4616.6, 4234.8, 3889., 3575.4, 3290.5, 3031.3, 2795.2, 2579.9, 2383.1, 2203.3, 2038.6, 1887.6, 1749.1, 1621.9, 1505., 1397.4, 1298.3, 1207., 1122.8, 1045., 973.1, 906.64,
01047
01048
01049
                       845.16, 788.22, 735.48, 686.57, 641.21, 599.1, 559.99, 523.64,
01051
                       489.85, 458.42, 429.16, 401.92, 376.54, 352.88, 330.82, 310.24,
01052
01053
                       291.03, 273.09, 256.34, 240.69, 226.05, 212.37, 199.57, 187.59,
                       176.37, 165.87, 156.03, 146.82, 138.17, 130.07, 122.47, 115.34, 108.65, 102.37, 96.473, 90.934, 85.73, 80.84, 76.243, 71.922, 67.858, 64.034, 60.438, 57.052, 53.866, 50.866, 48.04, 45.379,
01054
01055
                        42.872, 40.51, 38.285, 36.188, 34.211, 32.347, 30.588, 28.929,
                       27.362, 25.884, 24.489, 23.171, 21.929, 20.755, 19.646, 18.599,
01058
                      17.61, 16.677, 15.795, 14.961, 14.174, 13.43, 12.725, 12.06, 11.431, 10.834, 10.27, 9.7361, 9.2302, 8.7518, 8.2997, 7.8724, 7.4674, 7.0848, 6.7226, 6.3794, 6.054, 5.745, 5.4525, 5.1752,
01059
01060
01061
                       4.9121, 4.6625, 4.4259, 4.2015, 3.9888, 3.7872, 3.5961, 3.4149,
01062
                       3.2431, 3.0802, 2.9257, 2.7792, 2.6402, 2.5084, 2.3834, 2.2648,
                       2.1522, 2.0455, 1.9441, 1.848, 1.7567, 1.6701, 1.5878, 1.5097,
01064
01065
                       1.4356, 1.3651, 1.2981, 1.2345, 1.174, 1.1167, 1.062, 1.0101,
01066
                        .96087, .91414, .86986, .82781, .78777, .74971, .71339, .67882,
                       .64604, .61473, .58507, .55676, .52987, .5044, .48014, .45715, .43527, .41453, .3948, .37609, .35831, .34142, .32524, .30995, .29536, .28142, .26807, .25527, .24311, .23166, .22077, .21053, .20081, .19143, .18261, .17407, .16603, .15833, .15089, .14385,
01067
01068
01070
                       13707, 13065, .12449, .11865, .11306, .10774, .10266, .097818, .093203, .088815, .084641, .080671, .076892, .073296, .069873, .066613, .06351, .060555, .05774, .055058, .052504, .050071, .047752, .045543, .043438, .041432, .039521, .037699, .035962, .034307, .032729, .031225, .029791, .028423, .02712, .025877,
01071
01072
01073
01074
                       .024692, .023563, .022485, .021458, .020478, .019543, .018652, .017802, .016992, .016219, .015481, .014778, .014107, .013467,
01076
01077
                       .012856, .012274, .011718, .011188, .010682, .0102, .0097393, .0093001, .008881, .0084812, .0080997, .0077358, .0073885, .0070571, .0067409, .0064393, .0061514, .0058768, .0056147,
01078
01079
01080
```

```
.0053647, .0051262, .0048987, .0046816, .0044745, .0042769,
                          .0040884, .0039088, .0037373, .0035739, .003418, .0032693, .0031277, .0029926, .0028639, .0027413, .0026245, .0025133,
01082
01083
                           .0024074, .0023066, .0022108, .0021196, .002033, .0019507,
01084
01085
                           .0018726, .0017985, .0017282, .0016617, .0015988, .0015394,
                          .0014834, .0014306, .0013811, .0013346, .0012911, .0012506, .0012131, .0011784, .0011465, .0011175, .0010912, .0010678, .0010472, .0010295, .0010147, .001003, 9.9428e-4, 9.8883e-4,
01086
01088
01089
                           9.8673e-4, 9.8821e-4, 9.9343e-4, .0010027, .0010164, .0010348,
01090
                          .0010586, .0010882, .0011245, .0011685, .0012145, .0012666,
                          .0013095, .0013688, .0014048, .0014663, .0015309, .0015499, .0016144, .0016312, .001705, .0017892, .0018499, .0019715, .0021102, .0022442, .0024284, .0025893, .0027703, .0029445,
01091
01092
01093
                          .0031193, .003346, .0034552, .0036906, .0037584, .0040084, .0041934, .0044587, .0047093, .0049759, .0053421, .0055134,
01094
01095
01096
                          .0059048, .0058663, .0061036, .0063259, .0059657, .0060653,
                          .0060972, .0055539, .0055653, .0055772, .005331, .0054953, .0055919, .0058684, .006183, .0066675, .0069808, .0075142, .0078536, .0084282, .0089454, .0094625, .0093703, .0095857,
01097
01098
                          .0099283, .010063, .010521, .0097778, .0098175, .010379, .010447,
01100
                          .0105, .010617, .010706, .01078, .011177, .011212, .011304,
01101
01102
                           .011446, .011603, .011816, .012165, .012545, .013069, .013539,
                         .01411, .014776, .016103, .017016, .017994, .018978, .01998, .021799, .022745, .023681, .024627, .025562, .026992, .027958, .029013, .030154, .031402, .03228, .033651, .035272, .037088, .039021, .041213, .043597, .045977, .04877, .051809, .054943, .058064, .061528, .06537, .069309, .071928, .075752, .079589,
01103
01104
01105
01107
01108
                           .083352, .084096, .087497, .090817, .091198, .094966, .099045
                         .084352, .084096, .087497, .090817, .091198, .094966, .099045, .10429, .10867, .11518, .12269, .13126, .14087, .15161, .16388, .16423, .1759, .18721, .19994, .21275, .22513, .23041, .24231, .25299, .25396, .26396, .27696, .27929, .2908, .30595, .31433, .3282, .3429, .35944, .37467, .39277, .41245, .43326, .45649, .48152, .51897, .54686, .57877, .61263, .64962, .68983, .73945, .78619, .83537, .89622, .95002, 1.0067, 1.0742, 1.1355, 1.2007, .78619, .83537, .89622, .95002, 1.0067, 1.0742, 1.1355, 1.2007, .78619, .83537, .89622, .95002, 1.0067, 1.0742, 1.1355, 1.2007, .78619, .83537, .89622, .95002, 1.0067, 1.0742, 1.1355, 1.2007, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78619, .78
01109
01110
01111
01112
01113
01114
                          1.2738, 1.347, 1.4254, 1.5094, 1.6009, 1.6976, 1.8019, 1.9148, 2.0357, 2.166, 2.3066, 2.4579, 2.6208, 2.7966, 2.986, 3.188,
01115
01116
                          3.4081, 3.6456, 3.9, 4.1747, 4.4712, 4.7931, 5.1359, 5.5097, 5.9117, 6.3435, 6.8003, 7.3001, 7.8385, 8.3945, 9.011, 9.6869, 10.392, 11.18, 12.036, 12.938, 13.944, 14.881, 16.029, 17.255,
01117
01119
                           18.574, 19.945, 21.38, 22.9, 24.477, 26.128, 27.87, 29.037,
01120
                         30.988, 33.145, 35.506, 37.76, 40.885, 44.487, 48.505, 52.911, 57.56, 61.964, 67.217, 72.26, 78.343, 85.08, 91.867, 99.435, 107.68, 116.97, 127.12, 138.32, 150.26, 163.04, 174.81, 189.26, 205.61, 224.68, 240.98, 261.88, 285.1, 307.58, 334.35, 363.53, 394.68, 427.85, 458.85, 489.25, 472.87, 486.93, 496.27, 501.52, 407.44, 489.26, 472.87, 486.93, 496.27, 501.52, 489.25, 472.87, 486.93, 496.27, 501.52, 489.25, 472.87, 486.93, 496.27, 501.52, 489.25, 472.87, 486.93, 496.27, 501.52, 489.25, 472.87, 486.93, 496.27, 501.52, 489.25, 472.87, 486.93, 496.27, 501.52, 489.25, 472.87, 486.93, 496.27, 501.52, 489.25, 472.87, 486.93, 496.27, 501.52, 489.25, 472.87, 486.93, 496.27, 501.52, 489.25, 472.87, 486.93, 496.27, 501.52, 489.25, 472.87, 486.93, 496.27, 501.52, 489.25, 472.87, 486.93, 496.27, 501.52, 489.25, 472.87, 486.93, 496.27, 501.52, 489.25, 472.87, 486.93, 496.27, 501.52, 489.25, 472.87, 486.93, 496.27, 501.52, 489.25, 472.87, 486.93, 496.27, 501.52, 489.25, 472.87, 486.93, 496.27, 501.52, 489.25, 472.87, 486.93, 496.27, 501.52, 489.25, 472.87, 486.93, 496.27, 501.52, 489.25, 472.87, 486.93, 496.27, 501.52, 489.25, 472.87, 486.93, 496.27, 501.52, 489.25, 472.87, 486.93, 496.27, 501.52, 489.25, 472.87, 486.93, 496.27, 501.52, 489.25, 472.87, 486.93, 496.27, 501.52, 489.25, 472.87, 486.93, 496.27, 501.52, 489.25, 472.87, 486.93, 496.27, 501.52, 489.25, 472.87, 486.93, 489.25, 472.87, 486.93, 486.27, 501.52, 489.25, 472.87, 486.93, 486.27, 501.52, 489.25, 472.87, 486.93, 486.27, 501.52, 489.25, 472.87, 486.93, 486.27, 501.52, 489.25, 472.87, 486.93, 486.27, 501.52, 489.25, 472.87, 486.93, 486.27, 501.52, 489.25, 472.87, 486.93, 486.27, 501.52, 489.25, 472.87, 486.93, 486.27, 501.52, 489.25, 489.25, 489.25, 489.25, 489.25, 489.25, 489.25, 489.25, 489.25, 489.25, 489.25, 489.25, 489.25, 489.25, 489.25, 489.25, 489.25, 489.25, 489.25, 489.25, 489.25, 489.25, 489.25, 489.25, 489.25, 489.25, 489.25, 489.25, 489.25, 489.25, 489.25, 489.25, 489.25, 489.25, 489.25, 489.25, 489.25, 489.25, 489.25, 489.25, 489.25, 489.25, 489
01121
01122
01123
01125
01126
                           501.57, 497.14, 488.09, 476.32, 393.76, 388.51, 393.42, 414.45,
                          455.12, 514.62, 520.38, 547.42, 562.6, 487.47, 480.83, 391.06, 376.92, 303.7, 295.91, 256.03, 236.73, 280.38, 310.71, 335.53,
01127
01128
                          367.88, 401.94, 435.52, 469.13, 497.94, 588.82, 597.94, 597.2,
01129
                          588.28, 571.2, 555.75, 603.56, 638.15, 680.75, 801.72, 848.01, 962.15, 990.06, 1068.1, 1076.2, 1115.3, 1134.2, 1136.6, 1119.1,
01130
                           1108.9, 1090.6, 1068.7, 1041.9, 1005.4, 967.98, 927.08, 780.1,
01132
01133
                           751.41, 733.12, 742.65, 785.56, 855.16, 852.45, 878.1, 784.59,
                          777.81, 765.13, 622.93, 498.09, 474.89, 386.9, 378.48, 336.17, 322.04, 329.57, 350.5, 383.38, 420.02, 462.39, 499.71, 531.98, 654.99, 653.43, 639.99, 605.16, 554.16, 504.42, 540.64, 552.33,
01134
01135
01136
                           679.46, 699.51, 713.91, 832.17, 919.91, 884.96, 907.57, 846.56,
                          818.56, 768.93, 706.71, 642.17, 575.95, 515.38, 459.07, 409.02,
01138
                          364.61, 325.46, 291.1, 260.89, 234.39, 211.01, 190.38, 172.11,
01139
                          155.91, 141.49, 128.63, 117.13, 106.84, 97.584, 89.262, 81.756, 74.975, 68.842, 63.28, 58.232, 53.641, 49.46, 45.649, 42.168, 38.991, 36.078, 33.409, 30.96, 28.71, 26.642, 24.737, 22.985, 21.37, 19.882, 18.512, 17.242, 16.073, 14.987, 13.984, 13.05,
01140
01141
01142
01143
                          12.186, 11.384, 10.637, 9.9436, 9.2988, 8.6991, 8.141, 7.6215
7.1378, 6.6872, 6.2671, 5.8754, 5.51, 5.1691, 4.851, 4.5539,
01145
01146
                          4.2764, 4.0169, 3.7742, 3.5472, 3.3348, 3.1359, 2.9495, 2.7749,
01147
                          2.6113, 2.4578, 2.3139, 2.1789, 2.0523, 1.9334, 1.8219, 1.7171, 1.6188, 1.5263, 1.4395, 1.3579, 1.2812, 1.209, 1.1411, 1.0773,
01148
01149
                          1.0171, .96048, .90713, .85684, .80959, .76495, .72282, .68309,
                           .64563, .61035, .57707, .54573, .51622, .48834, .46199, .43709,
                           .41359, .39129, .37034, .35064, .33198, .31442, .29784, .28218, .26732, .25337, .24017, .22774, .21601, .20479, .19426
01151
01152
01153
01154
                    static double co2260[2001] = { 5.7971e-5, 6.0733e-5, 6.3628e-5, 6.6662e-5,
01155
                           6.9843e-5, 7.3176e-5, 7.6671e-5, 8.0334e-5, 8.4175e-5, 8.8201e-5,
                          9.2421e-5, 9.6846e-5, 1.0149e-4, 1.0635e-4, 1.1145e-4, 1.1679e-4,
01157
01158
                           1.224e-4, 1.2828e-4, 1.3444e-4, 1.409e-4, 1.4768e-4, 1.5479e-4,
                          1.6224e-4, 1.7006e-4, 1.7826e-4, 1.8685e-4, 1.9587e-4, 2.0532e-4, 2.1524e-4, 2.2565e-4, 2.3656e-4, 2.48e-4, 2.6001e-4, 2.7261e-4,
01159
01160
                          2.8582e-4, 2.9968e-4, 3.1422e-4, 3.2948e-4, 3.4548e-4, 3.6228e-4,
01161
                           3.799e-4, 3.9838e-4, 4.1778e-4, 4.3814e-4, 4.595e-4, 4.8191e-4,
                           5.0543e-4, 5.3012e-4, 5.5603e-4, 5.8321e-4, 6.1175e-4, 6.417e-4,
01163
01164
                           6.7314e-4, 7.0614e-4, 7.4078e-4, 7.7714e-4, 8.1531e-4, 8.5538e-4,
                          8.9745e-4, 9.4162e-4, 9.8798e-4, .0010367, .0010878, .0011415, .0011978, .001257, .0013191, .0013844, .001453, .0015249, .0016006, .00168, .0017634, .001851, .001943, .0020397, .0021412,
01165
01166
01167
```

```
.0022479, .00236, .0024778, .0026015, .0027316, .0028682,
                .0030117, .0031626, .0033211, .0034877, .0036628, .0038469, .0040403, .0042436, .0044574, .004682, .0049182, .0051665,
01170
01171
                 .0054276, .0057021, .0059907, .0062942, .0066133, .0069489,
                 .0073018, .0076729, .0080632, .0084738, .0089056,
01172
                                                                                                     .0093599.
                .0098377, .01034, .010869, .011426, .012011, .012627, .013276, .013958, .014676, .015431, .016226, .017063, .017944, .018872, .019848, .020876, .021958, .023098, .024298, .025561, .026892,
01173
01175
                 .028293, .029769, .031323, .032961, .034686, .036503, .038418, .040435, .042561, .044801, .047161, .049649, .052271, .055035,
01176
01177
                .047161, .0449649, .052271, .055035, .057948, .061019, .064256, .06767, .07127, .075066, .079069, .083291, .087744, .092441, .097396, .10262, .10814, .11396, .1201, .12658, .13342, .14064, .14826, .1563, .1648, .17376, .18323, .19324, .2038, .21496, .22674, .23919, .25234, .26624, .
01178
01179
01180
01181
01182
                 .28093, .29646, .31287, .33021, .34855, .36794, .38844, .41012,
                .43305, .45731, .48297, .51011, .53884, .56924, .60141, .63547, .67152, .70969, .75012, .79292, .83826, .8863, .93718, .99111, 1.0482, 1.1088, 1.173, 1.2411, 1.3133, 1.3898, 1.471, 1.5571, 1.6485, 1.7455, 1.8485, 1.9577, 2.0737, 2.197, 2.3278, 2.4668,
01183
01184
01185
                 2.6145, 2.7715, 2.9383, 3.1156, 3.3042, 3.5047, 3.7181, 3.9451,
                4.1866, 4.4437, 4.7174, 5.0089, 5.3192, 5.65, 6.0025, 6.3782, 6.7787, 7.206, 7.6617, 8.1479, 8.6669, 9.221, 9.8128, 10.445,
01188
01189
01190
                11.12, 11.843, 12.615, 13.441, 14.325, 15.271, 16.283, 17.367
                18.529, 19.776, 21.111, 22.544, 24.082, 25.731, 27.504, 29.409, 31.452, 33.654, 36.024, 38.573, 41.323, 44.29, 47.492, 50.951,
01191
01192
01193
                 54.608, 58.588, 62.929, 67.629, 72.712, 78.226, 84.207, 90.699
                 97.749, 105.42, 113.77, 122.86, 132.78, 143.61, 155.44, 168.33,
01194
01195
                182.48, 198.01, 214.87, 233.39, 253.86, 276.34, 300.3, 327.28,
01196
                 356.89, 389.48, 422.29, 458.99, 501.39, 548.13, 595.62, 652.74,
01197
                 716.54, 784.57, 866.78, 960.59, 1062.8, 1072.5, 1189.5, 1319.4,
                 1467.6, 1630.2, 1813.7, 2016.9, 2253., 2515.3, 2773.5, 3092.8,
01198
01199
                 3444.4, 3720.4, 4104.3, 4527.5, 4645.9, 5021.7, 5462.2, 5597.,
                 6110.6, 6732.5, 7513.8, 8270.6, 9640.6, 11487., 2796.1, 2680.1
01200
01201
                 2441.6, 2404.2, 2334.8, 2215.2, 1642.5, 1477.9, 1328.1, 1223.5,
                843.34, 766.96, 831.65, 834.84, 774.85, 1156.3, 1275.6, 1366.1, 1795.6, 1885., 1936.5, 1953.4, 2154.4, 2002.7, 1789.8, 10381., 9040., 8216.5, 7384.7, 6721.9, 6187.7, 6143.8, 5703.9, 5276.6, 4873.1, 4736., 4325.3, 3927., 3554.1, 3286.1, 2950.1, 2642.4,
01202
01203
01204
01206
                 2368.7, 2138.9, 1914., 1719.6, 1543.9, 1388.6, 1252.1, 1132.2,
                 1024.1, 1025.4, 920.58, 829.59, 750.54, 685.01, 624.25, 570.14
01207
01208
                 525.81, 481.85, 441.95, 408.71, 377.23, 345.86, 318.51, 292.26,
                268.34, 247.04, 227.14, 209.02, 192.69, 177.59, 163.78, 151.26, 139.73, 129.19, 119.53, 110.7, 102.57, 95.109, 88.264, 81.948, 76.13, 70.768, 65.827, 61.251, 57.022, 53.495, 49.824, 46.443,
01209
01210
                43.307, 40.405, 37.716, 35.241, 32.923, 30.77, 28.78, 26.915, 25.177, 23.56, 22.059, 20.654, 19.345, 18.126, 16.988, 15.93,
01212
01213
01214
                14.939, 14.014, 13.149, 12.343, 11.589, 10.884, 10.225, 9.6093,
                9.0327, 8.4934, 7.9889, 7.5166, 7.0744, 6.6604, 6.2727, 5.9098, 5.5701, 5.2529, 4.955, 4.676, 4.4148, 4.171, 3.9426, 3.7332, 3.5347, 3.3493, 3.1677, 3.0025, 2.8466, 2.6994, 2.5601, 2.4277, 2.3016, 2.1814, 2.0664, 1.9564, 1.8279, 1.7311, 1.6427, 1.5645,
01215
01216
01217
                 1.4982, 1.443, 1.374, 1.3146, 1.2562, 1.17, 1.1105, 1.0272,
01219
01220
                 .96863, .89718, .83654, .80226, .75908, .72431, .69573, .67174,
01221
                 .65126, .63315, .61693, .60182, .58715, .59554, .57649,
                                                                                                                 55526,
                .53129, .501313, .501633, .501634, .501634, .535324, .535324, .53177, .50622, .48176, .4813, .47642, .47492, .50273, .50293, .52687, .52239, .53419, .53814, .52626, .52211, .51492, .50622, .49746, .48841, .4792, .43534, .41999, .40349, .38586, .36799, .35108, .31089, .30803, .3171, .33599, .35041, .36149, .32924,
01222
01223
01225
                 .32462, .27309, .25961, .20922, .19504, .15683, .13098, .11588
01226
01227
                 .11478, .11204, .11363, .12135, .16423, .17785, .19094, .20236,
                21084, 2154, 24108, 22848, 20871, 18797, 17963, 17834, 21552, 22284, 26945, 27052, 30108, 28977, 29772, 29224, 27658, 24956, 22777, 20654, 18392, 16338, 1452, 12916, 1152, 10304, 092437, 083163, 075031, .067878, .061564,
01228
01229
01231
                 .055976, .051018, .046609, .042679, .03917, .036032, .033223,
01232
01233
                 .030706, .02845, .026428, .024617, .022998, .021554, .02027,
                .019136, .018141, .017278, .016541, .015926, .015432, .015058, .014807, .014666, .014635, .014728, .014947, .01527, .015728, .016345, .017026, .017798, .018839, .019752, .020636, .021886,
01234
01235
01236
                 .022695, .02327, .023478, .024292, .023544, .022222, .021932,
                .020052, .018143, .017722, .017031, .017782, .01938, .020734, .020476, .019255, .017477, .016878, .014617, .012489, .011765,
01238
01239
                .0099077, .0086446, .0079446, .0078644, .0079763, .008671, .01001, .0108, .012933, .015349, .016341, .018484, .020254,
01240
01241
                .020254, .020478, .019591, .018595, .018385, .019913, .022254, .024847, .025809, .028053, .029924, .030212, .031367, .03222, .032739, .032537, .03286, .033344, .033507, .033499, .033339,
01242
01243
01244
01245
                 .032809, .033041, .031723, .029837, .027511, .026603, .024032,
01246
                 .021914, .020948, .021701, .023425, .024259, .024987, .023818,
                .021768, .019223, .018144, .015282, .012604, .01163, .0097907, .008336, .0082473, .0079582, .0088077, .009779, .010129, .012145,
01247
01248
                .014378, .016761, .01726, .018997, .019998, .019899, .01819, .016358, .016099, .01617, .017939, .020223, .022521, .02277,
01250
                 .024279, .025247, .024222, .023989, .023224, .021493, .020362,
01251
                .018596, .017309, .015975, .014466, .013171, .011921, .01078, .0097229, .0087612, .0078729, .0070682, .0063494, .0057156, .0051459, .0046273, .0041712, .0037686, .0034119, .003095,
01252
01253
01254
```

```
.0028126, .0025603, .0023342, .0021314, .0019489, .0017845,
                   .001636, .0015017, .00138, .0012697, .0011694, .0010782,
01256
                   9.9507e-4, 9.1931e-4, 8.5013e-4, 7.869e-4, 7.2907e-4, 6.7611e-4,
01257
                   6.2758e-4, 5.8308e-4, 5.4223e-4, 5.0473e-4, 4.7027e-4, 4.3859e-4,
01258
                  4.0946e-4, 3.8265e-4, 3.5798e-4, 3.3526e-4, 3.1436e-4, 2.9511e-4, 2.7739e-4, 2.6109e-4, 2.4609e-4, 2.3229e-4, 2.1961e-4, 2.0797e-4,
01259
01260
                   1.9729e-4, 1.875e-4, 1.7855e-4, 1.7038e-4, 1.6294e-4, 1.5619e-4,
                   1.5007e-4, 1.4456e-4, 1.3961e-4, 1.3521e-4, 1.3131e-4, 1.2789e-4,
01262
01263
                   1.2494e-4, 1.2242e-4, 1.2032e-4, 1.1863e-4, 1.1733e-4, 1.1641e-4,
                  1.1585e-4, 1.1565e-4, 1.158e-4, 1.1629e-4, 1.1712e-4, 1.1827e-4, 1.1976e-4, 1.2158e-4, 1.2373e-4, 1.262e-4, 1.2901e-4, 1.3214e-4, 1.3562e-4, 1.3944e-4, 1.4361e-4, 1.4814e-4, 1.5303e-4, 1.5829e-4,
01264
01265
01266
                   1.6394e-4, 1.6999e-4, 1.7644e-4, 1.8332e-4, 1.9063e-4, 1.984e-4,
                  2.0663e-4, 2.1536e-4, 2.246e-4, 2.3436e-4, 2.4468e-4, 2.5558e-4, 2.6708e-4, 2.7921e-4, 2.92e-4, 3.0548e-4, 3.1968e-4, 3.3464e-4,
01268
01269
                  3.5039e-4, 3.6698e-4, 3.8443e-4, 4.0281e-4, 4.2214e-4, 4.4248e-4, 4.6389e-4, 4.864e-4, 5.1009e-4, 5.3501e-4, 5.6123e-4, 5.888e-4, 6.1781e-4, 6.4833e-4, 6.8043e-4, 7.142e-4, 7.4973e-4, 7.8711e-4,
01270
01271
01272
                  8.2644e-4, 8.6783e-4, 9.1137e-4, 9.5721e-4, .0010054, .0010562,
                  .0011096, .0011659, .0012251, .0012875, .0013532, .0014224,
01274
                  .0014953, .001572, .0016529, .0017381, .0018279, .0019226, .0020224, .0021277, .0022386, .0023557, .0024792, .0026095,
01275
01276
                  .002747, .0028921, .0030453, .0032071, .003788, .0035586, .0037494, .003951, .0041642, .0043897, .0046282, .0048805, .0051476, .0054304, .00573, .0060473, .0063837, .0067404, .0071188, .0075203, .0079466, .0083994, .0088806, .0093922,
01277
01278
                   .0099366, .010516, .011134, .011792, .012494, .013244, .014046,
01281
01282
                   .014898, .015808, .016781, .017822, .018929, .020108,
                  .014898, .015808, .016/81, .017822, .018929, .020108, .02138, .022729, .02419, .02576, .027412, .029233, .031198, .0333301, .035594, .038092, .040767, .04372, .046918, .050246, .053974, .058009, .061976, .066586, .071537, .076209, .081856, .087998, .093821, .10113, .10913, .11731, .12724, .13821, .15025, .1639, .17807, .19472, .21356, .23496, .25758, .28387, .31389, .34104, .37469, .40989, .43309, .46845, .5042, .5023, .52981, .5275,
01283
01284
01285
01287
01288
                  .37469, .40389, .43309, .446043, .3042, .3023, .32961, .33273, .51075, .51976, .52457, .44779, .44721, .4503, .4243, .45244, .49491, .55399, .39021, .24802, .2501, .2618, .27475, .28879, .31317, .33643, .36257, .4018, .43275, .46525, .53333, .56599, .60557, .70142, .74194, .77736, .88567, .91182, .93294, .98407, .98772, .99176, .9995, 1.2405, 1.3602, 1.338, 1.3255, 1.3267, 1.3404, 1.3634, 1.3967, 1.4407, 1.4961, 1.5603, 1.6328, 1.7153, 1.8094, 1.9091, 2.018, 2.1367, 2.264, 2.4035, 2.5562, 2.7179, 2.9017, 3.1052, 3.3304, 3.5731, 3.8488, 4.1553, 4.4769, 4.7818,
01289
01290
01291
01293
01294
01295
                  2.9017, 3.1052, 3.3304, 3.5731, 3.8488, 4.1553, 4.4769, 4.7818, 5.1711, 5.5204, 5.9516, 6.4097, 6.8899, 7.1118, 7.5469, 7.9735, 7.9511, 8.3014, 8.6418, 8.4757, 8.8256, 9.2294, 9.6923, 10.033,
01296
01297
                  10.842, 11.851, 11.78, 8.8435, 9.1381, 9.5956, 10.076, 10.629, 11.22, 11.883, 12.69, 13.163, 13.974, 14.846, 16.027, 17.053,
01299
01300
                  18.148, 19.715, 20.907, 22.163, 23.956, 25.235, 26.566, 27.94, 29.576, 30.956, 32.432, 35.337, 39.911, 41.128, 42.625, 44.386, 46.369, 48.619, 51.031, 53.674, 56.825, 59.921, 63.286, 66.929,
01301
01302
01303
                  70.859, 75.081, 79.618, 84.513, 89.739, 95.335, 101.35, 107.76, 114.63, 121.98, 129.87, 138.3, 147.34, 157.04, 167.56, 178.67,
01304
                   190.61, 203.43, 217.19, 231.99, 247.88, 264.98, 283.37, 303.17,
01306
01307
                   324.49, 347.47, 372.25, 398.98, 427.85, 459.06, 492.8, 529.31,
                  568.89, 611.79, 658.35, 708.91, 763.87, 823.65, 888.72, 959.58, 1036.8, 1121.8, 1213.9, 1314.3, 1423.8, 1543., 1672.8, 1813.4,
01308
01309
                  1966.1, 2131.4, 2309.5, 2499.3, 2705., 2925.7, 3161.6, 3411.3, 3611.5, 3889.2, 4191.1, 4519.3, 4877.9, 5272.9, 5712.9, 6142.7,
01310
                   6719.6, 7385., 8145., 8977.7, 9831.9, 10827., 11934., 13063.,
01312
                   14434., 15878., 17591., 19435., 21510., 23835., 26835., 29740.,
01313
01314
                  32878., 36305., 39830., 43273., 46931., 50499., 49586., 51598.,
                  53429., 54619., 55081., 55102., 54485., 53487., 52042., 42689., 42607., 44020., 47994., 54169., 53916., 55808., 56642., 46049., 44243., 32929., 30658., 21963., 20835., 15962., 13679., 17652.,
01315
01316
                  19680., 22388., 25625., 29184., 32520., 35720., 38414., 40523., 49228., 48173., 45678., 41768., 37600., 41313., 42654., 44465.,
01318
01319
01320
                  55736., 56630., 65409., 63308., 66572., 61845., 60379., 56777.
                  51920., 46601., 41367., 36529., 32219., 28470., 25192., 22362., 19907., 17772., 15907., 14273., 12835., 11567., 10445., 9450.2, 8565.1, 7776., 7070.8, 6439.2, 5872.3, 5362.4, 4903., 4488.3,
01321
01322
01323
                   4113.4, 3773.8, 3465.8, 3186.1, 2931.7, 2700.1, 2488.8, 2296.,
                  2119.8, 1958.6, 1810.9, 1675.6, 1551.4, 1437.3, 1332.4, 1236., 1147.2, 1065.3, 989.86, 920.22, 855.91, 796.48, 741.53, 690.69, 643.62, 600.02, 559.6, 522.13, 487.35, 455.06, 425.08, 397.21, 371.3, 347.2, 324.78, 303.9, 284.46, 266.34, 249.45, 233.7, 219.01, 205.3, 192.5, 180.55, 169.38, 158.95, 149.2, 140.07,
01325
01326
01327
01328
01329
                   131.54, 123.56, 116.09, 109.09, 102.54, 96.405, 90.655, 85.266,
                   80.213, 75.475, 71.031, 66.861, 62.948, 59.275, 55.827, 52.587,
01331
01332
                   49.544, 46.686, 43.998, 41.473, 39.099, 36.867, 34.768, 32.795,
01333
                  30.939, 29.192, 27.546, 25.998, 24.539, 23.164, 21.869, 20.65,
                  19.501, 18.419, 17.399, 16.438, 15.532, 14.678, 13.874, 13.115, 12.4, 11.726, 11.088, 10.488, 9.921, 9.3846, 8.8784, 8.3996, 7.9469, 7.5197, 7.1174, 6.738, 6.379, 6.0409, 5.7213, 5.419,
01334
01335
                  5.1327, 4.8611, 4.6046, 4.3617, 4.1316, 3.9138, 3.7077, 3.5125, 3.3281, 3.1536, 2.9885, 2.8323, 2.6846, 2.5447, 2.4124, 2.2871,
01337
01338
01339
                  2.1686, 2.0564, 1.9501, 1.8495, 1.7543, 1.6641, 1.5787, 1.4978,
                  1.4212, 1.3486, 1.2799, 1.2147, 1.1529, 1.0943, 1.0388, .98602, .93596, .8886, .84352, .80078, .76029, .722, .68585, .65161,
01340
```

```
.61901, .58808, .55854, .53044, .5039, .47853, .45459, .43173,
                     .41008, .38965, .37021, .35186, .33444, .31797, .30234, .28758, .2736, .26036, .24764, .2357, .22431, .21342, .20295, .19288,
01343
01344
01345
                      .18334, .17444, .166, .15815, .15072, .14348, .13674, .13015,
01346
                      .12399, .11807, .11231, .10689, .10164, .096696, .091955,
                     .087476, .083183, .079113, .075229, .071536, .068026, .064698, .06154, .058544, .055699, .052997, .050431, .047993, .045676,
01347
                      .043475, .041382, .039392, .037501, .035702, .033991, .032364,
01349
01350
                      .030817, .029345, .027945, .026613, .025345, .024139, .022991,
01351
                      .021899, .02086, .019871, .018929, .018033, .01718, .016368,
                    .015595, .014859, .014158, .013491, .012856, .012251, .011675, .011126, .010604, .010107, .0096331, .009182, .0087523, .0083431, .0079533, .0075821, .0072284, .0068915, .0065706, .0062649, .0059737, .0056963, .005432, .0051802, .0049404, .0047118,
01352
01353
01354
01355
01356
                     .0044941, .0042867, .0040891, .0039009, .0037216, .0035507
01357
                     .003388, .0032329, .0030852, .0029445, .0028105, .0026829, .0025613, .0024455, .0023353, .0022303, .0021304, .0020353,
01358
                    .0019448, .0018587, .0017767, .0016988, .0016247, .0015543, .0014874, .0014238, .0013635, .0013062, .0012519, .0012005, .0011517, .0011057, .0010621, .001021, 9.8233e-4, 9.4589e-4, 9.1167e-4, 8.7961e-4, 8.4964e-4, 8.2173e-4, 7.9582e-4, 7.7189e-4, 7.499e-4, 7.2983e-4, 7.1167e-4, 6.9542e-4, 6.8108e-4, 6.8666e-4, 6.8666e
01359
01361
01362
01363
                      6.5819e-4, 6.4971e-4, 6.4328e-4, 6.3895e-4, 6.3681e-4, 6.3697e-4,
01364
                     6.3956e-4, 6.4472e-4, 6.5266e-4, 6.6359e-4, 6.778e-4, 6.9563e-4, 7.1749e-4, 7.4392e-4, 7.7556e-4, 8.1028e-4, 8.4994e-4, 8.8709e-4,
01365
01366
                      9.3413e-4, 9.6953e-4, .0010202, .0010738, .0010976, .0011507,
                     .0011686, .0012264, .001291, .0013346, .0014246, .0015293,
01368
01369
                      .0016359, .0017824, .0019255, .0020854, .002247, .0024148,
                      .0026199, .0027523, .0029704, .0030702, .0033047, .0035013,
01370
                     .0037576, .0040275, .0043089, .0046927, .0049307, .0053486, .0053809, .0056699, .0059325, .0055488, .005634, .0056392, .004946, .0048855, .0048208, .0044386, .0045498, .0046377, .004839, .0052396, .0057324, .0060859, .0066906, .0071148,
01371
01372
01374
01375
                     .0077224, .0082687, .008769, .0084471, .008572, .0087729,
01376
                      .008775, .0090742, .0080704, .0080288, .0085747, .0086087
                    .0086408, .0088752, .0089381, .0089757, .0093532, .0092824, .0092566, .0092645, .0092735, .009342, .0095806, .0097991, .010213, .010611, .011129, .011756, .013237, .01412, .015034, .015936, .01682, .018597, .019315, .019995, .020658, .021289,
01377
01378
01380
                     .022363, .022996, .023716, .024512, .025434, .026067, .027118, .028396, .029865, .031442, .033253, .03525, .037296, .039701,
01381
01382
                      .042356, .045154, .048059, .051294, .054893, .058636, .061407,
01383
                     .065172, .068974, .072676, .073379, .076547, .079556, .079134, .082308, .085739, .090192, .09359, .099599, .10669, .11496,
01384
01385
                     .1244, .13512, .14752, .14494, .15647, .1668, .17863, .19029, .20124, .20254, .21179, .21982, .21625, .22364, .23405, .23382,
01387
01388
                      .2434, .25708, .26406, .27621, .28909, .30395, .31717, .33271,
                     .2494, .25706, .26406, .27621, .26909, .30393, .31717, .33271, .3496, .36765, .38774, .40949, .446, .46985, .49846, .5287, .562, .59841, .64598, .66834, .7327, .78978, .8373, .88708, .94744, .10006, 1.0574, 1.1215, 1.1856, 1.2546, 1.3292, 1.4107, 1.4974, 1.5913, 1.6931, 1.8028, 1.9212, 2.0492, 2.1874, 2.3365, 2.4978,
01389
01390
01391
                      2.6718, 2.8588, 3.062, 3.2818, 3.5188, 3.7752, 4.0527, 4.3542,
01393
01394
                      4.6782, 5.0312, 5.4123, 5.8246, 6.2639, 6.7435, 7.2636, 7.8064,
                     8.4091, 9.0696, 9.7677, 10.548, 11.4, 12.309, 13.324, 14.284, 15.445, 16.687, 18.019, 19.403, 20.847, 22.366, 23.925, 25.537,
01395
01396
                     27.213, 28.069, 29.864, 31.829, 33.988, 35.856, 38.829, 42.321, 46.319, 50.606, 55.126, 59.126, 64.162, 68.708, 74.615, 81.176,
01397
                      87.739, 95.494, 103.83, 113.38, 123.99, 135.8, 148.7, 162.58,
01399
                      176.32, 192.6, 211.47, 232.7, 252.64, 277.41, 305.38, 333.44,
01400
01401
                     366.42, 402.66, 442.14, 484.53, 526.42, 568.15, 558.78, 582.6
                     600.98, 613.94, 619.44, 618.24, 609.84, 595.96, 484.86, 475.59,
01402
                     478.49, 501.56, 552.19, 628.44, 630.39, 658.92, 671.96, 562.7, 545.88, 423.43, 400.14, 306.59, 294.13, 246.8, 226.51, 278.21,
01403
                     314.39, 347.22, 389.13, 433.16, 477.48, 521.67, 560.54, 683.6, 696.37, 695.91, 683.1, 658.24, 634.89, 698.85, 742.87, 796.66,
01406
01407
                     954.49, 1009.5, 1150.5, 1179.1, 1267.9, 1272.4, 1312.7, 1330.4,
01408
                     1331.6, 1315.8, 1308.3, 1293.3, 1274.6, 1249.5, 1213.2, 1172.1, 1124.4, 930.33, 893.36, 871.27, 883.54, 940.76, 1036., 1025.6,
01409
                     1053.1, 914.51, 894.15, 865.03, 670.63, 508.41, 475.15, 370.85,
01410
                      361.06, 319.38, 312.75, 331.87, 367.13, 415., 467.94, 525.49,
                     578.41, 624.66, 794.82, 796.97, 780.29, 736.49, 670.18, 603.75, 659.67, 679.8, 857.12, 884.05, 900.65, 1046.1, 1141.9, 1083., 1089.2, 1e3, 947.08, 872.31, 787.91, 704.75, 624.93, 553.68, 489.91, 434.21, 385.64, 343.3, 306.42, 274.18, 245.94, 221.11,
01412
01413
01414
01415
                     199.23, 179.88, 162.73, 147.48, 133.88, 121.73, 110.86, 101.1, 92.323, 84.417, 77.281, 70.831, 64.991, 59.694, 54.884, 50.509, 46.526, 42.893, 39.58, 36.549, 33.776, 31.236, 28.907, 26.77,
01416
01418
01419
                      24.805, 23., 21.339, 19.81, 18.404, 17.105, 15.909, 14.801,
                     13.778, 12.83, 11.954, 11.142, 10.389, 9.691, 9.0434, 8.4423, 7.8842, 7.3657, 6.8838, 6.4357, 6.0189, 5.6308, 5.2696, 4.9332, 4.6198, 4.3277, 4.0553, 3.8012, 3.5639, 3.3424, 3.1355, 2.9422, 2.7614, 2.5924, 2.4343, 2.2864, 2.148, 2.0184, 1.8971, 1.7835,
01420
01421
01422
                      1.677, 1.5773, 1.4838, 1.3961, 1.3139, 1.2369, 1.1645, 1.0966,
01425
                     1.0329, .97309, .91686, .86406, .81439, .76767, .72381, .68252
01426
                      .64359, .60695, .57247, .54008, .50957, .48092, .45401, .42862,
                     .40465, .38202, .36072, .34052, .3216, .30386, .28711, .27135, .25651, .24252, .2293, .21689, .20517, .19416, .18381, .17396,
01427
```

```
.16469
01429
01430
01431
              static double co2230[2001] = { 2.743e-5, 2.8815e-5, 3.027e-5, 3.1798e-5,
01432
                  3.3405e-5, 3.5094e-5, 3.6869e-5, 3.8734e-5, 4.0694e-5, 4.2754e-5, 4.492e-5, 4.7196e-5, 4.9588e-5, 5.2103e-5, 5.4747e-5, 5.7525e-5,
01433
01434
                   6.0446e-5, 6.3516e-5, 6.6744e-5, 7.0137e-5, 7.3704e-5, 7.7455e-5,
                  8.1397e-5, 8.5543e-5, 8.9901e-5, 9.4484e-5, 9.9302e-5, 1.0437e-4,
01436
01437
                  1.097e-4, 1.153e-4, 1.2119e-4, 1.2738e-4, 1.3389e-4, 1.4074e-4,
                  1.4795e-4, 1.5552e-4, 1.6349e-4, 1.7187e-4, 1.8068e-4, 1.8995e-4, 1.997e-4, 2.0996e-4, 2.2075e-4, 2.321e-4, 2.4403e-4, 2.5659e-4, 2.698e-4, 2.837e-4, 2.9832e-4, 3.137e-4, 3.2988e-4, 3.4691e-4,
01438
01439
01440
                  3.6483e-4, 3.8368e-4, 4.0351e-4, 4.2439e-4, 4.4635e-4, 4.6947e-4, 4.9379e-4, 5.1939e-4, 5.4633e-4, 5.7468e-4, 6.0452e-4, 6.3593e-4,
01441
01442
01443
                   6.69e-4, 7.038e-4, 7.4043e-4, 7.79e-4, 8.1959e-4, 8.6233e-4,
                  9.0732e-4, 9.5469e-4, .0010046, .0010571, .0011124, .0011706, .0012319, .0012964, .0013644, .001436, .0015114, .0015908,
01444
01445
                  .0016745, .0017625, .0018553, .0019531, .002056, .0021645, .0022788, .0023992, .002526, .0026596, .0028004, .0029488, .0031052, .0032699, .0034436, .0036265, .0038194, .0040227,
01446
01448
                  .0042369, .0044628, .0047008, .0049518, .0052164, .0054953,
01449
01450
                   .0057894, \ .0060995, \ .0064265, \ .0067713, \ .007135, \ .0075184,
                  .0079228, .0083494, .0087993, .0097738, .0097745, .010303, .01086, .011448, .012068, .012722, .013413, .014142, .014911, .015723, .01658, .017484, .018439, .019447, .020511, .021635, .022821, .024074, .025397, .026794, .02827, .029829, .031475, .033215, .035052, .036994, .039045, .041213, .043504, .045926
01451
01452
01453
01455
                  .033215, .035052, .035094, .039045, .041213, .043504, .045926, .048485, .05119, .05405, .057074, .060271, .063651, .067225, .071006, .075004, .079233, .083708, .088441, .093449, .098749, .10436, .11029, .11657, .12322, .13026, .13772, .14561, .15397, .16282, .1722, .18214, .19266, .20381, .21563, .22816, .24143, .2555, .27043, .28625, .30303, .32082, .3397, .35972, .38097, .40352, .42746, .45286, .47983, .50847, .53888, .57119, .6055,
01/156
01457
01458
01459
01460
01461
01462
                   .64196, .6807, .72187, .76564, .81217, .86165, .91427, .97025,
                  1.0298, 1.0932, 1.1606, 1.2324, 1.3088, 1.3902, 1.477, 1.5693, 1.6678, 1.7727, 1.8845, 2.0038, 2.131, 2.2666, 2.4114, 2.5659, 2.7309, 2.907, 3.0951, 3.2961, 3.5109, 3.7405, 3.986, 4.2485, 4.5293, 4.8299, 5.1516, 5.4961, 5.8651, 6.2605, 6.6842, 7.1385,
01463
01464
01465
01467
                   7.6256, 8.1481, 8.7089, 9.3109, 9.9573, 10.652, 11.398, 12.2,
                  13.063, 13.992, 14.99, 16.064, 17.222, 18.469, 19.813, 21.263, 22.828, 24.516, 26.34, 28.31, 30.437, 32.738, 35.226, 37.914,
01468
01469
                  40.824, 43.974, 47.377, 51.061, 55.011, 59.299, 63.961, 69.013, 74.492, 80.444, 86.919, 93.836, 101.23, 109.25, 117.98, 127.47, 137.81, 149.07, 161.35, 174.75, 189.42, 205.49, 223.02, 242.26,
01470
01471
01472
                  263.45, 286.75, 311.94, 340.01, 370.86, 404.92, 440.44, 480.27, 525.17, 574.71, 626.22, 686.8, 754.38, 827.07, 913.38, 1011.7,
01/17/
01475
                  1121.5, 1161.6, 1289.5, 1432.2, 1595.4, 1777., 1983.3, 2216.1,
                  2485.7, 2788.3, 3101.5, 3481., 3902.1, 4257.1, 4740., 5272.8, 5457.9, 5946.2, 6505.3, 6668.4, 7302.4, 8061.6, 9015.8, 9908.3, 11613., 13956., 3249.6, 3243., 2901.5, 2841.3, 2729.6, 2558.2, 1797.8, 1583.2, 1386., 1233.5, 787.74, 701.46, 761.66, 767.21,
01476
01477
01478
                   722.83, 1180.6, 1332.1, 1461.6, 2032.9, 2166., 2255.9, 2294.7,
01480
01481
                  2587.2, 2396.5, 2122.4, 12553., 10784., 9832.5, 8827.3, 8029.1,
                  7377.9, 7347.1, 6783.8, 6239.1, 5721.1, 5503., 4975.1, 4477.8, 4021.3, 3676.8, 3275.3, 2914.9, 2597.4, 2328.2, 2075.4, 1857.6,
01482
01483
                  1663.6, 1493.3, 1343.8, 1213.3, 1095.6, 1066.5, 958.91, 865.15, 783.31, 714.35, 650.77, 593.98, 546.2, 499.9, 457.87, 421.75, 387.61, 355.25, 326.62, 299.7, 275.21, 253.17, 232.83, 214.31,
01484
01486
                   197.5, 182.08, 167.98, 155.12, 143.32, 132.5, 122.58, 113.48,
01487
                  105.11, 97.415, 90.182, 83.463, 77.281, 71.587, 66.341, 61.493, 57.014, 53.062, 49.21, 45.663, 42.38, 39.348, 36.547, 33.967, 31.573, 29.357, 27.314, 25.415, 23.658, 22.03, 20.524, 19.125, 17.829, 16.627, 15.511, 14.476, 13.514, 12.618, 11.786, 11.013,
01488
01489
01490
                   10.294, 9.6246, 9.0018, 8.4218, 7.8816, 7.3783, 6.9092, 6.4719,
01492
01493
                   6.0641, 5.6838, 5.3289, 4.998, 4.6893, 4.4014, 4.1325, 3.8813,
01494
                  3.6469, 3.4283, 3.2241, 3.035, 2.8576, 2.6922, 2.5348, 2.3896,
01495
                  2.2535, 2.1258, 2.0059, 1.8929, 1.7862, 1.6854, 1.5898, 1.4992, 1.4017, 1.3218, 1.2479, 1.1809, 1.1215, 1.0693, 1.0116, .96016,
01496
                  .9105, .84859, .80105, .74381, .69982, .65127, .60899, .57843, .54592, .51792, .49336, .47155, .45201, .43426, .41807, .40303,
01497
                  .38876, .3863, .37098, .35492, .33801, .32032, .30341, .29874, .29193, .28689, .29584, .29155, .29826, .29195, .29287, .2904,
01499
01500
01501
                   .28199, .27709, .27162, .26622, .26133, .25676, .25235, .23137,
                  .22365, .21519, .20597, .19636, .18699, .16485, .16262, .16643,
01502
                  .17542, .18198, .18631, .16759, .16338, .13505, .1267, .10053, .092554, .074093, .062159, .055523, .054849, .05401, .05528, .058982, .07952, .08647, .093244, .099285, .10393, .10661,
01503
01505
                  12072, 11417, 10396, 1093265, 1089137, 1088909, 119002, 11277, 13625, 13565, 14907, 14167, 1428, 13744, 12768, 11382, 10244, 091686, 08109, 071739, 063616, 056579, 050504, 045251, 040689, 036715, 033237, 030181, 027488, 025107, 022988, 021125, 01946, 017979, 016661, 015489,
01506
01507
01508
01509
                  .014448, .013526, .012712, .011998, .011375, .010839,
01511
01512
                   .010007, .0097053, .0094783, .0093257, .0092489, .0092504,
01513
                   .0093346, .0095077, .0097676, .01012, .01058, .011157, .011844,
                  .012672, .013665, .014766, .015999, .017509, .018972, .020444, .022311, .023742, .0249, .025599, .026981, .026462, .025143,
01514
01515
```

```
.025066, .022814, .020458, .020026, .019142, .020189, .022371,
                .024163, .023728, .02199, .019506, .018591, .015576, .012784, .011744, .0094777, .0079148, .0070652, .006986, .0071758,
01517
01518
                 .008086, .0098025, .01087, .013609, .016764, .018137, .021061,
01519
                .023498, .023576, .023965, .022828, .021519, .021283, .023364,
01520
                .026457, .029782, .030856, .033486, .035515, .035543, .036558, .037198, .037472, .037045, .037284, .03777, .038085, .038366, .038526, .038282, .038915, .037697, .035667, .032941, .031959,
01523
                .028692, .025918, .024596, .025592, .027873, .028935, .02984, .028148, .025305, .021912, .020454, .016732, .013357, .01205, .009731, .0079881, .0077704, .0074387, .0083895, .0096776,
01524
01525
01526
                .009/31, .0079881, .0077704, .0074387, .0083895, .0096776, .010326, .01293, .015955, .019247, .020145, .02267, .024231, .024184, .022131, .019784, .01955, .01971, .022119, .025116, .027978, .028107, .029808, .030701, .029164, .028551, .027286, .024946, .023259, .020982, .019221, .017471, .015643, .014074,
01527
01529
01530
01531
                .01261, .011301, .010116, .0090582, .0081036, .0072542, .0065034, .0058436, .0052571, .0047321, .0042697, .0038607, .0034977,
01532
                .0031747, .0028864, .0026284, .002397, .002189, .0020017, .0018326, .0016798, .0015414, .0014159, .0013019, .0011983,
01533
                 .0011039, .0010177, 9.391e-4, 8.6717e-4, 8.0131e-4, 7.4093e-4,
                6.8553e-4, 6.3464e-4, 5.8787e-4, 5.4487e-4, 5.0533e-4, 4.69e-4, 4.3556e-4, 4.0474e-4, 3.7629e-4, 3.5e-4, 3.2569e-4, 3.032e-4,
01536
01537
                2.8239e-4, 2.6314e-4, 2.4535e-4, 2.2891e-4, 2.1374e-4, 1.9975e-4,
01538
                1.8685e-4, 1.7498e-4, 1.6406e-4, 1.5401e-4, 1.4479e-4, 1.3633e-4, 1.2858e-4, 1.2148e-4, 1.1499e-4, 1.0907e-4, 1.0369e-4, 9.8791e-5,
01539
                 9.4359e-5, 9.0359e-5, 8.6766e-5, 8.3555e-5, 8.0703e-5, 7.8192e-5,
                7.6003e-5, 7.4119e-5, 7.2528e-5, 7.1216e-5, 7.0171e-5, 6.9385e-5,
01542
                6.8848e-5, 6.8554e-5, 6.8496e-5, 6.8669e-5, 6.9069e-5, 6.9694e-5, 7.054e-5, 7.1608e-5, 7.2896e-5, 7.4406e-5, 7.6139e-5, 7.8097e-5, 8.0283e-5, 8.2702e-5, 8.5357e-5, 8.8255e-5, 9.1402e-5, 9.4806e-5,
015/3
01544
01545
01546
                9.8473e-5, 1.0241e-4, 1.0664e-4, 1.1115e-4, 1.1598e-4, 1.2112e-4,
                1.2659e-4, 1.3241e-4, 1.3859e-4, 1.4515e-4, 1.521e-4, 1.5947e-4,
                1.6728e-4, 1.7555e-4, 1.8429e-4, 1.9355e-4, 2.0334e-4, 2.1369e-4,
01548
01549
                2.2463e-4, 2.3619e-4, 2.4841e-4, 2.6132e-4, 2.7497e-4, 2.8938e-4,
                3.0462e-4, 3.2071e-4, 3.3771e-4, 3.5567e-4, 3.7465e-4, 3.947e-4, 4.1588e-4, 4.3828e-4, 4.6194e-4, 4.8695e-4, 5.1338e-4, 5.4133e-4, 5.7087e-4, 6.0211e-4, 6.3515e-4, 6.701e-4, 7.0706e-4, 7.4617e-4, 7.8756e-4, 8.3136e-4, 8.7772e-4, 9.2681e-4, 9.788e-4, .0010339,
01550
01551
01552
01554
                .0010922, .001154, .0012195, .0012889, .0013626, .0014407,
                .0015235, .0016114, .0017048, .0018038, .001909, .0020207,
01555
01556
                 .0021395, .0022657, .0023998, .0025426, .0026944, .002856,
                 .0030281, \ .0032114, \ .0034068, \ .003615, \ .0038371, \ .004074,
01557
                .004327, .0045971, .0048857, .0051942, .0055239, .0058766, .0062538, .0066573, .0070891, .007551, .0080455, .0085747, .0091412, .0097481, .010397, .011092, .011837, .012638, .013495, .014415, .01541, .016475, .017621, .018857, .020175, .02162,
01558
01560
01561
                .023185, .024876, .02672, .028732, .030916, .033319, .035939, .038736, .041847, .04524, .048715, .052678, .056977, .061203,
01562
01563
                .066184, .07164, .076952, .083477, .090674, .098049, .10697, .1169, .1277, .14011, .15323, .1684, .18601, .20626, .22831, .25417, .28407, .31405, .34957, .38823, .41923, .46026, .50409,
01564
01565
                .51227, .54805, .57976, .53818, .55056, .557, .46741, .46403,
01567
01568
                 .4636, .42265, .45166, .49852, .56663, .34306, .17779, .17697
                . 18346, .19129, .20014, .21778, .23604, .25649, .26676, .31238, .33856, .39998, .4288, .46568, .56654, .60786, .64473, .76466, .7897, .80778, .86443, .85736, .84798, .84157, 1.1385, 1.2446, 1.1923, 1.1552, 1.1338, 1.1266, 1.1292, 1.1431, 1.1683, 1.2059,
01569
01570
01571
                1.2521, 1.3069, 1.3712, 1.4471, 1.5275, 1.6165, 1.7145, 1.8189,
01573
                1.2321, 1.3069, 1.3712, 1.4471, 1.3273, 1.6163, 1.7143, 1.6169, 1.9359, 2.065, 2.2007, 2.3591, 2.5362, 2.7346, 2.9515, 3.2021, 3.4851, 3.7935, 4.0694, 4.4463, 4.807, 5.2443, 5.7178, 6.2231, 6.4796, 6.9461, 7.4099, 7.3652, 7.7182, 8.048, 7.7373, 8.0363, 8.3855, 8.8044, 9.0257, 9.8574, 10.948, 10.563, 6.8979, 7.0744, 7.4121, 7.7663, 8.1768, 8.6243, 9.1437, 9.7847, 10.182, 10.849,
01574
01575
01576
01577
                 11.572, 12.602, 13.482, 14.431, 15.907, 16.983, 18.11, 19.884,
01580
                21.02, 22.18, 23.355, 24.848, 25.954, 27.13, 30.186, 34.893,
01581
                35.682, 36.755, 38.111, 39.703, 41.58, 43.606, 45.868, 48.573,
01582
                51.298, 54.291, 57.559, 61.116, 64.964, 69.124, 73.628, 78.471, 83.683, 89.307, 95.341, 101.84, 108.83, 116.36, 124.46, 133.18,
01583
                142.57, 152.79, 163.69, 175.43, 188.11, 201.79, 216.55, 232.51,
01584
                249.74, 268.38, 288.54, 310.35, 333.97, 359.55, 387.26, 417.3, 449.88, 485.2, 523.54, 565.14, 610.28, 659.31, 712.56, 770.43,
01586
01587
                833.36, 901.82, 976.36, 1057.6, 1146.8, 1243.8, 1350., 1466.3,
01588
                1593.6, 1732.7, 1884.1, 2049.1, 2228.2, 2421.9, 2629.4, 2853.7,
                3094.4, 3351.1, 3622.3, 3829.8, 4123.1, 4438.3, 4777.2, 5144.1,
01589
                5545.4, 5990.5, 6404.5, 6996.8, 7687.6, 8482.9, 9349.4, 10203.,
01590
                11223., 12358., 13493., 14916., 16416., 18236., 20222., 22501.,
01591
                25102., 28358., 31707., 35404., 39538., 43911., 48391., 53193.,
01592
01593
                58028., 58082., 61276., 64193., 66294., 67480., 67921., 67423.,
01594
                66254., 64341., 51737., 51420., 53072., 58145., 66195., 65358.,
                67377., 67869., 53509., 50553., 35737., 32425., 21704., 19974.,
01595
                14457., 12142., 16798., 19489., 23049., 27270., 31910., 36457., 40877., 44748., 47876., 59793., 58626., 55454., 50337., 44893.,
01596
                50228., 52216., 54747., 69541., 70455., 81014., 77694., 80533., 73953., 70927., 65539., 59002., 52281., 45953., 40292., 35360.,
01598
01599
                31124., 27478., 24346., 21647., 19308., 17271., 15491., 13927., 12550., 11331., 10250., 9288.8, 8431.4, 7664.9, 6978.3, 6361.8, 5807.4, 5307.7, 4856.8, 4449., 4079.8, 3744.9, 3440.8, 3164.2,
01600
01601
01602
```

```
2912.3, 2682.7, 2473., 2281.4, 2106., 1945.3, 1797.9, 1662.5,
                 1538.1, 1423.6, 1318.1, 1221., 1131.5, 1049., 972.99, 902.87, 838.01, 777.95, 722.2, 670.44, 622.35, 577.68, 536.21, 497.76,
01604
01605
01606
                 462.12, 429.13, 398.61, 370.39, 344.29, 320.16, 297.85, 277.2,
                 258.08, 240.38, 223.97, 208.77, 194.66, 181.58, 169.43, 158.15, 147.67, 137.92, 128.86, 120.44, 112.6, 105.3, 98.499, 92.166,
01607
01608
                 86.264, 80.763, 75.632, 70.846, 66.381, 62.213, 58.321, 54.685,
                 51.288, 48.114, 45.145, 42.368, 39.772, 37.341, 35.065, 32.937,
01610
01611
                 30.943, 29.077, 27.33, 25.693, 24.158, 22.717, 21.367, 20.099,
                18.909, 17.792, 16.744, 15.761, 14.838, 13.971, 13.157, 12.393, 11.676, 11.003, 10.369, 9.775, 9.2165, 8.6902, 8.1963, 7.7314, 7.2923, 6.8794, 6.4898, 6.122, 5.7764, 5.4525, 5.1484, 4.8611,
01612
01613
01614
                 4.5918, 4.3379, 4.0982, 3.8716, 3.6567, 3.4545, 3.2634, 3.0828,
01615
                 2.9122, 2.7512, 2.5993, 2.4561, 2.3211, 2.1938, 2.0737, 1.9603,
01616
01617
                 1.8534, 1.7525, 1.6572, 1.5673, 1.4824, 1.4022, 1.3265, 1.2551,
                1.8534, 1.7253, 1.0572, 1.3673, 1.4624, 1.4022, 1.3263, 1.2531, 1.1876, 1.1239, 1.0637, 1.0069, .9532, .90248, .85454, .80921, .76631, .72569, .6872, .65072, .61635, .5836, .55261, .52336, .49581, .46998, .44559, .42236, .40036, .37929, .35924, .34043, .32238, .30547, .28931, .27405, .25975, .24616, .23341, .22133, .20997, .19924, .18917, .17967, .17075, .16211, .15411, .14646, .13912, .13201, .12509, .11857, .11261, .10698, .10186, .097039,
01618
01619
01620
01621
01622
01623
                 .092236, .087844, .083443, .07938, .075452, .071564, .067931,
01624
                 .064389, .061078, .057901, .054921, .052061, .049364, .046789,
01625
                .04435, .042044, .039866, .037808, .035863, .034023, .032282, .030634, .029073, .027595, .026194, .024866, .023608, .022415, .021283, .02021, .019193, .018228, .017312, .016443, .015619, .014837, .014094, .01339, .012721, .012086, .011483, .010911,
01626
01627
01628
01629
                .010368, .009852, .0093623, .0088972, .0084556, .0080362, .0076379, .0072596, .0069003, .006559, .0062349, .0059269, .0056344, .0053565, .0050925, .0048417, .0046034, .004377, .0041618, .0039575, .0037633, .0035788, .0034034, .0032368,
01630
01631
01632
01633
                 .0030785, .002928, .0027851, .0026492, .0025201, .0023975, .0022809, .0021701, .0020649, .0019649, .0018699, .0017796, .0016938, .0016122, .0015348, .0014612, .0013913, .001325,
01635
01636
                 .0012619, .0012021, .0011452, .0010913, .0010401, 9.9149e-4, 9.454e-4, 9.0169e-4, 8.6024e-4, 8.2097e-4, 7.8377e-4, 7.4854e-4, 7.1522e-4, 6.8371e-4, 6.5393e-4, 6.2582e-4, 5.9932e-4, 5.7435e-4,
01637
01638
01639
                 5.5087e-4, 5.2882e-4, 5.0814e-4, 4.8881e-4, 4.7076e-4, 4.5398e-4,
01641
                 4.3843e-4, 4.2407e-4, 4.109e-4, 3.9888e-4, 3.88e-4, 3.7826e-4,
                 3.6963e-4, 3.6213e-4, 3.5575e-4, 3.505e-4, 3.464e-4, 3.4346e-4
01642
01643
                 3.4173e-4, 3.4125e-4, 3.4206e-4, 3.4424e-4, 3.4787e-4, 3.5303e-4,
                 3.5986e-4, 3.6847e-4, 3.7903e-4, 3.9174e-4, 4.0681e-4, 4.2455e-4,
01644
                 4.4527e-4, 4.6942e-4, 4.9637e-4, 5.2698e-4, 5.5808e-4, 5.9514e-4, 6.2757e-4, 6.689e-4, 7.1298e-4, 7.3955e-4, 7.8403e-4, 8.0449e-4, 8.5131e-4, 9.0256e-4, 9.3692e-4, .0010051, .0010846, .0011678,
01645
01646
01647
01648
                 .001282, .0014016, .0015355, .0016764, .0018272, .0020055,
01649
                 .0021455, .0023421, .0024615, .0026786, .0028787, .0031259,
                01650
01651
01652
01654
01655
                 .0076269, .0063758, .006254, .0067749, .0067909, .0068231,
                .0072143, .0072762, .0072954, .007679, .0075107, .0073658, .0072441, .0071074, .0070378, .007176, .0072472, .0075844, .0079291, .008412, .0090165, .010688, .011535, .012375, .013166, .013895, .015567, .016011, .016392, .016737, .017043, .017731, .018031, .018419, .018877, .019474, .019868, .020604, .021538,
01656
01657
01658
01660
01661
                 .022653, .023869, .025288, .026879, .028547, .030524, .03274,
                .035132, .03769, .040567, .043793, .047188, .049962, .053542, .057205, .060776, .061489, .064419, .067124, .065945, .068487, .071209, .074783, .077039, .082444, .08902, .09692, .10617,
01662
01663
01664
                .11687, .12952, .12362, .13498, .14412, .15492, .16519, .1744, .17096, .17714, .18208, .17363, .17813, .18564, .18295, .19045, .20252, .20815, .21844, .22929, .24229, .25321, .26588, .2797,
01666
01667
01668
                 .29465, .31136, .32961, .36529, .38486, .41027, .43694, .4667,
01669
                 .49943, .54542, .58348, .62303, .67633, .71755, .76054, .81371, .85934, .90841, .96438, 1.0207, 1.0821, 1.1491, 1.2226, 1.3018,
01670
                 1.388, 1.4818, 1.5835, 1.6939, 1.8137, 1.9435, 2.0843, 2.237,
01671
                 2.4026, 2.5818, 2.7767, 2.9885, 3.2182, 3.4679, 3.7391, 4.0349,
                 4.3554, 4.7053, 5.0849, 5.4986, 5.9436, 6.4294, 6.9598, 7.5203,
01673
01674
                 8.143, 8.8253, 9.5568, 10.371, 11.267, 12.233, 13.31, 14.357,
                15.598, 16.93, 18.358, 19.849, 21.408, 23.04, 24.706, 26.409, 28.153, 28.795, 30.549, 32.43, 34.49, 36.027, 38.955, 42.465, 46.565, 50.875, 55.378, 59.002, 63.882, 67.949, 73.693, 80.095, 86.403, 94.264, 102.65, 112.37, 123.3, 135.54, 149.14, 163.83,
01675
01676
01677
                 179.17, 196.89, 217.91, 240.94, 264.13, 292.39, 324.83, 358.21,
01679
01680
                 397.16, 440.5, 488.6, 541.04, 595.3, 650.43, 652.03, 688.74,
                719.47, 743.54, 757.68, 762.35, 756.43, 741.42, 595.43, 580.97, 580.83, 605.68, 667.88, 764.49, 759.93, 789.12, 798.17, 645.66, 615.65, 455.05, 421.09, 306.45, 289.14, 235.7, 215.52, 274.57, 316.53, 357.73, 409.89, 465.06, 521.84, 579.02, 630.64, 794.46,
01681
01682
01683
                 813., 813.56, 796.25, 761.57, 727.97, 812.14, 866.75, 932.5,
01685
01686
                 1132.8, 1194.8, 1362.2, 1387.2, 1482.3, 1479.7, 1517.9, 1533.1,
01687
                 1534.2, 1523.3, 1522.5, 1515.5, 1505.2, 1486.5, 1454., 1412.,
                 1358.8, 1107.8, 1060.9, 1033.5, 1048.2, 1122.4, 1248.9, 1227.1, 1255.4, 1058.9, 1020.7, 970.59, 715.24, 512.56, 468.47, 349.3,
01688
01689
```

```
338.26, 299.22, 301.26, 332.38, 382.08, 445.49, 515.87, 590.85,
             662.3, 726.05, 955.59, 964.11, 945.17, 891.48, 807.11, 720.9, 803.36, 834.46, 1073.9, 1107.1, 1123.6, 1296., 1393.7, 1303.1,
01692
             1284.3, 1161.8, 1078.8, 976.13, 868.72, 767.4, 674.72, 593.73,
01693
             523.12, 462.24, 409.75, 364.34, 325., 290.73, 260.76, 234.46, 211.28, 190.78, 172.61, 156.44, 142.01, 129.12, 117.57, 107.2, 97.877, 89.47, 81.882, 75.021, 68.807, 63.171, 58.052, 53.396,
01694
01695
01696
             49.155, 45.288, 41.759, 38.531, 35.576, 32.868, 30.384, 28.102,
01697
01698
             26.003, 24.071, 22.293, 20.655, 19.147, 17.756, 16.476, 15.292,
             14.198, 13.183, 12.241, 11.367, 10.554, 9.7989, 9.0978, 8.4475, 7.845, 7.2868, 6.7704, 6.2927, 5.8508, 5.4421, 5.064, 4.714, 4.3902, 4.0902, 3.8121, 3.5543, 3.315, 3.093, 2.8869, 2.6953, 2.5172, 2.3517, 2.1977, 2.0544, 1.9211, 1.7969, 1.6812, 1.5735,
01699
01700
01701
01702
01703
             1.4731, 1.3794, 1.2921, 1.2107, 1.1346, 1.0637, .99744, .93554,
01704
             .87771, .82368, .77313, .72587, .6816, .64014, .60134, .565,
             .53086, .49883, .46881, .44074, .4144, .38979, .36679, .34513, .32474, .30552, .28751, .27045, .25458, .23976, .22584, .21278,
01705
01706
01707
             .20051, .18899, .17815, .16801, .15846, .14954, .14117, .13328,
01708
             .12584
01709
01710
01711
          double xw, dw, ew, cw296, cw260, cw230, dt230, dt260, dt296, ctw, ctmpth;
01712
01713
          int iw:
01714
01715
           /* Get CO2 continuum absorption... */
01716
          xw = nu / 2 + 1;
01717
          if (xw >= 1 && xw < 2001) {
01718
            iw = (int) xw;
            dw = xw - iw;

ew = 1 - dw;
01719
01720
            cw296 = ew * co2296[iw - 1] + dw * co2296[iw];

cw260 = ew * co2260[iw - 1] + dw * co2260[iw];

cw230 = ew * co2230[iw - 1] + dw * co2230[iw];
01721
01722
01723
01724
             dt230 = t - 230;
             dt260 = t - 260;
01725
             dt296 = t - 296;
01726
            ctw = dt260 * 5.050505e-4 * dt296 * cw230 - dt230 * 9.259259e-4
  * dt296 * cw260 + dt230 * 4.208754e-4 * dt260 * cw296;
01728
01729
             ctmpth = u / NA / 1000 * p / P0 * ctw;
01730
          } else
            ctmpth = 0;
01731
01732
          return ctmpth;
01733 }
01734
01736
01737 double ctmh2o(
01738
          double nu.
01739
          double p.
01740
          double t,
01741
          double q,
01742
          double u) {
01743
01744
          static double h2o296[2001] = { .17, .1695, .172, .168, .1687, .1624, .1606,
            .1508, .1447, .1344, .1214, .1133, .1009, .09217, .08297, .06989, .06513, .05469, .05056, .04417, .03779, .03484, .02994, .0272, .02325, .02063, .01818, .01592, .01405, .01251, .0108, .009647,
01745
01746
01747
             .008424, .007519, .006555, .00588, .005136, .004511, .003989, .003509, .003114, .00274, .002446, .002144, .001895, .001676,
01748
01749
             .001486, .001312, .001164, .001031, 9.129e-4, 8.106e-4, 7.213e-4, 6.4e-4, 5.687e-4, 5.063e-4, 4.511e-4, 4.029e-4, 3.596e-4,
01750
01751
             3.22e-4, 2.889e-4, 2.597e-4, 2.337e-4, 2.108e-4, 1.907e-4,
             1.728e-4, 1.57e-4, 1.43e-4, 1.305e-4, 1.195e-4, 1.097e-4,
01753
01754
             1.009e-4, 9.307e-5, 8.604e-5, 7.971e-5, 7.407e-5, 6.896e-5,
01755
             6.433e-5, 6.013e-5, 5.631e-5, 5.283e-5, 4.963e-5, 4.669e-5,
01756
             4.398e-5, 4.148e-5, 3.917e-5, 3.702e-5, 3.502e-5, 3.316e-5,
             3.142e-5, 2.978e-5, 2.825e-5, 2.681e-5, 2.546e-5, 2.419e-5,
01757
01758
             2.299e-5, 2.186e-5, 2.079e-5, 1.979e-5, 1.884e-5, 1.795e-5,
             1.711e-5, 1.633e-5, 1.559e-5, 1.49e-5, 1.426e-5, 1.367e-5,
01760
             1.312e-5, 1.263e-5, 1.218e-5, 1.178e-5, 1.143e-5, 1.112e-5,
01761
             1.088e-5, 1.07e-5, 1.057e-5, 1.05e-5, 1.051e-5, 1.059e-5,
            1.076e-5, 1.1e-5, 1.133e-5, 1.18e-5, 1.237e-5, 1.308e-5, 1.393e-5, 1.483e-5, 1.614e-5, 1.758e-5, 1.93e-5, 2.123e-5, 2.346e-5, 2.647e-5, 2.93e-5, 3.279e-5, 3.745e-5, 4.152e-5, 4.813e-5, 5.477e-5, 6.203e-5, 7.331e-5, 8.056e-5, 9.882e-5,
01762
01763
01764
01765
             1.05e-4, 1.21e-4, 1.341e-4, 1.572e-4, 1.698e-4, 1.968e-4,
01766
01767
             2.175e-4, 2.431e-4, 2.735e-4, 2.867e-4, 3.19e-4, 3.371e-4
01768
             3.554e-4, 3.726e-4, 3.837e-4, 3.878e-4, 3.864e-4, 3.858e-4,
             3.841e-4, 3.852e-4, 3.815e-4, 3.762e-4, 3.618e-4, 3.579e-4,
01769
             3.45e-4, 3.202e-4, 3.018e-4, 2.785e-4, 2.602e-4, 2.416e-4,
01770
             2.097e-4, 1.939e-4, 1.689e-4, 1.498e-4, 1.308e-4, 1.17e-4,
01771
01772
             1.011e-4, 9.237e-5, 7.909e-5, 7.006e-5, 6.112e-5, 5.401e-5,
01773
             4.914e-5, 4.266e-5, 3.963e-5, 3.316e-5, 3.037e-5, 2.598e-5,
             2.294e-5, 2.066e-5, 1.813e-5, 1.583e-5, 1.423e-5, 1.247e-5, 1.116e-5, 9.76e-6, 8.596e-6, 7.72e-6, 6.825e-6, 6.108e-6, 5.366e-6, 4.733e-6, 4.229e-6, 3.731e-6, 3.346e-6, 2.972e-6,
01774
01775
01776
```

```
2.628e-6, 2.356e-6, 2.102e-6, 1.878e-6, 1.678e-6, 1.507e-6,
             1.348e-6, 1.21e-6, 1.089e-6, 9.806e-7, 8.857e-7, 8.004e-7, 7.261e-7, 6.599e-7, 6.005e-7, 5.479e-7, 5.011e-7, 4.595e-7,
01778
01779
             4.219e-7, 3.885e-7, 3.583e-7, 3.314e-7, 3.071e-7, 2.852e-7,
01780
             2.654e-7, 2.474e-7, 2.311e-7, 2.162e-7, 2.026e-7, 1.902e-7,
01781
             1.788e-7, 1.683e-7, 1.587e-7, 1.497e-7, 1.415e-7,
01782
                                                                                 1.338e-7.
             1.266e-7, 1.2e-7, 1.138e-7, 1.08e-7, 1.027e-7, 9.764e-8,
01783
01784
             9.296e-8, 8.862e-8, 8.458e-8, 8.087e-8, 7.744e-8, 7.429e-8,
01785
             7.145e-8, 6.893e-8, 6.664e-8, 6.468e-8, 6.322e-8, 6.162e-8,
             6.07e-8, 5.992e-8, 5.913e-8, 5.841e-8, 5.796e-8, 5.757e-8, 5.746e-8, 5.731e-8, 5.679e-8, 5.577e-8, 5.671e-8, 5.656e-8,
01786
01787
             5.594e-8, 5.593e-8, 5.602e-8, 5.62e-8, 5.693e-8, 5.725e-8,
01788
             5.858e-8, 6.037e-8, 6.249e-8, 6.535e-8, 6.899e-8, 7.356e-8,
             7.918e-8, 8.618e-8, 9.385e-8, 1.039e-7, 1.158e-7, 1.29e-7,
01790
             1.437e-7, 1.65e-7, 1.871e-7, 2.121e-7, 2.427e-7, 2.773e-7, 3.247e-7, 3.677e-7, 4.037e-7, 4.776e-7, 5.101e-7, 6.214e-7, 6.936e-7, 7.581e-7, 8.486e-7, 9.355e-7, 9.942e-7, 1.063e-6,
01791
01792
01793
             1.123e-6, 1.191e-6, 1.215e-6, 1.247e-6, 1.26e-6, 1.271e-6,
01794
             1.284e-6, 1.317e-6, 1.323e-6, 1.349e-6, 1.353e-6, 1.362e-6,
01796
             1.344e-6, 1.329e-6, 1.336e-6, 1.327e-6, 1.325e-6, 1.359e-6,
             1.374e-6, 1.415e-6, 1.462e-6, 1.526e-6, 1.619e-6, 1.735e-6,
01797
01798
             1.863e-6, 2.034e-6, 2.265e-6, 2.482e-6, 2.756e-6, 3.103e-6,
             3.466e-6, 3.832e-6, 4.378e-6, 4.913e-6, 5.651e-6, 6.311e-6,
01799
             7.169e-6, 8.057e-6, 9.253e-6, 1.047e-5, 1.212e-5, 1.36e-5, 1.569e-5, 1.776e-5, 2.02e-5, 2.281e-5, 2.683e-5, 2.994e-5,
01800
01801
             3.488e-5, 3.896e-5, 4.499e-5, 5.175e-5, 6.035e-5, 6.34e-5,
01803
             7.281e-5, 7.923e-5, 8.348e-5, 9.631e-5, 1.044e-4, 1.102e-4,
01804
             1.176e-4, 1.244e-4, 1.283e-4, 1.326e-4, 1.4e-4, 1.395e-4,
             1.387e-4, 1.363e-4, 1.314e-4, 1.241e-4, 1.228e-4, 1.148e-4, 1.086e-4, 1.018e-4, 8.89e-5, 8.316e-5, 7.292e-5, 6.452e-5, 5.625e-5, 5.045e-5, 4.38e-5, 3.762e-5, 3.29e-5, 2.836e-5,
01805
01806
01807
01808
             2.485e-5, 2.168e-5, 1.895e-5, 1.659e-5, 1.453e-5, 1.282e-5,
             1.132e-5, 1.001e-5, 8.836e-6, 7.804e-6, 6.922e-6, 6.116e-6,
01809
01810
             5.429e-6, 4.824e-6, 4.278e-6, 3.788e-6, 3.371e-6, 2.985e-6,
             2.649e-6, 2.357e-6, 2.09e-6, 1.858e-6, 1.647e-6, 1.462e-6, 1.299e-6, 1.155e-6, 1.028e-6, 9.142e-7, 8.132e-7, 7.246e-7, 6.451e-7, 5.764e-7, 5.151e-7, 4.603e-7, 4.121e-7, 3.694e-7,
01811
01812
01813
             3.318e-7, 2.985e-7, 2.69e-7, 2.428e-7, 2.197e-7, 1.992e-7, 1.81e-7, 1.649e-7, 1.506e-7, 1.378e-7, 1.265e-7, 1.163e-7,
01815
             1.073e-7, 9.918e-8, 9.191e-8, 8.538e-8, 7.949e-8, 7.419e-8,
01816
             6.94e-8, 6.508e-8, 6.114e-8, 5.761e-8, 5.437e-8, 5.146e-8,
01817
             4.89e-8, 4.636e-8, 4.406e-8, 4.201e-8, 4.015e-8, 3.84e-8,
01818
             3.661e-8, 3.51e-8, 3.377e-8, 3.242e-8, 3.13e-8, 3.015e-6, 2.918e-8, 2.83e-8, 2.758e-8, 2.707e-8, 2.656e-8, 2.619e-8,
01819
01820
             2.609e-8, 2.615e-8, 2.63e-8, 2.675e-8, 2.745e-8, 2.842e-8,
01821
01822
             2.966e-8, 3.125e-8, 3.318e-8, 3.565e-8, 3.85e-8,
01823
             4.59e-8, 5.059e-8, 5.607e-8, 6.239e-8, 6.958e-8, 7.796e-8,
             8.773e-8, 9.88e-8, 1.114e-7, 1.258e-7, 1.422e-7, 1.61e-7, 1.822e-7, 2.06e-7, 2.337e-7, 2.645e-7, 2.996e-7, 3.393e-7, 3.843e-7, 4.363e-7, 4.935e-7, 5.607e-7, 6.363e-7, 7.242e-7, 8.23e-7, 9.411e-7, 1.071e-6, 1.232e-6, 1.402e-6, 1.6e-6, 1.82e-6,
01824
01825
01826
             2.128e-6, 2.386e-6, 2.781e-6, 3.242e-6, 3.653e-6, 4.323e-6,
01828
01829
             4.747e-6, 5.321e-6, 5.919e-6, 6.681e-6, 7.101e-6, 7.983e-6,
01830
             8.342e-6, 8.741e-6, 9.431e-6, 9.952e-6, 1.026e-5, 1.055e-5,
             1.095e-5, 1.095e-5, 1.087e-5, 1.056e-5, 1.026e-5, 9.715e-6,
01831
             9.252e-6, 8.452e-6, 7.958e-6, 7.268e-6, 6.295e-6, 6.003e-6, 5e-6,
01832
             4.591e-6, 3.983e-6, 3.479e-6, 3.058e-6, 2.667e-6, 2.293e-6,
             1.995e-6, 1.747e-6, 1.517e-6, 1.335e-6, 1.165e-6, 1.028e-6,
01834
             9.007e-7, 7.956e-7, 7.015e-7, 6.192e-7, 5.491e-7, 4.859e-7
01835
             4.297e-7, 3.799e-7, 3.38e-7, 3.002e-7, 2.659e-7, 2.366e-7, 2.103e-7, 1.861e-7, 1.655e-7, 1.469e-7, 1.309e-7, 1.162e-7, 1.032e-7, 9.198e-8, 8.181e-8, 7.294e-8, 6.516e-8, 5.787e-8,
01836
01837
01838
             5.163e-8, 4.612e-8, 4.119e-8, 3.695e-8, 3.308e-8, 2.976e-8,
             2.67e-8, 2.407e-8, 2.171e-8, 1.965e-8, 1.78e-8, 1.617e-8, 1.47e-8, 1.341e-8, 1.227e-8, 1.125e-8, 1.033e-8, 9.524e-9,
01840
01841
01842
             8.797e-9, 8.162e-9, 7.565e-9, 7.04e-9, 6.56e-9, 6.129e-9,
01843
             5.733e-9, 5.376e-9, 5.043e-9, 4.75e-9, 4.466e-9, 4.211e-9, 3.977e-9, 3.759e-9, 3.558e-9, 3.373e-9, 3.201e-9, 3.043e-9,
01844
             2.895e-9, 2.76e-9, 2.635e-9, 2.518e-9, 2.411e-9, 2.314e-9, 2.23e-9, 2.151e-9, 2.087e-9, 2.035e-9, 1.988e-9, 1.946e-9,
01845
             1.927e-9, 1.916e-9, 1.916e-9, 1.933e-9, 1.966e-9, 2.018e-9,
01847
01848
             2.09e-9, 2.182e-9, 2.299e-9, 2.442e-9, 2.623e-9, 2.832e-9,
             3.079e-9, 3.368e-9, 3.714e-9, 4.104e-9, 4.567e-9, 5.091e-9, 5.701e-9, 6.398e-9, 7.194e-9, 8.127e-9, 9.141e-9, 1.035e-8,
01849
01850
             1.177e-8, 1.338e-8, 1.508e-8, 1.711e-8, 1.955e-8, 2.216e-8,
01851
             2.534e-8, 2.871e-8, 3.291e-8, 3.711e-8, 4.285e-8, 4.868e-8,
01852
01853
             5.509e-8, 6.276e-8, 7.262e-8, 8.252e-8, 9.4e-8, 1.064e-7,
             1.247e-7, 1.411e-7, 1.626e-7, 1.827e-7, 2.044e-7, 2.284e-7, 2.452e-7, 2.854e-7, 3.026e-7, 3.278e-7, 3.474e-7, 3.693e-7,
01854
01855
             3.93e-7, 4.104e-7, 4.22e-7, 4.439e-7, 4.545e-7, 4.778e-7,
01856
             4.812e-7, 5.018e-7, 4.899e-7, 5.075e-7, 5.073e-7, 5.171e-7, 5.131e-7, 5.25e-7, 5.617e-7, 5.846e-7, 6.239e-7, 6.696e-7,
01857
             7.398e-7, 8.073e-7, 9.15e-7, 1.009e-6, 1.116e-6, 1.264e-6,
01859
01860
             1.439e-6, 1.644e-6, 1.856e-6, 2.147e-6, 2.317e-6, 2.713e-6,
             2.882e-6, 2.99e-6, 3.489e-6, 3.581e-6, 4.033e-6, 4.26e-6, 4.543e-6, 4.84e-6, 4.826e-6, 5.013e-6, 5.252e-6, 5.277e-6, 5.306e-6, 5.236e-6, 5.123e-6, 5.171e-6, 4.843e-6, 4.615e-6,
01861
01862
01863
```

```
4.385e-6, 3.97e-6, 3.693e-6, 3.231e-6, 2.915e-6, 2.495e-6,
            2.144e-6, 1.91e-6, 1.639e-6, 1.417e-6, 1.226e-6, 1.065e-6, 9.29e-7, 8.142e-7, 7.161e-7, 6.318e-7, 5.581e-7, 4.943e-7,
01865
01866
            4.376e-7, 3.884e-7, 3.449e-7, 3.06e-7, 2.712e-7, 2.412e-7, 2.139e-7, 1.903e-7, 1.689e-7, 1.499e-7, 1.331e-7, 1.183e-7, 1.05e-7, 9.362e-8, 8.306e-8, 7.403e-8, 6.578e-8, 5.853e-8,
01867
01868
01869
             5.216e-8, 4.632e-8, 4.127e-8, 3.678e-8, 3.279e-8, 2.923e-8,
             2.612e-8, 2.339e-8, 2.094e-8, 1.877e-8, 1.686e-8, 1.516e-8,
01871
01872
             1.366e-8, 1.234e-8, 1.114e-8, 1.012e-8, 9.182e-9, 8.362e-9,
            7.634e-9, 6.981e-9, 6.406e-9, 5.888e-9, 5.428e-9, 5.021e-9, 4.65e-9, 4.326e-9, 4.033e-9, 3.77e-9, 3.536e-9, 3.327e-9,
01873
01874
            3.141e-9, 2.974e-9, 2.825e-9, 2.697e-9, 2.584e-9, 2.488e-9,
01875
             2.406e-9, 2.34e-9, 2.292e-9, 2.259e-9, 2.244e-9, 2.243e-9,
01876
             2.272e-9, 2.31e-9, 2.378e-9, 2.454e-9, 2.618e-9, 2.672e-9,
01877
01878
            2.831e-9, 3.05e-9, 3.225e-9, 3.425e-9, 3.677e-9, 3.968e-9,
            4.221e-9, 4.639e-9, 4.96e-9, 5.359e-9, 5.649e-9, 6.23e-9, 6.716e-9, 7.218e-9, 7.746e-9, 7.988e-9, 8.627e-9, 8.999e-9,
01879
01880
            9.442e-9, 9.82e-9, 1.015e-8, 1.06e-8, 1.079e-8, 1.109e-8, 1.137e-8, 1.186e-8, 1.18e-8, 1.187e-8, 1.194e-8, 1.192e-8,
01881
01883
             1.224e-8, 1.245e-8, 1.246e-8, 1.318e-8, 1.377e-8, 1.471e-8,
             1.582e-8, 1.713e-8, 1.853e-8, 2.063e-8, 2.27e-8, 2.567e-8,
01884
01885
            2.891e-8, 3.264e-8, 3.744e-8, 4.286e-8, 4.915e-8, 5.623e-8,
01886
             6.336e-8, 7.293e-8, 8.309e-8, 9.319e-8, 1.091e-7, 1.243e-7,
            1.348e-7, 1.449e-7, 1.62e-7, 1.846e-7, 1.937e-7, 2.04e-7, 2.179e-7, 2.298e-7, 2.433e-7, 2.439e-7, 2.464e-7, 2.611e-7,
01887
01888
            2.617e-7, 2.582e-7, 2.453e-7, 2.401e-7, 2.349e-7, 2.203e-7, 2.066e-7, 1.939e-7, 1.78e-7, 1.558e-7, 1.391e-7, 1.203e-7,
01890
01891
            1.048e-7, 9.464e-8, 8.306e-8, 7.239e-8, 6.317e-8, 5.52e-8
            4.847e-8, 4.282e-8, 3.796e-8, 3.377e-8, 2.996e-8, 2.678e-8, 2.4e-8, 2.134e-8, 1.904e-8, 1.705e-8, 1.523e-8, 1.35e-8,
01892
01893
             1.204e-8, 1.07e-8, 9.408e-9, 8.476e-9, 7.47e-9, 6.679e-9,
01894
01895
             5.929e-9, 5.267e-9, 4.711e-9, 4.172e-9, 3.761e-9, 3.288e-9,
             2.929e-9, 2.609e-9, 2.315e-9, 2.042e-9, 1.844e-9, 1.64e-9,
01896
01897
             1.47e-9, 1.31e-9, 1.176e-9, 1.049e-9, 9.377e-10, 8.462e-10,
            7.616e-10, 6.854e-10, 6.191e-10, 5.596e-10, 5.078e-10, 4.611e-10, 4.197e-10, 3.83e-10, 3.505e-10, 3.215e-10, 2.956e-10, 2.726e-10,
01898
01899
01900
            2.521e-10, 2.338e-10, 2.173e-10, 2.026e-10, 1.895e-10, 1.777e-10,
             1.672e-10, 1.579e-10, 1.496e-10, 1.423e-10, 1.358e-10, 1.302e-10,
             1.254e-10, 1.216e-10, 1.187e-10, 1.163e-10, 1.147e-10, 1.145e-10,
01902
             1.15e-10, 1.17e-10, 1.192e-10, 1.25e-10, 1.298e-10, 1.345e-10,
01903
01904
            1.405e-10, 1.538e-10, 1.648e-10, 1.721e-10, 1.872e-10, 1.968e-10,
            2.089e-10, 2.172e-10, 2.317e-10, 2.389e-10, 2.503e-10, 2.585e-10,
01905
            2.686e-10, 2.8e-10, 2.895e-10, 3.019e-10, 3.037e-10, 3.076e-10,
01906
             3.146e-10, 3.198e-10, 3.332e-10, 3.397e-10, 3.54e-10, 3.667e-10,
             3.895e-10, 4.071e-10, 4.565e-10, 4.983e-10, 5.439e-10, 5.968e-10,
01908
01909
             6.676e-10, 7.456e-10, 8.405e-10, 9.478e-10, 1.064e-9, 1.218e-9,
01910
            1.386e-9, 1.581e-9, 1.787e-9, 2.032e-9, 2.347e-9, 2.677e-9,
01911
            3.008e-9, 3.544e-9, 4.056e-9, 4.687e-9, 5.331e-9, 6.227e-9,
             6.854e-9, 8.139e-9, 8.945e-9, 9.865e-9, 1.125e-8, 1.178e-8,
01912
            1.364e-8, 1.436e-8, 1.54e-8, 1.672e-8, 1.793e-8, 1.906e-8,
01913
             2.036e-8, 2.144e-8, 2.292e-8, 2.371e-8, 2.493e-8, 2.606e-8,
             2.706e-8, 2.866e-8, 3.036e-8, 3.136e-8, 3.405e-8, 3.665e-8,
01915
01916
            3.837e-8, 4.229e-8, 4.748e-8, 5.32e-8, 5.763e-8, 6.677e-8,
01917
            7.216e-8, 7.716e-8, 8.958e-8, 9.419e-8, 1.036e-7, 1.108e-7,
            1.189e-7, 1.246e-7, 1.348e-7, 1.31e-7, 1.361e-7, 1.364e-7, 1.363e-7, 1.343e-7, 1.293e-7, 1.254e-7, 1.235e-7, 1.158e-7,
01918
01919
             1.107e-7, 9.961e-8, 9.011e-8, 7.91e-8, 6.916e-8, 6.338e-8,
             5.564e-8, 4.827e-8, 4.198e-8, 3.695e-8, 3.276e-8, 2.929e-8,
01921
            2.633e-8, 2.391e-8, 2.192e-8, 2.021e-8, 1.89e-8, 1.772e-8,
01922
01923
            1.667e-8, 1.603e-8, 1.547e-8, 1.537e-8, 1.492e-8, 1.515e-8,
            1.479e-8, 1.45e-8, 1.513e-8, 1.495e-8, 1.529e-8, 1.565e-8, 1.564e-8, 1.553e-8, 1.569e-8, 1.584e-8, 1.57e-8, 1.538e-8, 1.513e-8, 1.472e-8, 1.425e-8, 1.349e-8, 1.328e-8, 1.249e-8,
01924
01925
             1.17e-8, 1.077e-8, 9.514e-9, 8.614e-9, 7.46e-9, 6.621e-9,
01927
01928
             5.775e-9, 5.006e-9, 4.308e-9, 3.747e-9, 3.24e-9, 2.84e-9
01929
            2.481e-9, 2.184e-9, 1.923e-9, 1.71e-9, 1.504e-9, 1.334e-9,
01930
            1.187e-9, 1.053e-9, 9.367e-10, 8.306e-10, 7.419e-10, 6.63e-10,
            3.036e-10, 5.277e-10, 4.717e-10, 4.222e-10, 3.783e-10, 3.39e-10, 3.036e-10, 2.729e-10, 2.455e-10, 2.211e-10, 1.995e-10, 1.804e-10,
01931
01932
             1.635e-10, 1.485e-10, 1.355e-10, 1.24e-10, 1.139e-10, 1.051e-10,
            9.757e-11, 9.114e-11, 8.577e-11, 8.139e-11, 7.792e-11, 7.52e-11, 7.39e-11, 7.31le-11, 7.277e-11, 7.482e-11, 7.698e-11, 8.162e-11,
01934
01935
            8.517e-11, 8.968e-11, 9.905e-11, 1.075e-10, 1.187e-10, 1.291e-10, 1.426e-10, 1.573e-10, 1.734e-10, 1.905e-10, 2.097e-10, 2.28e-10, 2.473e-10, 2.718e-10, 2.922e-10, 3.128e-10, 3.361e-10, 3.641e-10,
01936
01937
01938
             3.91e-10, 4.196e-10, 4.501e-10, 4.932e-10, 5.258e-10, 5.755e-10,
01939
01940
             6.253e-10, 6.664e-10, 7.344e-10, 7.985e-10, 8.877e-10, 1.005e-9,
01941
             1.118e-9, 1.251e-9, 1.428e-9, 1.61e-9, 1.888e-9, 2.077e-9,
01942
             2.331e-9, 2.751e-9, 3.061e-9, 3.522e-9, 3.805e-9, 4.181e-9,
            4.575e-9, 5.167e-9, 5.634e-9, 6.007e-9, 6.501e-9, 6.829e-9, 7.211e-9, 7.262e-9, 7.696e-9, 7.832e-9, 7.799e-9, 7.651e-9,
01943
01944
             7.304e-9, 7.15e-9, 6.977e-9, 6.603e-9, 6.209e-9, 5.69e-9,
            5.432e-9, 4.764e-9, 4.189e-9, 3.64e-9, 3.203e-9, 2.848e-9
2.51e-9, 2.194e-9, 1.946e-9, 1.75e-9, 1.567e-9, 1.426e-9,
01946
01947
01948
            1.302e-9, 1.197e-9, 1.109e-9, 1.035e-9, 9.719e-10, 9.207e-10,
            8.957e-10, 8.578e-10, 8.262e-10, 8.117e-10, 7.987e-10, 7.875e-10, 7.741e-10, 7.762e-10, 7.537e-10, 7.424e-10, 7.474e-10, 7.294e-10,
01949
01950
```

```
7.216e-10, 7.233e-10, 7.075e-10, 6.892e-10, 6.618e-10, 6.314e-10,
           6.208e-10, 5.689e-10, 5.55e-10, 4.984e-10, 4.6e-10, 4.078e-10,
01952
01953
           3.879e-10, 3.459e-10, 2.982e-10, 2.626e-10, 2.329e-10, 1.988e-10,
           1.735e-10, 1.487e-10, 1.297e-10, 1.133e-10, 9.943e-11, 8.736e-11,
01954
           7.726e-11, 6.836e-11, 6.053e-11, 5.384e-11, 4.789e-11, 4.267e-11, 3.804e-11, 3.398e-11, 3.034e-11, 2.71e-11, 2.425e-11, 2.173e-11,
01955
01956
           1.95e-11, 1.752e-11, 1.574e-11, 1.418e-11, 1.278e-11, 1.154e-11,
           1.044e-11, 9.463e-12, 8.602e-12, 7.841e-12, 7.171e-12, 6.584e-12,
01958
01959
           6.073e-12, 5.631e-12, 5.254e-12, 4.937e-12, 4.679e-12, 4.476e-12,
01960
           4.328e-12, 4.233e-12, 4.194e-12, 4.211e-12, 4.286e-12, 4.424e-12,
           4.628e-12, 4.906e-12, 5.262e-12, 5.708e-12, 6.254e-12, 6.914e-12,
01961
01962
           7.714e-12, 8.677e-12, 9.747e-12, 1.101e-11, 1.256e-11, 1.409e-11,
           1.597e-11, 1.807e-11, 2.034e-11, 2.316e-11, 2.622e-11, 2.962e-11,
01963
           3.369e-11, 3.819e-11, 4.329e-11, 4.932e-11, 5.589e-11, 6.364e-11,
01964
01965
           7.284e-11, 8.236e-11, 9.447e-11, 1.078e-10, 1.229e-10, 1.417e-10,
01966
           1.614e-10, 1.843e-10, 2.107e-10, 2.406e-10, 2.728e-10, 3.195e-10,
01967
           3.595e-10, 4.153e-10, 4.736e-10, 5.41e-10, 6.088e-10, 6.769e-10,
           7.691e-10, 8.545e-10, 9.621e-10, 1.047e-9, 1.161e-9, 1.296e-9,
01968
           1.424e-9, 1.576e-9, 1.739e-9, 1.893e-9, 2.08e-9, 2.336e-9,
           2.604e-9, 2.76e-9, 3.001e-9, 3.365e-9, 3.55e-9, 3.895e-9,
01970
01971
            4.183e-9, 4.614e-9, 4.846e-9, 5.068e-9, 5.427e-9, 5.541e-9,
01972
           5.864e-9, 5.997e-9, 5.997e-9, 6.061e-9, 5.944e-9, 5.855e-9,
           5.661e-9, 5.523e-9, 5.374e-9, 4.94e-9, 4.688e-9, 4.17e-9,
01973
           3.913e-9, 3.423e-9, 2.997e-9, 2.598e-9, 2.253e-9, 1.946e-9, 1.71e-9, 1.507e-9, 1.336e-9, 1.19e-9, 1.068e-9, 9.623e-10,
01974
01975
            8.772e-10, 8.007e-10, 7.42e-10, 6.884e-10, 6.483e-10, 6.162e-10,
01976
01977
           5.922e-10, 5.688e-10, 5.654e-10, 5.637e-10, 5.701e-10, 5.781e-10,
01978
           5.874e-10, 6.268e-10, 6.357e-10, 6.525e-10, 7.137e-10, 7.441e-10,
01979
           8.024e-10, 8.485e-10, 9.143e-10, 9.536e-10, 9.717e-10,
                                                                            1.018e-9.
           1.042e-9, 1.054e-9, 1.092e-9, 1.079e-9, 1.064e-9, 1.043e-9, 1.02e-9, 9.687e-10, 9.273e-10, 9.208e-10, 9.068e-10, 7.687e-10,
01980
01981
            7.385e-10, 6.595e-10, 5.87e-10, 5.144e-10, 4.417e-10, 3.804e-10,
           3.301e-10, 2.866e-10, 2.509e-10, 2.202e-10, 1.947e-10, 1.719e-10,
01983
01984
           1.525e-10, 1.361e-10, 1.21e-10, 1.084e-10, 9.8e-11, 8.801e-11,
           7.954e-11, 7.124e-11, 6.335e-11, 5.76e-11, 5.132e-11, 4.601e-11, 4.096e-11, 3.657e-11, 3.25e-11, 2.909e-11, 2.587e-11, 2.297e-11,
01985
01986
           2.05e-11, 1.828e-11, 1.632e-11, 1.462e-11, 1.314e-11, 1.185e-11, 1.073e-11, 9.76e-12, 8.922e-12, 8.206e-12, 7.602e-12, 7.1e-12,
01987
01989
            6.694e-12, 6.378e-12, 6.149e-12, 6.004e-12, 5.941e-12, 5.962e-12,
           6.069e-12, 6.265e-12, 6.551e-12, 6.935e-12, 7.457e-12, 8.074e-12,
01990
01991
           8.811e-12, 9.852e-12, 1.086e-11, 1.207e-11, 1.361e-11, 1.553e-11,
           1.737e-11,\ 1.93e-11,\ 2.175e-11,\ 2.41e-11,\ 2.706e-11,\ 3.023e-11,
01992
           3.313e-11, 3.657e-11, 4.118e-11, 4.569e-11, 5.025e-11, 5.66e-11, 6.231e-11, 6.881e-11, 7.996e-11, 8.526e-11, 9.694e-11, 1.106e-10,
01993
           1.222e-10, 1.355e-10, 1.525e-10, 1.775e-10, 1.924e-10, 2.181e-10,
01995
01996
           2.379e-10, 2.662e-10, 2.907e-10, 3.154e-10, 3.366e-10, 3.579e-10,
01997
           3.858e-10, 4.046e-10, 4.196e-10, 4.166e-10, 4.457e-10, 4.466e-10,
           4.404e-10, 4.337e-10, 4.15e-10, 4.083e-10, 3.91e-10, 3.723e-10, 3.514e-10, 3.303e-10, 2.847e-10, 2.546e-10, 2.23e-10, 1.994e-10,
01998
01999
02000
           1.733e-10, 1.488e-10, 1.297e-10, 1.144e-10, 1.004e-10, 8.741e-11,
            7.928e-11, 7.034e-11, 6.323e-11, 5.754e-11, 5.25e-11, 4.85e-11,
           4.502e-11, 4.286e-11, 4.028e-11, 3.899e-11, 3.824e-11, 3.761e-11,
02002
02003
           3.804e-11, 3.839e-11, 3.845e-11, 4.244e-11, 4.382e-11, 4.582e-11,
02004
           4.847e-11, 5.209e-11, 5.384e-11, 5.887e-11, 6.371e-11, 6.737e-11,
02005
           7.168e-11, 7.415e-11, 7.827e-11, 8.037e-11, 8.12e-11, 8.071e-11,
02006
           8.008e-11, 7.851e-11, 7.544e-11, 7.377e-11, 7.173e-11, 6.801e-11,
           6.267e-11, 5.727e-11, 5.288e-11, 4.853e-11, 4.082e-11, 3.645e-11,
           3.136e-11, 2.672e-11, 2.304e-11, 1.986e-11, 1.725e-11, 1.503e-11,
02008
02009
           1.315e-11, 1.153e-11, 1.014e-11, 8.942e-12, 7.901e-12, 6.993e-12,
02010
           6.199e-12, 5.502e-12, 4.89e-12, 4.351e-12, 3.878e-12, 3.461e-12,
02011
           3.094e-12, 2.771e-12, 2.488e-12, 2.241e-12, 2.025e-12, 1.838e-12,
           1.677e-12, 1.541e-12, 1.427e-12, 1.335e-12, 1.262e-12, 1.209e-12,
02012
           1.176e-12, 1.161e-12, 1.165e-12, 1.189e-12, 1.234e-12, 1.3e-12,
            1.389e-12, 1.503e-12, 1.644e-12, 1.814e-12, 2.017e-12, 2.255e-12,
02014
02015
           2.534e-12, 2.858e-12, 3.231e-12, 3.661e-12, 4.153e-12, 4.717e-12,
02016
           5.36e-12, 6.094e-12, 6.93e-12, 7.882e-12, 8.966e-12, 1.02e-11,
           1.162e-11, 1.324e-11, 1.51e-11, 1.72e-11, 1.965e-11, 2.237e-11, 2.56e-11, 2.927e-11, 3.371e-11, 3.842e-11, 4.429e-11, 5.139e-11, 5.798e-11, 6.697e-11, 7.626e-11, 8.647e-11, 1.022e-10, 1.136e-10,
02017
02018
02019
           1.3e-10, 1.481e-10, 1.672e-10, 1.871e-10, 2.126e-10, 2.357e-10,
02021
           2.583e-10, 2.997e-10, 3.289e-10, 3.702e-10, 4.012e-10, 4.319e-10,
02022
           4.527e-10, 5.001e-10, 5.448e-10, 5.611e-10, 5.76e-10, 5.965e-10,
           6.079e-10, 6.207e-10, 6.276e-10, 6.222e-10, 6.137e-10, 6e-10, 5.814e-10, 5.393e-10, 5.35e-10, 4.947e-10, 4.629e-10, 4.117e-10, 3.712e-10, 3.372e-10, 2.923e-10, 2.55e-10, 2.232e-10, 1.929e-10,
02023
02024
02025
            1.679e-10, 1.46e-10, 1.289e-10, 1.13e-10, 9.953e-11, 8.763e-11,
02026
           7.76e-11, 6.9e-11, 6.16e-11, 5.525e-11, 4.958e-11, 4.489e-11,
02027
02028
           4.072e-11, 3.728e-11, 3.438e-11, 3.205e-11, 3.006e-11, 2.848e-11,
           2.766e-11, 2.688e-11, 2.664e-11, 2.67e-11, 2.696e-11, 2.786e-11,
02029
           2.861e-11, 3.009e-11, 3.178e-11, 3.389e-11, 3.587e-11, 3.819e-11,
02030
           4.054e-11, 4.417e-11, 4.703e-11, 5.137e-11, 5.46e-11, 6.055e-11, 6.333e-11, 6.773e-11, 7.219e-11, 7.717e-11, 8.131e-11, 8.491e-11,
02031
           8.574e-11, 9.01e-11, 9.017e-11, 8.999e-11, 8.959e-11, 8.838e-11,
02033
02034
           8.579e-11, 8.162e-11, 8.098e-11, 7.472e-11, 7.108e-11, 6.559e-11,
02035
           5.994 e^{-11}, \ 5.172 e^{-11}, \ 4.424 e^{-11}, \ 3.951 e^{-11}, \ 3.34 e^{-11}, \ 2.902 e^{-11},
           2.541e-11, 2.215e-11, 1.945e-11, 1.716e-11, 1.503e-11, 1.339e-11, 1.185e-11, 1.05e-11, 9.336e-12, 8.307e-12, 7.312e-12, 6.55e-12,
02036
02037
```

```
5.836e-12, 5.178e-12, 4.6e-12, 4.086e-12, 3.639e-12, 3.247e-12,
                 2.904e-12, 2.604e-12, 2.341e-12, 2.112e-12, 1.914e-12, 1.744e-12, 1.598e-12, 1.476e-12, 1.374e-12, 1.293e-12, 1.23e-12, 1.185e-12,
02039
02040
02041
                 1.158e-12, 1.147e-12, 1.154e-12, 1.177e-12, 1.219e-12, 1.28e-12,
                 1.36e-12, 1.463e-12, 1.591e-12, 1.75e-12, 1.94e-12, 2.156e-12, 2.43e-12, 2.748e-12, 3.052e-12, 3.533e-12, 3.967e-12, 4.471e-12,
02042
02043
                 5.041e-12, 5.86e-12, 6.664e-12, 7.522e-12, 8.342e-12, 9.412e-12,
                  1.072e-11, 1.213e-11, 1.343e-11, 1.496e-11, 1.664e-11, 1.822e-11,
02045
02046
                 2.029e-11, 2.233e-11, 2.457e-11, 2.709e-11, 2.928e-11, 3.115e-11,
02047
                 3.356e-11, 3.592e-11, 3.818e-11, 3.936e-11, 4.061e-11, 4.149e-11,
02048
                  4.299e-11, 4.223e-11, 4.251e-11, 4.287e-11, 4.177e-11, 4.094e-11,
                 3.942 e^{-11}, \ 3.772 e^{-11}, \ 3.614 e^{-11}, \ 3.394 e^{-11}, \ 3.222 e^{-11}, \ 2.791 e^{-11}, \\
02049
                 2.665e-11, 2.309e-11, 2.032e-11, 1.74e-11, 1.535e-11, 1.323e-11, 1.151e-11, 9.803e-12, 8.65e-12, 7.54e-12, 6.619e-12, 5.832e-12,
02050
02051
02052
                  5.113e-12, 4.503e-12, 3.975e-12, 3.52e-12, 3.112e-12, 2.797e-12,
                 2.5e-12, 2.24e-12, 2.013e-12, 1.819e-12, 1.653e-12, 1.513e-12, 1.395e-12, 1.299e-12, 1.225e-12, 1.168e-12, 1.124e-12, 1.148e-12,
02053
02054
                 1.107e-12, 1.128e-12, 1.169e-12, 1.233e-12, 1.307e-12, 1.359e-12, 1.543e-12, 1.686e-12, 1.794e-12, 2.028e-12, 2.21e-12, 2.441e-12,
02055
                 2.653e-12, 2.828e-12, 3.093e-12, 3.28e-12, 3.551e-12, 3.677e-12,
02057
                  3.803e-12, 3.844e-12, 4.068e-12, 4.093e-12, 4.002e-12, 3.904e-12,
02058
02059
                 3.624e-12, 3.633e-12, 3.622e-12, 3.443e-12, 3.184e-12, 2.934e-12,
                 2.476e-12, 2.212e-12, 1.867e-12, 1.594e-12, 1.37e-12, 1.192e-12, 1.045e-12, 9.211e-13, 8.17e-13, 7.29e-13, 6.55e-13, 5.929e-13, 5.415e-13, 4.995e-13, 4.661e-13, 4.406e-13, 4.225e-13, 4.116e-13,
02060
02061
02062
02063
                  4.075e-13, 4.102e-13, 4.198e-13, 4.365e-13, 4.606e-13, 4.925e-13,
                  5.326e-13, 5.818e-13, 6.407e-13, 7.104e-13, 7.92e-13, 8.868e-13,
02064
02065
                 9.964e-13, 1.123e-12, 1.268e-12, 1.434e-12, 1.626e-12, 1.848e-12,
02066
                 2.107e-12, 2.422e-12, 2.772e-12, 3.145e-12, 3.704e-12, 4.27e-12,
                 4.721e-12, 5.361e-12, 6.083e-12, 7.095e-12, 7.968e-12, 9.228e-12,
02067
02068
                 1.048e-11, 1.187e-11, 1.336e-11, 1.577e-11, 1.772e-11, 2.017e-11,
                  2.25e-11, 2.63e-11, 2.911e-11, 3.356e-11, 3.82e-11, 4.173e-11,
02070
                  4.811e-11, 5.254e-11, 5.839e-11, 6.187e-11, 6.805e-11, 7.118e-11,
02071
                  7.369e-11, 7.664e-11, 7.794e-11, 7.947e-11, 8.036e-11, 7.954e-11,
02072
                  7.849e-11, 7.518e-11, 7.462e-11, 6.926e-11, 6.531e-11, 6.197e-11,
                 5.421e-11, 4.777e-11, 4.111e-11, 3.679e-11, 3.166e-11, 2.786e-11,
02073
02074
                 2.436e-11, 2.144e-11, 1.859e-11, 1.628e-11, 1.414e-11, 1.237e-11,
                 1.093e-11, 9.558e-12
02076
02077
02078
              static double h2o260[2001] = \{ .2752, .2732, .2749, .2676, .2667, .2545, .2545, .2749, .2676, .2667, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .2687, .268
                 .2497, .2327, .2218, .2036, .1825, .1694, .1497, .1353, .121, .1014, .09405, .07848, .07195, .06246, .05306, .04853, .04138, .03735, .03171, .02785, .02431, .02111, .01845, .0164, .01405,
02079
02080
                 .01255, .01098, .009797, .008646, .007779, .006898, .006099,
02082
02083
                  .005453, .004909, .004413, .003959, .003581, .003199, .002871,
                  .002583, .00233, .002086, .001874, .001684, .001512, .001361, .001225, .0011, 9.89e-4, 8.916e-4, 8.039e-4, 7.256e-4, 6.545e-4,
02084
02085
                 5.918e-4, 5.359e-4, 4.867e-4, 4.426e-4, 4.033e-4, 3.682e-4, 3.366e-4, 3.085e-4, 2.833e-4, 2.605e-4, 2.403e-4, 2.221e-4,
02086
02087
                  2.055e-4, 1.908e-4, 1.774e-4, 1.653e-4, 1.544e-4, 1.443e-4,
                  1.351e-4, 1.267e-4, 1.19e-4, 1.119e-4, 1.053e-4, 9.922e-5
02089
02090
                  9.355e-5, 8.831e-5, 8.339e-5, 7.878e-5, 7.449e-5, 7.043e-5,
                 6.664e-5, 6.307e-5, 5.969e-5, 5.654e-5, 5.357e-5, 5.075e-5, 4.81e-5, 4.56e-5, 4.322e-5, 4.102e-5, 3.892e-5, 3.696e-5, 3.511e-5, 3.339e-5, 3.177e-5, 3.026e-5, 2.886e-5, 2.756e-5, 2.636e-5, 2.527e-5, 2.427e-5, 2.337e-5, 2.257e-5, 2.185e-5,
02091
02092
02093
                  2.127e-5, 2.08e-5, 2.041e-5, 2.013e-5, 2e-5, 1.997e-5, 2.009e-5,
02095
02096
                 2.031e-5, 2.068e-5, 2.124e-5, 2.189e-5, 2.267e-5, 2.364e-5,
                 2.463e-5, 2.618e-5, 2.774e-5, 2.937e-5, 3.144e-5, 3.359e-5,
02097
                  3.695e-5, 4.002e-5, 4.374e-5, 4.947e-5, 5.431e-5, 6.281e-5,
02098
                 7.169e-5, 8.157e-5, 9.728e-5, 1.079e-4, 1.337e-4, 1.442e-4, 1.683e-4, 1.879e-4, 2.223e-4, 2.425e-4, 2.838e-4, 3.143e-4,
02099
                  3.527e-4, 4.012e-4, 4.237e-4, 4.747e-4, 5.057e-4, 5.409e-4,
02101
02102
                  5.734e-4, 5.944e-4, 6.077e-4, 6.175e-4, 6.238e-4, 6.226e-4,
02103
                  6.248e-4, 6.192e-4, 6.098e-4, 5.818e-4, 5.709e-4, 5.465e-4,
02104
                 5.043e-4, 4.699e-4, 4.294e-4, 3.984e-4, 3.672e-4, 3.152e-4, 2.883e-4, 2.503e-4, 2.211e-4, 1.92e-4, 1.714e-4, 1.485e-4,
02105
                 1.358e-4, 1.156e-4, 1.021e-4, 8.887e-5, 7.842e-5, 7.12e-5,
02106
                  6.186e-5, 5.73e-5, 4.792e-5, 4.364e-5, 3.72e-5, 3.28e-5,
                  2.946e-5, 2.591e-5, 2.261e-5, 2.048e-5, 1.813e-5, 1.63e-5,
02108
02109
                 1.447e-5, 1.282e-5, 1.167e-5, 1.041e-5, 9.449e-6, 8.51e-6,
02110
                  7.596e-6, 6.961e-6, 6.272e-6, 5.728e-6, 5.198e-6, 4.667e-6,
                 4.288e-6, 3.897e-6, 3.551e-6, 3.235e-6, 2.952e-6, 2.688e-6, 2.449e-6, 2.241e-6, 2.05e-6, 1.879e-6, 1.722e-6, 1.582e-6,
02111
02112
                  1.456e-6, 1.339e-6, 1.236e-6, 1.144e-6, 1.06e-6, 9.83e-7,
02113
                  9.149e-7, 8.535e-7, 7.973e-7, 7.466e-7, 6.999e-7, 6.574e-7,
02114
                 6.18e-7, 5.821e-7, 5.487e-7, 5.18e-7, 4.896e-7, 4.631e-7, 4.386e-7, 4.16e-7, 3.945e-7, 3.748e-7, 3.562e-7, 3.385e-7, 3.222e-7, 3.068e-7, 2.922e-7, 2.788e-7, 2.659e-7, 2.539e-7,
02115
02116
02117
                 2.425e-7, 2.318e-7, 2.219e-7, 2.127e-7, 2.039e-7, 1.958e-7, 1.885e-7, 1.818e-7, 1.758e-7, 1.711e-7, 1.662e-7, 1.63e-7,
02118
                 1.605e-7, 1.58e-7, 1.559e-7, 1.545e-7, 1.532e-7, 1.522e-7, 1.51e-7, 1.495e-7, 1.465e-7, 1.483e-7, 1.469e-7, 1.448e-7,
02120
02121
02122
                 1.444e-7, 1.436e-7, 1.426e-7, 1.431e-7, 1.425e-7, 1.445e-7,
                 1.477e-7, 1.515e-7, 1.567e-7, 1.634e-7, 1.712e-7, 1.802e-7, 1.914e-7, 2.024e-7, 2.159e-7, 2.295e-7, 2.461e-7, 2.621e-7,
02123
```

```
2.868e-7, 3.102e-7, 3.394e-7, 3.784e-7, 4.223e-7, 4.864e-7,
              5.501e-7, 6.039e-7, 7.193e-7, 7.728e-7, 9.514e-7, 1.073e-6, 1.18e-6, 1.333e-6, 1.472e-6, 1.566e-6, 1.677e-6, 1.784e-6,
02126
02127
              1.904e-6, 1.953e-6, 2.02e-6, 2.074e-6, 2.128e-6, 2.162e-6,
02128
              2.219e-6, 2.221e-6, 2.249e-6, 2.239e-6, 2.235e-6, 2.185e-6, 2.141e-6, 2.124e-6, 2.09e-6, 2.068e-6, 2.1e-6, 2.104e-6,
02129
02130
              2.142e-6, 2.181e-6, 2.257e-6, 2.362e-6, 2.5e-6, 2.664e-6,
              2.884e-6, 3.189e-6, 3.48e-6, 3.847e-6, 4.313e-6, 4.79e-6,
02132
02133
              5.25e-6, 5.989e-6, 6.692e-6, 7.668e-6, 8.52e-6, 9.606e-6,
              1.073e-5, 1.225e-5, 1.377e-5, 1.582e-5, 1.761e-5, 2.029e-5, 2.284e-5, 2.602e-5, 2.94e-5, 3.483e-5, 3.928e-5, 4.618e-5, 5.24e-5, 6.132e-5, 7.183e-5, 8.521e-5, 9.111e-5, 1.07e-4,
02134
02135
02136
02137
               1.184e-4, 1.264e-4, 1.475e-4, 1.612e-4, 1.704e-4, 1.818e-4,
              1.924e-4, 1.994e-4, 2.061e-4, 2.18e-4, 2.187e-4, 2.2e-4,
02138
02139
              2.196e-4, 2.131e-4, 2.015e-4, 1.988e-4, 1.847e-4, 1.729e-4,
              1.597e-4, 1.373e-4, 1.262e-4, 1.087e-4, 9.439e-5, 8.061e-5, 7.093e-5, 6.049e-5, 5.12e-5, 4.435e-5, 3.817e-5, 3.34e-5, 2.927e-5, 2.573e-5, 2.291e-5, 2.04e-5, 1.827e-5, 1.636e-5,
02140
02141
02142
              1.463e-5, 1.309e-5, 1.17e-5, 1.047e-5, 9.315e-6, 8.328e-6, 7.458e-6, 6.665e-6, 5.94e-6, 5.316e-6, 4.752e-6, 4.252e-6,
02144
              3.825e-6, 3.421e-6, 3.064e-6, 2.746e-6, 2.465e-6, 2.216e-6,
02145
              1.99e-6, 1.79e-6, 1.609e-6, 1.449e-6, 1.306e-6, 1.177e-6, 1.063e-6, 9.607e-7, 8.672e-7, 7.855e-7, 7.118e-7, 6.46e-7, 5.871e-7, 5.34e-7, 4.868e-7, 4.447e-7, 4.068e-7, 3.729e-7, 3.423e-7, 3.151e-7, 2.905e-7, 2.686e-7, 2.484e-7, 2.306e-7,
02146
02147
02148
02149
              2.142e-7, 1.995e-7, 1.86e-7, 1.738e-7, 1.626e-7, 1.522e-7, 1.427e-7, 1.338e-7, 1.258e-7, 1.183e-7, 1.116e-7, 1.056e-7,
02150
02151
02152
              9.972e-8, 9.46e-8, 9.007e-8, 8.592e-8, 8.195e-8, 7.816e-8,
              7.483e-8, 7.193e-8, 6.892e-8, 6.642e-8, 6.386e-8, 6.154e-8, 5.949e-8, 5.764e-8, 5.622e-8, 5.479e-8, 5.364e-8, 5.301e-8,
02153
02154
02155
              5.267e-8, 5.263e-8, 5.313e-8, 5.41e-8, 5.55e-8, 5.745e-8,
02156
               6.003e-8, 6.311e-8, 6.713e-8, 7.173e-8, 7.724e-8, 8.368e-8,
              9.121e-8, 9.986e-8, 1.097e-7, 1.209e-7, 1.338e-7, 1.486e-7,
02157
02158
              1.651e-7, 1.837e-7, 2.048e-7, 2.289e-7, 2.557e-7, 2.857e-7,
              3.195e-7, 3.587e-7, 4.015e-7, 4.497e-7, 5.049e-7, 5.665e-7, 6.366e-7, 7.121e-7, 7.996e-7, 8.946e-7, 1.002e-6, 1.117e-6,
02159
02160
              1.262e-6, 1.416e-6, 1.611e-6, 1.807e-6, 2.056e-6, 2.351e-6, 2.769e-6, 3.138e-6, 3.699e-6, 4.386e-6, 5.041e-6, 6.074e-6,
02161
02162
02163
              6.812e-6, 7.79e-6, 8.855e-6, 1.014e-5, 1.095e-5, 1.245e-5,
              1.316e-5, 1.39e-5, 1.504e-5, 1.583e-5, 1.617e-5, 1.652e-5,
02164
              1.713e-5, 1.724e-5, 1.715e-5, 1.668e-5, 1.629e-5, 1.552e-5, 1.478e-5, 1.34e-5, 1.245e-5, 1.121e-5, 9.575e-6, 8.956e-6, 7.345e-6, 6.597e-6, 5.612e-6, 4.818e-6, 4.165e-6, 3.579e-6, 3.041e-6, 2.623e-6, 2.29e-6, 1.984e-6, 1.748e-6, 1.534e-6,
02165
02166
02167
02168
              1.369e-6, 1.219e-6, 1.092e-6, 9.8e-7, 8.762e-7, 7.896e-7, 7.104e-7, 6.364e-7, 5.691e-7, 5.107e-7, 4.575e-7, 4.09e-7
02169
02170
02171
              3.667e-7, 3.287e-7, 2.931e-7, 2.633e-7, 2.356e-7, 2.111e-7,
              1.895e-7, 1.697e-7, 1.525e-7, 1.369e-7, 1.233e-7, 1.114e-7, 9.988e-8, 9.004e-8, 8.149e-8, 7.352e-8, 6.662e-8, 6.03e-8,
02172
02173
02174
              5.479e-8, 4.974e-8, 4.532e-8, 4.129e-8, 3.781e-8, 3.462e-8,
              3.176e-8, 2.919e-8, 2.687e-8, 2.481e-8, 2.292e-8, 2.119e-8,
              1.967e-8, 1.828e-8, 1.706e-8, 1.589e-8, 1.487e-8, 1.393e-8,
02176
02177
              1.307e-8, 1.228e-8, 1.156e-8, 1.089e-8, 1.028e-8, 9.696e-9,
              9.159e-9, 8.658e-9, 8.187e-9, 7.746e-9, 7.34e-9, 6.953e-9, 6.594e-9, 6.259e-9, 5.948e-9, 5.66e-9, 5.386e-9, 5.135e-9,
02178
02179
              4.903e-9, 4.703e-9, 4.515e-9, 4.362e-9, 4.233e-9, 4.117e-9,
02180
              4.017e-9, 3.962e-9, 3.924e-9, 3.905e-9, 3.922e-9, 3.967e-9,
              4.046e-9, 4.165e-9, 4.32e-9, 4.522e-9, 4.769e-9, 5.083e-9,
02182
              5.443e-9, 5.872e-9, 6.366e-9, 6.949e-9, 7.601e-9, 8.371e-9,
02183
02184
              9.22e-9, 1.02e-8, 1.129e-8, 1.251e-8, 1.393e-8, 1.542e-8,
              1.72e-8, 1.926e-8, 2.152e-8, 2.392e-8, 2.678e-8, 3.028e-8, 3.39e-8, 3.836e-8, 4.309e-8, 4.9e-8, 5.481e-8, 6.252e-8,
02185
02186
02187
               7.039e-8, 7.883e-8, 8.849e-8, 1.012e-7, 1.142e-7, 1.3e-7,
               1.475e-7, 1.732e-7, 1.978e-7, 2.304e-7, 2.631e-7, 2.988e-7,
02188
              3.392e-7, 3.69e-7, 4.355e-7, 4.672e-7, 5.11e-7, 5.461e-7, 5.828e-7, 6.233e-7, 6.509e-7, 6.672e-7, 6.969e-7, 7.104e-7,
02189
02190
              7.439e-7, 7.463e-7, 7.708e-7, 7.466e-7, 7.668e-7, 7.549e-7, 7.586e-7, 7.384e-7, 7.439e-7, 7.785e-7, 7.915e-7, 8.31e-7,
02191
02192
02193
              8.745e-7, 9.558e-7, 1.038e-6, 1.173e-6, 1.304e-6, 1.452e-6,
02194
              1.671e-6, 1.931e-6, 2.239e-6, 2.578e-6, 3.032e-6, 3.334e-6,
02195
              3.98e-6, 4.3e-6, 4.518e-6, 5.321e-6, 5.508e-6, 6.211e-6, 6.59e-6,
02196
              7.046e-6, 7.555e-6, 7.558e-6, 7.875e-6, 8.319e-6, 8.433e-6,
              8.59e-6, 8.503e-6, 8.304e-6, 8.336e-6, 7.739e-6, 7.301e-6, 6.827e-6, 6.078e-6, 5.551e-6, 4.762e-6, 4.224e-6, 3.538e-6, 2.984e-6, 2.619e-6, 2.227e-6, 1.923e-6, 1.669e-6, 1.462e-6,
02197
02198
02199
              1.294e-6, 1.155e-6, 1.033e-6, 9.231e-7, 8.238e-7, 7.36e-7,
02200
02201
              6.564e-7, 5.869e-7, 5.236e-7, 4.673e-7, 4.174e-7, 3.736e-7,
              3.33e-7, 2.976e-7, 2.657e-7, 2.367e-7, 2.106e-7, 1.877e-7, 1.671e-7, 1.494e-7, 1.332e-7, 1.192e-7, 1.065e-7, 9.558e-8, 8.586e-8, 7.717e-8, 6.958e-8, 6.278e-8, 5.666e-8, 5.121e-8,
02202
02203
02204
              4.647e-8, 4.213e-8, 3.815e-8, 3.459e-8, 3.146e-8, 2.862e-8, 2.604e-8, 2.375e-8, 2.162e-8, 1.981e-8, 1.817e-8, 1.67e-8,
02205
               1.537e-8, 1.417e-8, 1.31e-8, 1.215e-8, 1.128e-8, 1.05e-8,
02207
              9.793e-9, 9.158e-9, 8.586e-9, 8.068e-9, 7.595e-9, 7.166e-9,
02208
02209
              6.778e-9, 6.427e-9, 6.108e-9, 5.826e-9, 5.571e-9, 5.347e-9,
              5.144e-9, 4.968e-9, 4.822e-9, 4.692e-9, 4.589e-9, 4.506e-9, 4.467e-9, 4.44e-9, 4.466e-9, 4.515e-9, 4.718e-9, 4.729e-9,
02210
02211
```

```
4.937e-9, 5.249e-9, 5.466e-9, 5.713e-9, 6.03e-9, 6.436e-9,
            6.741e-9, 7.33e-9, 7.787e-9, 8.414e-9, 8.908e-9, 9.868e-9, 1.069e-8, 1.158e-8, 1.253e-8, 1.3e-8, 1.409e-8, 1.47e-8,
02213
02214
            1.548e-8, 1.612e-8, 1.666e-8, 1.736e-8, 1.763e-8, 1.812e-8,
02215
            1.852e-8, 1.923e-8, 1.897e-8, 1.893e-8, 1.888e-8, 1.868e-8, 1.895e-8, 1.899e-8, 1.876e-8, 1.96e-8, 2.02e-8, 2.121e-8,
02216
02217
            2.239e-8, 2.379e-8, 2.526e-8, 2.766e-8, 2.994e-8, 3.332e-8
            3.703e-8, 4.158e-8, 4.774e-8, 5.499e-8, 6.355e-8, 7.349e-8,
02219
02220
            8.414e-8, 9.846e-8, 1.143e-7, 1.307e-7, 1.562e-7, 1.817e-7,
            2.011e-7, 2.192e-7, 2.485e-7, 2.867e-7, 3.035e-7, 3.223e-7
02221
            3.443e-7, 3.617e-7, 3.793e-7, 3.793e-7, 3.839e-7, 4.081e-7, 4.117e-7, 4.085e-7, 3.92e-7, 3.851e-7, 3.754e-7, 3.49e-7, 3.229e-7, 2.978e-7, 2.691e-7, 2.312e-7, 2.029e-7, 1.721e-7,
02222
02223
02224
            1.472e-7, 1.308e-7, 1.132e-7, 9.736e-8, 8.458e-8, 7.402e-8,
02225
02226
            6.534e-8, 5.811e-8, 5.235e-8, 4.762e-8, 4.293e-8, 3.896e-8,
            3.526e-8, 3.165e-8, 2.833e-8, 2.551e-8, 2.288e-8, 2.036e-8, 1.82e-8, 1.626e-8, 1.438e-8, 1.299e-8, 1.149e-8, 1.03e-8,
02227
02228
            9.148e-9, 8.122e-9, 7.264e-9, 6.425e-9, 5.777e-9, 5.06e-9,
02229
            4.502e-9, 4.013e-9, 3.567e-9, 3.145e-9, 2.864e-9, 2.553e-9,
02231
            2.311e-9, 2.087e-9, 1.886e-9, 1.716e-9, 1.556e-9, 1.432e-9,
            1.311e-9, 1.202e-9, 1.104e-9, 1.013e-9, 9.293e-10, 8.493e-10,
02232
02233
            7.79e-10, 7.185e-10, 6.642e-10, 6.141e-10, 5.684e-10, 5.346e-10,
            5.032e-10, 4.725e-10, 4.439e-10, 4.176e-10, 3.93e-10, 3.714e-10, 3.515e-10, 3.332e-10, 3.167e-10, 3.02e-10, 2.887e-10, 2.769e-10, 2.665e-10, 2.578e-10, 2.503e-10, 2.436e-10, 2.377e-10, 2.342e-10,
02234
02235
02236
            2.305e-10, 2.296e-10, 2.278e-10, 2.321e-10, 2.355e-10, 2.402e-10,
            2.478e-10, 2.67e-10, 2.848e-10, 2.982e-10, 3.263e-10, 3.438e-10,
02238
02239
            3.649e-10, 3.829e-10, 4.115e-10, 4.264e-10, 4.473e-10, 4.63e-10,
02240
            4.808e-10, 4.995e-10, 5.142e-10, 5.313e-10, 5.318e-10, 5.358e-10,
02241
            5.452e-10, 5.507e-10, 5.698e-10, 5.782e-10, 5.983e-10, 6.164e-10,
02242
            6.532e-10, 6.811e-10, 7.624e-10, 8.302e-10, 9.067e-10, 9.937e-10,
02243
            1.104e-9, 1.221e-9, 1.361e-9, 1.516e-9, 1.675e-9, 1.883e-9,
            2.101e-9, 2.349e-9, 2.614e-9, 2.92e-9, 3.305e-9, 3.724e-9,
02244
02245
            4.142e-9, 4.887e-9, 5.614e-9, 6.506e-9, 7.463e-9, 8.817e-9,
            9.849e-9, 1.187e-8, 1.321e-8, 1.474e-8, 1.698e-8, 1.794e-8, 2.09e-8, 2.211e-8, 2.362e-8, 2.556e-8, 2.729e-8, 2.88e-8,
02246
02247
            3.046e-8, 3.167e-8, 3.367e-8, 3.457e-8, 3.59e-8, 3.711e-8,
02248
            3.826e-8, 4.001e-8, 4.211e-8, 4.315e-8, 4.661e-8, 5.01e-8, 5.249e-8, 5.84e-8, 6.628e-8, 7.512e-8, 8.253e-8, 9.722e-8,
02250
            1.067e-7, 1.153e-7, 1.347e-7, 1.428e-7, 1.577e-7, 1.694e-7, 1.833e-7, 1.938e-7, 2.108e-7, 2.059e-7, 2.157e-7, 2.185e-7,
02251
02252
            2.208e-7, 2.182e-7, 2.093e-7, 2.014e-7, 1.962e-7, 1.819e-7,
02253
            1.713e-7, 1.51e-7, 1.34e-7, 1.154e-7, 9.89e-8, 8.88e-8, 7.673e-8, 6.599e-8, 5.73e-8, 5.081e-8, 4.567e-8, 4.147e-8, 3.773e-8,
02254
            3.46e-8, 3.194e-8, 2.953e-8, 2.759e-8, 2.594e-8, 2.442e-8,
02256
02257
            2.355e-8, 2.283e-8, 2.279e-8, 2.231e-8, 2.279e-8, 2.239e-8
            2.21e-8, 2.309e-8, 2.293e-8, 2.352e-8, 2.415e-8, 2.43e-8, 2.426e-8, 2.465e-8, 2.5e-8, 2.496e-8, 2.465e-8, 2.445e-8, 2.383e-8, 2.299e-8, 2.165e-8, 2.113e-8, 1.968e-8, 1.819e-8,
02258
02259
02260
            1.644e-8, 1.427e-8, 1.27e-8, 1.082e-8, 9.428e-9, 8.091e-9,
02261
            6.958e-9, 5.988e-9, 5.246e-9, 4.601e-9, 4.098e-9, 3.664e-9,
            3.287e-9, 2.942e-9, 2.656e-9, 2.364e-9, 2.118e-9, 1.903e-9,
02263
02264
            1.703e-9, 1.525e-9, 1.365e-9, 1.229e-9, 1.107e-9, 9.96e-10,
            8.945e-10, 8.08e-10, 7.308e-10, 6.616e-10, 5.994e-10, 5.422e-10, 4.929e-10, 4.478e-10, 4.07e-10, 3.707e-10, 3.379e-10, 3.087e-10,
02265
02266
            2.823e-10, 2.592e-10, 2.385e-10, 2.201e-10, 2.038e-10, 1.897e-10,
02267
            1.774e-10, 1.667e-10, 1.577e-10, 1.502e-10, 1.437e-10, 1.394e-10,
            1.358e-10, 1.324e-10, 1.329e-10, 1.324e-10, 1.36e-10, 1.39e-10,
02269
            1.424e-10, 1.544e-10, 1.651e-10, 1.817e-10, 1.984e-10, 2.195e-10,
02270
02271
            2.438e-10, 2.7e-10, 2.991e-10, 3.322e-10, 3.632e-10, 3.957e-10,
            4.36e-10, 4.701e-10, 5.03e-10, 5.381e-10, 5.793e-10, 6.19e-10,
02272
            6.596e-10, 7.004e-10, 7.561e-10, 7.934e-10, 8.552e-10, 9.142e-10,
02273
            9.57e-10, 1.027e-9, 1.097e-9, 1.193e-9, 1.334e-9, 1.47e-9,
            1.636e-9, 1.871e-9, 2.122e-9, 2.519e-9, 2.806e-9, 3.203e-9
02275
02276
            3.846e-9, 4.362e-9, 5.114e-9, 5.643e-9, 6.305e-9, 6.981e-9,
02277
            7.983e-9, 8.783e-9, 9.419e-9, 1.017e-8, 1.063e-8, 1.121e-8,
02278
            1.13e-8, 1.201e-8, 1.225e-8, 1.232e-8, 1.223e-8, 1.177e-8, 1.151e-8, 1.116e-8, 1.047e-8, 9.698e-9, 8.734e-9, 8.202e-9,
02279
02280
            7.041e-9, 6.074e-9, 5.172e-9, 4.468e-9, 3.913e-9, 3.414e-9,
            2.975e-9, 2.65e-9, 2.406e-9, 2.173e-9, 2.009e-9, 1.861e-9,
            1.727e-9, 1.612e-9, 1.514e-9, 1.43e-9, 1.362e-9, 1.333e-9,
02282
02283
            1.288e-9, 1.249e-9, 1.238e-9, 1.228e-9, 1.217e-9, 1.202e-9,
            1.209e-9, 1.177e-9, 1.157e-9, 1.165e-9, 1.142e-9, 1.131e-9, 1.138e-9, 1.117e-9, 1.1e-9, 1.069e-9, 1.023e-9, 1.005e-9,
02284
02285
            9.159e-10, 8.863e-10, 7.865e-10, 7.153e-10, 6.247e-10, 5.846e-10, 5.133e-10, 4.36e-10, 3.789e-10, 3.335e-10, 2.833e-10, 2.483e-10,
02286
02287
            2.155e-10, 1.918e-10, 1.709e-10, 1.529e-10, 1.374e-10, 1.235e-10,
02288
02289
            1.108e-10, 9.933e-11, 8.932e-11, 8.022e-11, 7.224e-11, 6.52e-11,
02290
            5.896e-11, 5.328e-11, 4.813e-11, 4.365e-11, 3.961e-11, 3.594e-11,
02291
            3.266e-11, 2.967e-11, 2.701e-11, 2.464e-11, 2.248e-11, 2.054e-11,
            1.878e-11, 1.721e-11, 1.579e-11, 1.453e-11, 1.341e-11, 1.241e-11,
02292
            1.154e-11, 1.078e-11, 1.014e-11, 9.601e-12, 9.167e-12, 8.838e-12,
            8.614e-12, 8.493e-12, 8.481e-12, 8.581e-12, 8.795e-12, 9.131e-12,
02294
02295
            9.601e-12, 1.021e-11, 1.097e-11, 1.191e-11, 1.303e-11, 1.439e-11,
02296
            1.601e-11, 1.778e-11, 1.984e-11, 2.234e-11, 2.474e-11, 2.766e-11,
            3.085e-11, 3.415e-11, 3.821e-11, 4.261e-11, 4.748e-11, 5.323e-11, 5.935e-11, 6.619e-11, 7.418e-11, 8.294e-11, 9.26e-11, 1.039e-10,
02297
02298
```

```
1.156e-10, 1.297e-10, 1.46e-10, 1.641e-10, 1.858e-10, 2.1e-10,
              2.383e-10, 2.724e-10, 3.116e-10, 3.538e-10, 4.173e-10, 4.727e-10, 5.503e-10, 6.337e-10, 7.32e-10, 8.298e-10, 9.328e-10, 1.059e-9,
02300
02301
              1.176e-9, 1.328e-9, 1.445e-9, 1.593e-9, 1.77e-9, 1.954e-9,
02302
              2.175e-9, 2.405e-9, 2.622e-9, 2.906e-9, 3.294e-9, 3.713e-9, 3.98e-9, 4.384e-9, 4.987e-9, 5.311e-9, 5.874e-9, 6.337e-9,
02303
02304
              7.027e-9, 7.39e-9, 7.769e-9, 8.374e-9, 8.605e-9, 9.165e-9,
              9.415e-9, 9.511e-9, 9.704e-9, 9.588e-9, 9.45e-9, 9.086e-9,
02306
02307
              8.798e-9, 8.469e-9, 7.697e-9, 7.168e-9, 6.255e-9, 5.772e-9,
              4.97e-9, 4.271e-9, 3.653e-9, 3.154e-9, 2.742e-9, 2.435e-9, 2.166e-9, 1.936e-9, 1.731e-9, 1.556e-9, 1.399e-9, 1.272e-9,
02308
02309
              1.157e-9, 1.066e-9, 9.844e-10, 9.258e-10, 8.787e-10, 8.421e-10,
02310
02311
              8.083e-10, 8.046e-10, 8.067e-10, 8.181e-10, 8.325e-10, 8.517e-10,
              9.151e-10, 9.351e-10, 9.677e-10, 1.071e-9, 1.126e-9, 1.219e-9,
02312
02313
              1.297e-9, 1.408e-9, 1.476e-9, 1.517e-9, 1.6e-9, 1.649e-9,
              1.678e-9, 1.746e-9, 1.742e-9, 1.728e-9, 1.699e-9, 1.655e-9, 1.561e-9, 1.48e-9, 1.451e-9, 1.411e-9, 1.171e-9, 1.106e-9,
02314
02315
              9.714e-10, 8.523e-10, 7.346e-10, 6.241e-10, 5.371e-10, 4.704e-10,
02316
              4.144e-10, 3.683e-10, 3.292e-10, 2.942e-10, 2.62e-10, 2.341e-10,
              2.104e-10, 1.884e-10, 1.7e-10, 1.546e-10, 1.394e-10, 1.265e-10,
02318
              1.14e-10, 1.019e-10, 9.279e-11, 8.283e-11, 7.458e-11, 6.668e-11
02319
02320
              5.976e-11, 5.33e-11, 4.794e-11, 4.289e-11, 3.841e-11, 3.467e-11,
              3.13e-11,\ 2.832e-11,\ 2.582e-11,\ 2.356e-11,\ 2.152e-11,\ 1.97e-11,
02321
              1.808e-11, 1.664e-11, 1.539e-11, 1.434e-11, 1.344e-11, 1.269e-11, 1.209e-11, 1.162e-11, 1.129e-11, 1.108e-11, 1.099e-11, 1.103e-11,
02322
02323
02324
              1.119e-11, 1.148e-11, 1.193e-11, 1.252e-11, 1.329e-11, 1.421e-11,
02325
              1.555e-11, 1.685e-11, 1.839e-11, 2.054e-11, 2.317e-11, 2.571e-11,
02326
              2.839e-11, 3.171e-11, 3.49e-11, 3.886e-11, 4.287e-11, 4.645e-11,
             5.047e-11, 5.592e-11, 6.109e-11, 6.628e-11, 7.381e-11, 8.088e-11, 8.966e-11, 1.045e-10, 1.12e-10, 1.287e-10, 1.486e-10, 1.662e-10, 1.866e-10, 2.133e-10, 2.524e-10, 2.776e-10, 3.204e-10, 3.559e-10,
02327
02328
02329
              4.028e-10, 4.448e-10, 4.882e-10, 5.244e-10, 5.605e-10, 6.018e-10,
02331
              6.328e-10, 6.579e-10, 6.541e-10,
                                                            7.024e-10, 7.074e-10, 7.068e-10,
02332
              7.009e-10, 6.698e-10, 6.545e-10, 6.209e-10, 5.834e-10, 5.412e-10,
02333
              5.001e-10, 4.231e-10, 3.727e-10, 3.211e-10, 2.833e-10, 2.447e-10,
              2.097e-10, 1.843e-10, 1.639e-10, 1.449e-10, 1.27e-10, 1.161e-10,
02334
              1.033e-10, 9.282e-11, 8.407e-11, 7.639e-11, 7.023e-11, 6.474e-11, 6.142e-11, 5.76e-11, 5.568e-11, 5.472e-11, 5.39e-11, 5.455e-11, 5.54e-11, 5.587e-11, 6.23e-11, 6.49e-11, 6.868e-11, 7.382e-11,
02335
02337
             8.022e-11, 8.372e-11, 9.243e-11, 1.004e-10, 1.062e-10, 1.13e-10, 1.176e-10, 1.244e-10, 1.279e-10, 1.298e-10, 1.302e-10, 1.312e-10,
02338
02339
              1.295e-10, 1.244e-10, 1.211e-10, 1.167e-10, 1.098e-10, 9.927e-11,
02340
             8.854e-11, 8.011e-11, 7.182e-11, 5.923e-11, 5.212e-11, 4.453e-11, 3.832e-11, 3.371e-11, 2.987e-11, 2.651e-11, 2.354e-11, 2.093e-11,
02341
02342
              1.863e-11, 1.662e-11, 1.486e-11, 1.331e-11, 1.193e-11, 1.071e-11, 9.628e-12, 8.66e-12, 7.801e-12, 7.031e-12, 6.347e-12, 5.733e-12,
02343
02344
02345
              5.182e-12, 4.695e-12, 4.26e-12, 3.874e-12, 3.533e-12, 3.235e-12,
              2.979e-12, 2.76e-12, 2.579e-12, 2.432e-12, 2.321e-12, 2.246e-12,
02346
              2.205e-12, 2.196e-12, 2.223e-12, 2.288e-12, 2.387e-12, 2.525e-12,
02347
02348
              2.704e-12, 2.925e-12, 3.191e-12, 3.508e-12, 3.876e-12, 4.303e-12,
              4.793e-12, 5.347e-12, 5.978e-12, 6.682e-12, 7.467e-12, 8.34e-12,
              9.293e-12, 1.035e-11, 1.152e-11, 1.285e-11, 1.428e-11, 1.586e-11,
02350
02351
              1.764e-11, 1.972e-11, 2.214e-11, 2.478e-11, 2.776e-11, 3.151e-11,
              3.591e-11, 4.103e-11, 4.66e-11, 5.395e-11, 6.306e-11, 7.172e-11, 8.358e-11, 9.67e-11, 1.11e-10, 1.325e-10, 1.494e-10, 1.736e-10,
02352
02353
              2.007e-10, 2.296e-10, 2.608e-10, 3.004e-10, 3.361e-10, 3.727e-10,
02354
              4.373e-10, 4.838e-10, 5.483e-10, 6.006e-10, 6.535e-10, 6.899e-10,
              7.687e-10, 8.444e-10, 8.798e-10, 9.135e-10, 9.532e-10, 9.757e-10,
02356
             7.507e-10, 3.444e-10, 3.75e-10, 3.135e-10, 3.32e-10, 3.75re-10, 9.968e-10, 1.006e-9, 9.949e-10, 9.789e-10, 9.564e-10, 9.215e-10, 8.51e-10, 8.394e-10, 7.707e-10, 7.152e-10, 6.274e-10, 5.598e-10, 5.028e-10, 4.3e-10, 3.71e-10, 3.245e-10, 2.809e-10, 2.461e-10, 2.154e-10, 1.91e-10, 1.685e-10, 1.487e-10, 1.313e-10, 1.163e-10, 1.031e-10, 9.172e-11, 8.221e-11, 7.382e-11, 6.693e-11, 6.079e-11,
02357
02358
02359
02360
02361
              5.581e-11, 5.167e-11, 4.81e-11, 4.506e-11, 4.255e-11, 4.083e-11, 3.949e-11, 3.881e-11, 3.861e-11, 3.858e-11, 3.951e-11, 4.045e-11,
02362
02363
02364
              4.24e-11, 4.487e-11, 4.806e-11, 5.133e-11, 5.518e-11, 5.919e-11,
              6.533e-11, 7.031e-11, 7.762e-11, 8.305e-11, 9.252e-11, 9.727e-11, 1.045e-10, 1.117e-10, 1.2e-10, 1.275e-10, 1.341e-10, 1.362e-10, 1.438e-10, 1.45e-10, 1.455e-10, 1.455e-10, 1.434e-10, 1.381e-10,
02365
02366
02367
02368
              1.301e-10, 1.276e-10, 1.163e-10, 1.089e-10, 9.911e-11, 8.943e-11,
              7.618e-11, 6.424e-11, 5.717e-11, 4.866e-11, 4.257e-11, 3.773e-11,
02369
02370
              3.331e-11, 2.958e-11, 2.629e-11, 2.316e-11, 2.073e-11, 1.841e-11,
             1.635e-11, 1.464e-11, 1.31e-11, 1.16e-11, 1.047e-11, 9.408e-12, 8.414e-12, 7.521e-12, 6.705e-12, 5.993e-12, 5.371e-12, 4.815e-12,
02371
02372
02373
              4.338e-12, 3.921e-12, 3.567e-12, 3.265e-12, 3.01e-12, 2.795e-12,
              2.613e-12, 2.464e-12, 2.346e-12, 2.256e-12, 2.195e-12, 2.165e-12,
02374
02375
              2.166e-12, 2.198e-12, 2.262e-12, 2.364e-12, 2.502e-12, 2.682e-12,
02376
              2.908e-12, 3.187e-12, 3.533e-12, 3.946e-12, 4.418e-12, 5.013e-12,
              5.708e-12, 6.379e-12, 7.43e-12, 8.39e-12, 9.51e-12, 1.078e-11, 1.259e-11, 1.438e-11, 1.63e-11, 1.814e-11, 2.055e-11, 2.348e-11, 2.664e-11, 2.956e-11, 3.3e-11, 3.677e-11, 4.032e-11, 4.494e-11, 4.951e-11, 5.452e-11, 6.014e-11, 6.5e-11, 6.915e-11, 7.45e-11,
02377
02378
02379
              7.971e-11, 8.468e-11, 8.726e-11, 8.995e-11, 9.182e-11, 9.509e-11, 9.333e-11, 9.386e-11, 9.457e-11, 9.21e-11, 9.019e-11, 8.68e-11,
02381
02382
              8.298e-11, 7.947e-11, 7.46e-11, 7.082e-11, 6.132e-11, 5.855e-11,
02383
              5.073e-11, 4.464e-11, 3.825e-11, 3.375e-11, 2.911e-11, 2.535e-11, 2.16e-11, 1.907e-11, 1.665e-11, 1.463e-11, 1.291e-11, 1.133e-11,
02384
02385
```

```
9.997e-12, 8.836e-12, 7.839e-12, 6.943e-12, 6.254e-12, 5.6e-12,
            5.029e-12, 4.529e-12, 4.102e-12, 3.737e-12, 3.428e-12, 3.169e-12, 2.959e-12, 2.798e-12, 2.675e-12, 2.582e-12, 2.644e-12, 2.557e-12,
02387
02388
02389
            2.614e-12, 2.717e-12, 2.874e-12, 3.056e-12, 3.187e-12, 3.631e-12,
            3.979e-12, 4.248e-12, 4.817e-12, 5.266e-12, 5.836e-12, 6.365e-12, 6.807e-12, 7.47e-12, 7.951e-12, 8.636e-12, 8.972e-12, 9.314e-12,
02390
02391
             9.445e-12, 1.003e-11, 1.013e-11, 9.937e-12, 9.729e-12, 9.064e-12,
             9.119e-12, 9.124e-12, 8.704e-12, 8.078e-12, 7.47e-12, 6.329e-12,
02393
02394
             5.674e-12, 4.808e-12, 4.119e-12, 3.554e-12, 3.103e-12, 2.731e-12,
            1.34e-12, 1.25e-12, 1.926e-12, 1.737e-12, 1.578e-12, 1.447e-12, 1.34e-12, 1.255e-12, 1.191e-12, 1.146e-12, 1.121e-12, 1.114e-12, 1.126e-12, 1.156e-12, 1.207e-12, 1.278e-12, 1.372e-12, 1.49e-12, 1.633e-12, 1.805e-12, 2.01e-12, 2.249e-12, 2.528e-12, 2.852e-12,
02395
02396
02397
02398
             3.228e-12, 3.658e-12, 4.153e-12, 4.728e-12, 5.394e-12, 6.176e-12,
02399
02400
            7.126e-12, 8.188e-12, 9.328e-12, 1.103e-11, 1.276e-11, 1.417e-11,
            1.615e-11, 1.84e-11, 2.155e-11, 2.429e-11, 2.826e-11, 3.222e-11, 3.664e-11, 4.14e-11, 4.906e-11, 5.536e-11, 6.327e-11, 7.088e-11,
02401
02402
            8.316e-11, 9.242e-11, 1.07e-10, 1.223e-10, 1.341e-10, 1.553e-10,
02403
             1.703e-10, 1.9e-10, 2.022e-10, 2.233e-10, 2.345e-10, 2.438e-10,
02405
             2.546e-10, 2.599e-10, 2.661e-10, 2.703e-10, 2.686e-10, 2.662e-10,
             2.56e-10, 2.552e-10, 2.378e-10, 2.252e-10, 2.146e-10, 1.885e-10,
02406
02407
             1.668e-10, 1.441e-10, 1.295e-10, 1.119e-10, 9.893e-11, 8.687e-11,
02408
             7.678e-11, 6.685e-11, 5.879e-11, 5.127e-11, 4.505e-11, 3.997e-11,
02409
            3.511e-11
02410
02411
02412
          static double h2ofrn[2001] = { .01095, .01126, .01205, .01322, .0143,
02/13
            .01506, .01548, .01534, .01486, .01373, .01262, .01134, .01001,
            .008702, .007475, .006481, .00548, .0046, .003833, .00311, .002543, .002049, .00168, .001374, .001046, 8.193e-4, 6.267e-4, 4.968e-4, 3.924e-4, 2.983e-4, 2.477e-4, 1.997e-4, 1.596e-4, 1.331e-4, 1.061e-4, 8.942e-5, 7.168e-5, 5.887e-5, 4.848e-5,
02414
02415
02416
02417
             3.817e-5, 3.17e-5, 2.579e-5, 2.162e-5, 1.768e-5, 1.49e-5,
02418
02419
            1.231e-5, 1.013e-5, 8.555e-6, 7.328e-6, 6.148e-6, 5.207e-6,
            1.387e-6, 3.741e-6, 3.22e-6, 2.753e-6, 2.346e-6, 1.985e-6, 1.716e-6, 1.475e-6, 1.286e-6, 1.122e-6, 9.661e-7, 8.284e-7, 7.057e-7, 6.119e-7, 5.29e-7, 4.571e-7, 3.948e-7, 3.432e-7, 2.983e-7, 2.589e-7, 2.265e-7, 1.976e-7, 1.704e-7, 1.456e-7,
02420
02421
02422
02424
             1.26e-7, 1.101e-7, 9.648e-8, 8.415e-8, 7.34e-8, 6.441e-8,
             5.643e-8, 4.94e-8, 4.276e-8, 3.703e-8, 3.227e-8, 2.825e-8
02425
02426
            2.478e-8, 2.174e-8, 1.898e-8, 1.664e-8, 1.458e-8, 1.278e-8,
            1.126e-8, 9.891e-9, 8.709e-9, 7.652e-9, 6.759e-9, 5.975e-9, 5.31e-9, 4.728e-9, 4.214e-9, 3.792e-9, 3.463e-9, 3.226e-9, 2.992e-9, 2.813e-9, 2.749e-9, 2.809e-9, 2.913e-9, 3.037e-9,
02427
02428
02429
             3.413e-9, 3.738e-9, 4.189e-9, 4.808e-9, 5.978e-9, 7.088e-9,
02430
02431
             8.071e-9, 9.61e-9, 1.21e-8, 1.5e-8, 1.764e-8, 2.221e-8, 2.898e-8,
            3.948e-8, 5.068e-8, 6.227e-8, 7.898e-8, 1.033e-7, 1.437e-7, 1.889e-7, 2.589e-7, 3.59e-7, 4.971e-7, 7.156e-7, 9.983e-7,
02432
02433
             1.381e-6, 1.929e-6, 2.591e-6, 3.453e-6, 4.57e-6, 5.93e-6,
02434
             7.552e-6, 9.556e-6, 1.183e-5, 1.425e-5, 1.681e-5, 1.978e-5,
02435
             2.335e-5, 2.668e-5, 3.022e-5, 3.371e-5, 3.715e-5, 3.967e-5,
             4.06e-5, 4.01e-5, 3.809e-5, 3.491e-5, 3.155e-5, 2.848e-5,
02437
02438
            2.678e-5, 2.66e-5, 2.811e-5, 3.071e-5, 3.294e-5, 3.459e-5,
02439
             3.569e-5, 3.56e-5, 3.434e-5, 3.186e-5, 2.916e-5, 2.622e-5,
            2.275e-5, 1.918e-5, 1.62e-5, 1.373e-5, 1.182e-5, 1.006e-5,
02440
            8.556e-6, 7.26e-6, 6.107e-6, 5.034e-6, 4.211e-6, 3.426e-6,
02441
             2.865e-6, 2.446e-6, 1.998e-6, 1.628e-6, 1.242e-6, 1.005e-6,
02442
             7.853e-7, 6.21e-7, 5.071e-7, 4.156e-7, 3.548e-7, 2.825e-7,
02443
            2.261e-7, 1.916e-7, 1.51e-7, 1.279e-7, 1.059e-7, 9.14e-8, 7.707e-8, 6.17e-8, 5.311e-8, 4.263e-8, 3.518e-8, 2.961e-8,
02444
02445
02446
            2.457e-8, 2.119e-8, 1.712e-8, 1.439e-8, 1.201e-8, 1.003e-8,
02447
            8.564e-9, 7.199e-9, 6.184e-9, 5.206e-9, 4.376e-9, 3.708e-9,
02448
            3.157e-9, 2.725e-9, 2.361e-9, 2.074e-9, 1.797e-9, 1.562e-9,
             1.364e-9, 1.196e-9, 1.042e-9, 8.862e-10, 7.648e-10, 6.544e-10,
02449
02450
             5.609e-10, 4.791e-10, 4.108e-10, 3.531e-10, 3.038e-10, 2.618e-10,
02451
            2.268e-10, 1.969e-10, 1.715e-10, 1.496e-10, 1.308e-10, 1.147e-10,
            1.008e-10, 8.894e-11, 7.885e-11, 7.031e-11, 6.355e-11, 5.854e-11, 5.534e-11, 5.466e-11, 5.725e-11, 6.447e-11, 7.943e-11, 1.038e-10,
02452
02453
            1.437e-10, 2.04e-10, 2.901e-10, 4.051e-10, 5.556e-10, 7.314e-10,
02454
             9.291e-10, 1.134e-9, 1.321e-9, 1.482e-9, 1.596e-9, 1.669e-9,
02456
             1.715e-9, 1.762e-9, 1.817e-9, 1.828e-9, 1.848e-9, 1.873e-9,
02457
            1.902e-9, 1.894e-9, 1.864e-9, 1.841e-9, 1.797e-9, 1.704e-9,
02458
            1.559e-9, 1.382e-9, 1.187e-9, 1.001e-9, 8.468e-10, 7.265e-10,
             6.521e-10, 6.381e-10, 6.66e-10, 7.637e-10, 9.705e-10, 1.368e-9,
02459
            1.856e-9, 2.656e-9, 3.954e-9, 5.96e-9, 8.72e-9, 1.247e-8, 1.781e-8, 2.491e-8, 3.311e-8, 4.272e-8, 5.205e-8, 6.268e-8,
02460
02461
             7.337e-8, 8.277e-8, 9.185e-8, 1.004e-7, 1.091e-7, 1.159e-7,
02462
02463
             1.188e-7, 1.175e-7, 1.124e-7, 1.033e-7, 9.381e-8, 8.501e-8,
            7.956e-8, 7.894e-8, 8.331e-8, 9.102e-8, 9.836e-8, 1.035e-7, 1.064e-7, 1.06e-7, 1.032e-7, 9.808e-8, 9.139e-8, 8.442e-8,
02464
02465
             7.641e-8, 6.881e-8, 6.161e-8, 5.404e-8, 4.804e-8, 4.446e-8,
02466
             4.328e-8, 4.259e-8, 4.421e-8, 4.673e-8, 4.985e-8, 5.335e-8,
             5.796e-8, 6.542e-8, 7.714e-8, 8.827e-8, 1.04e-7, 1.238e-7,
02468
02469
            1.499e-7, 1.829e-7, 2.222e-7, 2.689e-7, 3.303e-7, 3.981e-7,
02470
             4.84e-7, 5.91e-7, 7.363e-7, 9.087e-7, 1.139e-6, 1.455e-6,
            1.866e-6, 2.44e-6, 3.115e-6, 3.941e-6, 4.891e-6, 5.992e-6, 7.111e-6, 8.296e-6, 9.21e-6, 9.987e-6, 1.044e-5, 1.073e-5,
02471
02472
```

```
1.092e-5, 1.106e-5, 1.138e-5, 1.171e-5, 1.186e-5, 1.186e-5,
            1.179e-5, 1.166e-5, 1.151e-5, 1.16e-5, 1.197e-5, 1.241e-5, 1.268e-5, 1.26e-5, 1.184e-5, 1.063e-5, 9.204e-6, 7.584e-6,
02474
02475
02476
            6.053e-6, 4.482e-6, 3.252e-6, 2.337e-6, 1.662e-6, 1.18e-6,
            8.15e-7, 5.95e-7, 4.354e-7, 3.302e-7, 2.494e-7, 1.93e-7, 1.545e-7, 1.25e-7, 1.039e-7, 8.602e-8, 7.127e-8, 5.897e-8,
02477
02478
             4.838e-8, 4.018e-8, 3.28e-8, 2.72e-8, 2.307e-8, 1.972e-8,
            1.654e-8, 1.421e-8, 1.174e-8, 1.004e-8, 8.739e-9, 7.358e-9,
02480
            6.242e-9, 5.303e-9, 4.567e-9, 3.94e-9, 3.375e-9, 2.864e-9, 2.422e-9, 2.057e-9, 1.75e-9, 1.505e-9, 1.294e-9, 1.101e-9, 9.401e-10, 8.018e-10, 6.903e-10, 5.965e-10, 5.087e-10, 4.364e-10,
02/81
02482
02483
02484
            3.759e-10, 3.247e-10, 2.809e-10, 2.438e-10, 2.123e-10, 1.853e-10,
02485
            1.622e-10, 1.426e-10, 1.26e-10, 1.125e-10, 1.022e-10, 9.582e-11,
            9.388e-11, 9.801e-11, 1.08e-10, 1.276e-10, 1.551e-10, 1.903e-10,
02486
02487
            2.291e-10, 2.724e-10, 3.117e-10, 3.4e-10, 3.562e-10, 3.625e-10,
02488
            3.619e-10, 3.429e-10, 3.221e-10, 2.943e-10, 2.645e-10, 2.338e-10,
            2.062e-10, 1.901e-10, 1.814e-10, 1.827e-10, 1.906e-10, 1.984e-10,
02489
            2.04e-10, 2.068e-10, 2.075e-10, 2.018e-10, 1.959e-10, 1.897e-10, 1.852e-10, 1.791e-10, 1.696e-10, 1.634e-10, 1.598e-10, 1.561e-10,
02490
             1.518e-10, 1.443e-10, 1.377e-10, 1.346e-10, 1.342e-10, 1.375e-10,
02492
            1.525e-10, 1.767e-10, 2.108e-10, 2.524e-10, 2.981e-10,
02493
02494
            4.262e-10, 5.326e-10, 6.646e-10, 8.321e-10, 1.069e-9, 1.386e-9,
            1.743e-9, 2.216e-9, 2.808e-9, 3.585e-9, 4.552e-9, 5.907e-9,
02495
            7.611e-9, 9.774e-9, 1.255e-8, 1.666e-8, 2.279e-8, 3.221e-8, 4.531e-8, 6.4e-8, 9.187e-8, 1.295e-7, 1.825e-7, 2.431e-7,
02496
02497
            3.181e-7, 4.009e-7, 4.941e-7, 5.88e-7, 6.623e-7, 7.155e-7, 7.451e-7, 7.594e-7, 7.541e-7, 7.467e-7, 7.527e-7, 7.935e-7,
02499
02500
            8.461e-7, 8.954e-7, 9.364e-7, 9.843e-7, 1.024e-6, 1.05e-6,
02501
            1.059e-6, 1.074e-6, 1.072e-6, 1.043e-6, 9.789e-7, 8.803e-7,
02502
             7.662e-7, 6.378e-7, 5.133e-7, 3.958e-7, 2.914e-7, 2.144e-7
            1.57e-7, 1.14e-7, 8.47e-8, 6.2e-8, 4.657e-8, 3.559e-8, 2.813e-8,
02503
            2.222e-8, 1.769e-8, 1.391e-8, 1.125e-8, 9.186e-9, 7.704e-9,
02505
            6.447e-9, 5.381e-9, 4.442e-9, 3.669e-9, 3.057e-9, 2.564e-9,
02506
            2.153e-9, 1.784e-9, 1.499e-9, 1.281e-9, 1.082e-9, 9.304e-10,
            8.169e-10, 6.856e-10, 5.866e-10, 5.043e-10, 4.336e-10, 3.731e-10, 3.175e-10, 2.745e-10, 2.374e-10, 2.007e-10, 1.737e-10, 1.508e-10,
02507
02508
            1.302e-10, 1.13e-10, 9.672e-11, 8.375e-11, 7.265e-11, 6.244e-11, 5.343e-11, 4.654e-11, 3.975e-11, 3.488e-11, 3.097e-11, 2.834e-11,
02509
02511
            2.649e-11, 2.519e-11, 2.462e-11, 2.443e-11, 2.44e-11, 2.398e-11,
02512
            2.306e-11, 2.183e-11, 2.021e-11, 1.821e-11, 1.599e-11, 1.403e-11,
02513
            1.196e-11, 1.023e-11, 8.728e-12, 7.606e-12, 6.941e-12, 6.545e-12,
            6.484e-12, 6.6e-12, 6.718e-12, 6.785e-12, 6.746e-12, 6.724e-12, 6.764e-12, 6.995e-12, 7.144e-12, 7.32e-12, 7.33e-12, 7.208e-12, 6.789e-12, 6.09e-12, 5.337e-12, 4.62e-12, 4.037e-12, 3.574e-12,
02514
02515
            3.311e-12, 3.346e-12, 3.566e-12, 3.836e-12, 4.076e-12, 4.351e-12, 4.691e-12, 5.114e-12, 5.427e-12, 6.167e-12, 7.436e-12, 8.842e-12,
02517
02518
02519
            1.038e-11, 1.249e-11, 1.54e-11, 1.915e-11, 2.48e-11, 3.256e-11,
            4.339e-11, 5.611e-11, 7.519e-11, 1.037e-10, 1.409e-10, 1.883e-10, 2.503e-10, 3.38e-10, 4.468e-10, 5.801e-10, 7.335e-10, 8.98e-10,
02520
02521
            1.11e-9, 1.363e-9, 1.677e-9, 2.104e-9, 2.681e-9, 3.531e-9,
02522
             4.621e-9, 6.106e-9, 8.154e-9, 1.046e-8, 1.312e-8, 1.607e-8,
            1.948e-8, 2.266e-8, 2.495e-8, 2.655e-8, 2.739e-8, 2.739e-8,
02524
02525
            2.662e-8, 2.589e-8, 2.59e-8, 2.664e-8, 2.833e-8, 3.023e-8,
02526
            3.305e-8, 3.558e-8, 3.793e-8, 3.961e-8, 4.056e-8, 4.102e-8,
02527
            4.025e-8, 3.917e-8, 3.706e-8, 3.493e-8, 3.249e-8, 3.096e-8,
            3.011e-8, 3.111e-8, 3.395e-8, 3.958e-8, 4.875e-8, 6.066e-8,
02528
            7.915e-8, 1.011e-7, 1.3e-7, 1.622e-7, 2.003e-7, 2.448e-7, 2.863e-7, 3.317e-7, 3.655e-7, 3.96e-7, 4.098e-7, 4.168e-7
02530
02531
            4.198e-7, 4.207e-7, 4.289e-7, 4.384e-7, 4.471e-7, 4.524e-7
            4.574e-7, 4.633e-7, 4.785e-7, 5.028e-7, 5.371e-7, 5.727e-7,
02532
            5.955e-7, 5.998e-7, 5.669e-7, 5.082e-7, 4.397e-7, 3.596e-7,
02533
            2.814e-7, 2.074e-7, 1.486e-7, 1.057e-7, 7.25e-8, 4.946e-8,
02534
            3.43e-8, 2.447e-8, 1.793e-8, 1.375e-8, 1.096e-8, 9.091e-9,
            7.709e-9, 6.631e-9, 5.714e-9, 4.886e-9, 4.205e-9, 3.575e-9, 3.07e-9, 2.631e-9, 2.284e-9, 2.002e-9, 1.745e-9, 1.509e-9,
02536
02537
02538
            1.284e-9, 1.084e-9, 9.163e-10, 7.663e-10, 6.346e-10, 5.283e-10,
02539
            4.354e-10, 3.59e-10, 2.982e-10, 2.455e-10, 2.033e-10, 1.696e-10, 1.432e-10, 1.211e-10, 1.02e-10, 8.702e-11, 7.38e-11, 6.293e-11,
02540
            5.343e-11, 4.532e-11, 3.907e-11, 3.365e-11, 2.945e-11, 2.558e-11,
02541
            2.192e-11, 1.895e-11, 1.636e-11, 1.42e-11, 1.28e-11, 1.063e-11, 9.348e-12, 8.2e-12, 7.231e-12, 6.43e-12, 5.702e-12, 5.052e-12,
02543
            4.469e-12, 4e-12, 3.679e-12, 3.387e-12, 3.197e-12, 3.158e-12,
02544
02545
            3.327e-12, 3.675e-12, 4.292e-12, 5.437e-12, 7.197e-12, 1.008e-11,
02546
            1.437e-11, 2.035e-11, 2.905e-11, 4.062e-11, 5.528e-11, 7.177e-11,
            9.064e-11, 1.109e-10, 1.297e-10, 1.473e-10, 1.652e-10, 1.851e-10,
02547
            2.079e-10, 2.313e-10, 2.619e-10, 2.958e-10, 3.352e-10, 3.796e-10,
02548
02549
            4.295e-10, 4.923e-10, 5.49e-10, 5.998e-10, 6.388e-10, 6.645e-10,
            6.712e-10, 6.549e-10, 6.38e-10, 6.255e-10, 6.253e-10, 6.459e-10, 6.977e-10, 7.59e-10, 8.242e-10, 8.92e-10, 9.403e-10, 9.701e-10,
02550
02551
            9.483e-10, 9.135e-10, 8.617e-10, 7.921e-10, 7.168e-10, 6.382e-10,
02552
02553
            5.677e-10, 5.045e-10, 4.572e-10, 4.312e-10, 4.145e-10, 4.192e-10,
             4.541e-10, 5.368e-10, 6.771e-10, 8.962e-10, 1.21e-9, 1.659e-9,
            2.33e-9, 3.249e-9, 4.495e-9, 5.923e-9, 7.642e-9, 9.607e-9, 1.178e-8, 1.399e-8, 1.584e-8, 1.73e-8, 1.816e-8, 1.87e-8,
02555
02556
02557
            1.868e-8, 1.87e-8, 1.884e-8, 1.99e-8, 2.15e-8, 2.258e-8,
            2.364e-8, 2.473e-8, 2.602e-8, 2.689e-8, 2.731e-8, 2.816e-8, 2.859e-8, 2.839e-8, 2.703e-8, 2.451e-8, 2.149e-8, 1.787e-8,
02558
02559
```

```
1.449e-8, 1.111e-8, 8.282e-9, 6.121e-9, 4.494e-9, 3.367e-9,
              2.487e-9, 1.885e-9, 1.503e-9, 1.249e-9, 1.074e-9, 9.427e-10, 8.439e-10, 7.563e-10, 6.772e-10, 6.002e-10, 5.254e-10, 4.588e-10,
02561
02562
02563
              3.977e-10,\ 3.449e-10,\ 3.003e-10,\ 2.624e-10,\ 2.335e-10,\ 2.04e-10,
              1.771e-10, 1.534e-10, 1.296e-10, 1.097e-10, 9.173e-11, 7.73e-11, 6.547e-11, 5.191e-11, 4.198e-11, 3.361e-11, 2.732e-11, 2.244e-11,
02564
02565
              1.791e-11, 1.509e-11, 1.243e-11, 1.035e-11, 8.969e-12, 7.394e-12,
              6.323e-12, 5.282e-12, 4.543e-12, 3.752e-12, 3.14e-12, 2.6e-12,
02567
02568
              2.194e-12, 1.825e-12, 1.511e-12, 1.245e-12, 1.024e-12, 8.539e-13,
             7.227e-13, 6.102e-13, 5.189e-13, 4.43e-13, 3.774e-13, 3.236e-13, 2.8e-13, 2.444e-13, 2.156e-13, 1.932e-13, 1.775e-13, 1.695e-13, 1.672e-13, 1.704e-13, 1.825e-13, 2.087e-13, 2.614e-13, 3.377e-13, 4.817e-13, 6.989e-13, 1.062e-12, 1.562e-12, 2.288e-12, 3.295e-12,
02569
02570
02571
              4.55e-12, 5.965e-12, 7.546e-12, 9.395e-12, 1.103e-11, 1.228e-11,
02573
02574
              1.318e-11, 1.38e-11, 1.421e-11, 1.39e-11, 1.358e-11, 1.336e-11,
02575
              1.342e-11, 1.356e-11, 1.424e-11, 1.552e-11, 1.73e-11, 1.951e-11,
              2.128e-11, 2.249e-11, 2.277e-11, 2.226e-11, 2.111e-11, 1.922e-11,
02576
02577
              1.775e-11, 1.661e-11, 1.547e-11, 1.446e-11, 1.323e-11, 1.21e-11,
              1.054e-11, 9.283e-12, 8.671e-12, 8.67e-12, 9.429e-12, 1.062e-11,
              1.255e-11, 1.506e-11, 1.818e-11, 2.26e-11, 2.831e-11, 3.723e-11,
02579
02580
              5.092e-11, 6.968e-11, 9.826e-11, 1.349e-10, 1.87e-10, 2.58e-10,
02581
              3.43 e^{-10},\ 4.424 e^{-10},\ 5.521 e^{-10},\ 6.812 e^{-10},\ 8.064 e^{-10},\ 9.109 e^{-10},
             9.839e-10, 1.028e-9, 1.044e-9, 1.029e-9, 1.005e-9, 1.002e-9, 1.038e-9, 1.122e-9, 1.233e-9, 1.372e-9, 1.524e-9, 1.665e-9, 1.804e-9, 1.908e-9, 2.015e-9, 2.117e-9, 2.219e-9, 2.336e-9,
02582
02583
02584
              2.531e-9, 2.805e-9, 3.189e-9, 3.617e-9, 4.208e-9, 4.911e-9,
              5.619e-9, 6.469e-9, 7.188e-9, 7.957e-9, 8.503e-9, 9.028e-9,
02586
02587
              9.571e-9, 9.99e-9, 1.055e-8, 1.102e-8, 1.132e-8, 1.141e-8,
             1.145e-8, 1.145e-8, 1.176e-8, 1.224e-8, 1.304e-8, 1.388e-8, 1.445e-8, 1.453e-8, 1.368e-8, 1.22e-8, 1.042e-8, 8.404e-9, 6.403e-9, 4.643e-9, 3.325e-9, 2.335e-9, 1.638e-9, 1.19e-9,
02588
02589
02590
              9.161e-10, 7.412e-10, 6.226e-10, 5.516e-10, 5.068e-10, 4.831e-10,
              4.856e-10, 5.162e-10, 5.785e-10, 6.539e-10, 7.485e-10,
                                                                                          8.565e-10,
02592
02593
              9.534e-10, 1.052e-9, 1.115e-9, 1.173e-9, 1.203e-9, 1.224e-9,
              1.243e-9, 1.248e-9, 1.261e-9, 1.265e-9, 1.25e-9, 1.217e-9, 1.176e-9, 1.145e-9, 1.153e-9, 1.199e-9, 1.278e-9, 1.366e-9,
02594
02595
              1.426e-9, 1.444e-9, 1.365e-9, 1.224e-9, 1.051e-9, 8.539e-10, 6.564e-10, 4.751e-10, 3.404e-10, 2.377e-10, 1.631e-10, 1.114e-10,
02596
02598
              7.87e-11, 5.793e-11, 4.284e-11, 3.3e-11, 2.62e-11, 2.152e-11,
02599
              1.777e-11, 1.496e-11, 1.242e-11, 1.037e-11, 8.725e-12, 7.004e-12,
02600
              5.718e-12, 4.769e-12, 3.952e-12, 3.336e-12, 2.712e-12, 2.213e-12,
              1.803e-12, 1.492e-12, 1.236e-12, 1.006e-12, 8.384e-13, 7.063e-13, 5.879e-13, 4.93e-13, 4.171e-13, 3.569e-13, 3.083e-13, 2.688e-13, 2.333e-13, 2.035e-13, 1.82e-13, 1.682e-13, 1.635e-13, 1.674e-13,
02601
02602
02603
              1.769e-13, 2.022e-13, 2.485e-13, 3.127e-13, 4.25e-13, 5.928e-13,
02605
              8.514e-13, 1.236e-12, 1.701e-12, 2.392e-12, 3.231e-12, 4.35e-12,
02606
              5.559e-12, 6.915e-12, 8.519e-12, 1.013e-11, 1.146e-11, 1.24e-11,
02607
              1.305e-11, 1.333e-11, 1.318e-11, 1.263e-11, 1.238e-11, 1.244e-11,
              1.305e-11, 1.432e-11, 1.623e-11, 1.846e-11, 2.09e-11, 2.328e-11, 2.526e-11, 2.637e-11, 2.702e-11, 2.794e-11, 2.889e-11, 2.989e-11, 3.231e-11, 3.68e-11, 4.375e-11, 5.504e-11, 7.159e-11, 9.502e-11,
02608
02609
              1.279e-10, 1.645e-10, 2.098e-10, 2.618e-10, 3.189e-10, 3.79e-10,
02611
02612
              4.303e-10, 4.753e-10, 5.027e-10, 5.221e-10, 5.293e-10, 5.346e-10,
              5.467e-10, 5.796e-10, 6.2e-10, 6.454e-10, 6.705e-10, 6.925e-10, 7.233e-10, 7.35e-10, 7.538e-10, 7.861e-10, 8.077e-10, 8.132e-10,
02613
02614
              7.749e-10, 7.036e-10, 6.143e-10, 5.093e-10, 4.089e-10, 3.092e-10, 2.299e-10, 1.705e-10, 1.277e-10, 9.723e-11, 7.533e-11, 6.126e-11, 5.154e-11, 4.428e-11, 3.913e-11, 3.521e-11, 3.297e-11, 3.275e-11,
02615
02617
              3.46e-11, 3.798e-11, 4.251e-11, 4.745e-11, 5.232e-11, 5.606e-11, 5.82e-11, 5.88e-11, 5.79e-11, 5.661e-11, 5.491e-11, 5.366e-11, 5.341e-11, 5.353e-11, 5.336e-11, 5.293e-11, 5.248e-11, 5.235e-11,
02618
02619
02620
02621
              5.208e-11, 5.322e-11, 5.521e-11, 5.725e-11, 5.827e-11, 5.685e-11,
              5.245e-11, 4.612e-11, 3.884e-11, 3.129e-11, 2.404e-11, 1.732e-11,
              1.223e-11, 8.574e-12, 5.888e-12, 3.986e-12, 2.732e-12, 1.948e-12,
02623
02624
              1.414e-12, 1.061e-12, 8.298e-13, 6.612e-13, 5.413e-13, 4.472e-13,
02625
              3.772e-13, 3.181e-13, 2.645e-13, 2.171e-13, 1.778e-13, 1.464e-13,
              1.183e-13, 9.637e-14, 7.991e-14, 6.668e-14, 5.57e-14, 4.663e-14, 3.848e-14, 3.233e-14, 2.706e-14, 2.284e-14, 1.944e-14, 1.664e-14,
02626
02627
              1.43e-14, 1.233e-14, 1.066e-14, 9.234e-15, 8.023e-15, 6.993e-15,
02628
              6.119e-15, 5.384e-15, 4.774e-15, 4.283e-15, 3.916e-15, 3.695e-15,
              3.682e-15, 4.004e-15, 4.912e-15, 6.853e-15, 1.056e-14, 1.712e-14,
02630
02631
              2.804e-14, 4.516e-14, 7.113e-14, 1.084e-13, 1.426e-13, 1.734e-13,
02632
              1.978e-13, 2.194e-13, 2.388e-13, 2.489e-13, 2.626e-13, 2.865e-13,
              3.105e-13, 3.387e-13, 3.652e-13, 3.984e-13, 4.398e-13, 4.906e-13, 5.55e-13, 6.517e-13, 7.813e-13, 9.272e-13, 1.164e-12, 1.434e-12, 1.849e-12, 2.524e-12, 3.328e-12, 4.523e-12, 6.108e-12, 8.207e-12,
02633
02634
02635
              1.122e-11, 1.477e-11, 1.9e-11, 2.412e-11, 2.984e-11, 3.68e-11,
02636
02637
              4.353e-11, 4.963e-11, 5.478e-11, 5.903e-11, 6.233e-11, 6.483e-11,
              6.904e-11, 7.569e-11, 8.719e-11, 1.048e-10, 1.278e-10, 1.557e-10, 1.869e-10, 2.218e-10, 2.61e-10, 2.975e-10, 3.371e-10, 3.746e-10,
02638
02639
              1.055e-10, 4.336e-10, 4.503e-10, 4.701e-10, 4.8e-10, 4.917e-10, 5.038e-10, 5.128e-10, 5.143e-10, 5.071e-10, 5.019e-10, 5.025e-10,
02640
02641
              5.183e-10, 5.496e-10, 5.877e-10, 6.235e-10, 6.42e-10, 6.234e-10,
02642
02643
              5.698e-10, 4.916e-10, 4.022e-10, 3.126e-10, 2.282e-10, 1.639e-10,
02644
              1.142 e^{-10},\ 7.919 e^{-11},\ 5.69 e^{-11},\ 4.313 e^{-11},\ 3.413 e^{-11},\ 2.807 e^{-11},
              2.41e-11, 2.166e-11, 2.024e-11, 1.946e-11, 1.929e-11, 1.963e-11, 2.035e-11, 2.162e-11, 2.305e-11, 2.493e-11, 2.748e-11, 3.048e-11,
02645
02646
```

```
3.413e-11, 3.754e-11, 4.155e-11, 4.635e-11, 5.11e-11, 5.734e-11,
           6.338e-11, 6.99e-11, 7.611e-11, 8.125e-11, 8.654e-11, 8.951e-11, 9.182e-11, 9.31e-11, 9.273e-11, 9.094e-11, 8.849e-11, 8.662e-11,
02648
02649
02650
           8.67e-11, 8.972e-11, 9.566e-11, 1.025e-10, 1.083e-10, 1.111e-10,
           02651
02652
            4.138e-12, 3.304e-12, 2.784e-12, 2.473e-12, 2.273e-12, 2.186e-12,
           2.118e-12, 2.066e-12, 1.958e-12, 1.818e-12, 1.675e-12, 1.509e-12,
02654
02655
           1.349e-12, 1.171e-12, 9.838e-13, 8.213e-13, 6.765e-13, 5.378e-13,
           4.161e-13, 3.119e-13, 2.279e-13, 1.637e-13, 1.152e-13, 8.112e-14, 5.919e-14, 4.47e-14, 3.492e-14, 2.811e-14, 2.319e-14, 1.948e-14, 1.66e-14, 1.432e-14, 1.251e-14, 1.109e-14, 1.006e-14, 9.45e-15,
02656
02657
02658
            9.384e-15, 1.012e-14, 1.216e-14, 1.636e-14, 2.305e-14, 3.488e-14,
02659
           5.572e-14, 8.479e-14, 1.265e-13, 1.905e-13, 2.73e-13, 3.809e-13,
02660
02661
           4.955e-13, 6.303e-13, 7.861e-13, 9.427e-13, 1.097e-12, 1.212e-12,
02662
           1.328e-12, 1.415e-12, 1.463e-12, 1.495e-12, 1.571e-12, 1.731e-12,
           1.981e-12, 2.387e-12, 2.93e-12, 3.642e-12, 4.584e-12, 5.822e-12,
02663
            7.278e-12, 9.193e-12, 1.135e-11, 1.382e-11, 1.662e-11, 1.958e-11,
02664
           2.286e-11, 2.559e-11, 2.805e-11, 2.988e-11, 3.106e-11, 3.182e-11,
02665
02666
           3.2e-11, 3.258e-11, 3.362e-11, 3.558e-11, 3.688e-11, 3.8e-11,
           3.929e-11, 4.062e-11, 4.186e-11, 4.293e-11, 4.48e-11, 4.643e-11,
02667
02668
           4.704e-11, 4.571e-11, 4.206e-11, 3.715e-11, 3.131e-11, 2.541e-11,
           1.978e-11, 1.508e-11, 1.146e-11, 8.7e-12, 6.603e-12, 5.162e-12,
02669
           4.157e-12, 3.408e-12, 2.829e-12, 2.405e-12, 2.071e-12, 1.826e-12, 1.648e-12, 1.542e-12, 1.489e-12, 1.485e-12, 1.493e-12, 1.545e-12,
02670
02671
           1.637e-12, 1.814e-12, 2.061e-12, 2.312e-12, 2.651e-12, 3.03e-12,
           3.46e-12, 3.901e-12, 4.306e-12, 4.721e-12, 5.008e-12, 5.281e-12,
02673
02674
           5.541e-12, 5.791e-12, 6.115e-12, 6.442e-12, 6.68e-12, 6.791e-12,
           6.831e-12, 6.839e-12, 6.946e-12, 7.128e-12, 7.537e-12, 8.036e-12, 8.392e-12, 8.526e-12, 8.11e-12, 7.325e-12, 6.329e-12, 5.183e-12, 4.081e-12, 2.985e-12, 2.141e-12, 1.492e-12, 1.015e-12, 6.684e-13,
02675
02676
02677
            4.414e-13, 2.987e-13, 2.038e-13, 1.391e-13, 9.86e-14, 7.24e-14,
           5.493e-14, 4.288e-14, 3.427e-14, 2.787e-14, 2.296e-14, 1.909e-14,
02679
02680
           1.598e-14, 1.344e-14, 1.135e-14, 9.616e-15, 8.169e-15, 6.957e-15,
02681
           5.938e-15, 5.08e-15, 4.353e-15, 3.738e-15, 3.217e-15, 2.773e-15,
           2.397e-15, 2.077e-15, 1.805e-15, 1.575e-15, 1.382e-15, 1.221e-15, 1.09e-15, 9.855e-16, 9.068e-16, 8.537e-16, 8.27e-16, 8.29e-16,
02682
02683
           8.634e-16, 9.359e-16, 1.055e-15, 1.233e-15, 1.486e-15, 1.839e-15,
02685
           2.326e-15, 2.998e-15, 3.934e-15, 5.256e-15, 7.164e-15, 9.984e-15,
           1.427e-14, 2.099e-14, 3.196e-14, 5.121e-14, 7.908e-14, 1.131e-13,
02686
02687
           1.602e-13, 2.239e-13, 3.075e-13, 4.134e-13, 5.749e-13, 7.886e-13,
           1.071e-12,\ 1.464e-12,\ 2.032e-12,\ 2.8e-12,\ 3.732e-12,\ 4.996e-12,
02688
           6.483e-12, 8.143e-12, 1.006e-11, 1.238e-11, 1.484e-11, 1.744e-11, 2.02e-11, 2.274e-11, 2.562e-11, 2.848e-11, 3.191e-11, 3.617e-11,
02689
02690
           4.081e-11, 4.577e-11, 4.937e-11, 5.204e-11, 5.401e-11, 5.462e-11, 5.507e-11, 5.51e-11, 5.605e-11, 5.686e-11, 5.739e-11, 5.766e-11,
02692
02693
           5.74e-11, 5.754e-11, 5.761e-11, 5.777e-11, 5.712e-11, 5.51e-11,
02694
           5.088e-11,\ 4.438e-11,\ 3.728e-11,\ 2.994e-11,\ 2.305e-11,\ 1.715e-11,
02695
           1.256e-11, 9.208e-12, 6.745e-12, 5.014e-12, 3.785e-12, 2.9e-12,
           2.239e-12, 1.757e-12, 1.414e-12, 1.142e-12, 9.482e-13, 8.01e-13,
02696
            6.961e-13, 6.253e-13, 5.735e-13, 5.433e-13, 5.352e-13, 5.493e-13,
           5.706e-13, 6.068e-13, 6.531e-13, 7.109e-13, 7.767e-13,
02698
                                                                            8.59e-13,
02699
           9.792e-13, 1.142e-12, 1.371e-12, 1.65e-12, 1.957e-12, 2.302e-12,
02700
           2.705e-12, 3.145e-12, 3.608e-12, 4.071e-12, 4.602e-12, 5.133e-12,
02701
           5.572e-12, 5.987e-12, 6.248e-12, 6.533e-12, 6.757e-12, 6.935e-12,
            7.224e-12, 7.422e-12, 7.538e-12, 7.547e-12, 7.495e-12, 7.543e-12,
02702
            7.725e-12, 8.139e-12, 8.627e-12, 9.146e-12, 9.443e-12, 9.318e-12,
02703
           8.649e-12, 7.512e-12, 6.261e-12, 4.915e-12, 3.647e-12, 2.597e-12,
02704
           1.785e-12, 1.242e-12, 8.66e-13, 6.207e-13, 4.61e-13, 3.444e-13, 2.634e-13, 2.1e-13, 1.725e-13, 1.455e-13, 1.237e-13, 1.085e-13,
02705
02706
           9.513e-14, 7.978e-14, 6.603e-14, 5.288e-14, 4.084e-14, 2.952e-14,
02707
02708
           2.157e-14, 1.593e-14, 1.199e-14, 9.267e-15, 7.365e-15, 6.004e-15,
           4.995e-15, 4.218e-15, 3.601e-15, 3.101e-15, 2.692e-15, 2.36e-15,
           2.094e-15, 1.891e-15, 1.755e-15, 1.699e-15, 1.755e-15, 1.987e-15,
02710
02711
           2.506e-15, 3.506e-15, 5.289e-15, 8.311e-15, 1.325e-14, 2.129e-14,
02712
           3.237e-14, 4.595e-14, 6.441e-14, 8.433e-14, 1.074e-13, 1.383e-13,
           1.762e-13, 2.281e-13, 2.831e-13, 3.523e-13, 4.38e-13, 5.304e-13, 6.29e-13, 7.142e-13, 8.032e-13, 8.934e-13, 9.888e-13, 1.109e-12, 1.261e-12, 1.462e-12, 1.74e-12, 2.099e-12, 2.535e-12, 3.008e-12,
02713
02714
02715
           3.462e-12, 3.856e-12, 4.098e-12, 4.239e-12, 4.234e-12, 4.132e-12,
           3.986e-12, 3.866e-12, 3.829e-12, 3.742e-12, 3.705e-12, 3.694e-12,
02717
02718
           3.765e-12, 3.849e-12, 3.929e-12, 4.056e-12, 4.092e-12, 4.047e-12,
02719
           3.792e-12, 3.407e-12, 2.953e-12, 2.429e-12, 1.931e-12, 1.46e-12,
02720
           1.099e-12, 8.199e-13, 6.077e-13, 4.449e-13, 3.359e-13, 2.524e-13,
            1.881e-13, 1.391e-13, 1.02e-13, 7.544e-14, 5.555e-14, 4.22e-14,
02721
            3.321e-14, 2.686e-14, 2.212e-14, 1.78e-14, 1.369e-14, 1.094e-14,
02722
           9.13e-15, 8.101e-15, 7.828e-15, 8.393e-15, 1.012e-14, 1.259e-14,
02723
02724
           1.538e-14, 1.961e-14, 2.619e-14, 3.679e-14, 5.049e-14, 6.917e-14,
           8.88e-14, 1.115e-13, 1.373e-13, 1.619e-13, 1.878e-13, 2.111e-13, 2.33e-13, 2.503e-13, 2.613e-13, 2.743e-13, 2.826e-13, 2.976e-13,
02725
02726
           3.162e-13, 3.36e-13, 3.491e-13, 3.541e-13, 3.595e-13, 3.608e-13,
02727
           3.709e-13, 3.869e-13, 4.12e-13, 4.366e-13, 4.504e-13, 4.379e-13,
            3.955e-13, 3.385e-13, 2.741e-13, 2.089e-13, 1.427e-13, 9.294e-14,
02729
02730
           5.775e-14, 3.565e-14, 2.21e-14, 1.398e-14, 9.194e-15, 6.363e-15,
02731
           4.644e-15, 3.55e-15, 2.808e-15, 2.274e-15, 1.871e-15, 1.557e-15,
           1.308e-15, 1.108e-15, 9.488e-16, 8.222e-16, 7.238e-16, 6.506e-16, 6.008e-16, 5.742e-16, 5.724e-16, 5.991e-16, 6.625e-16, 7.775e-16,
02732
02733
```

```
9.734e-16, 1.306e-15, 1.88e-15, 2.879e-15, 4.616e-15, 7.579e-15,
            1.248e-14, 2.03e-14, 3.244e-14, 5.171e-14, 7.394e-14, 9.676e-14, 1.199e-13, 1.467e-13, 1.737e-13, 2.02e-13, 2.425e-13, 3.016e-13, 3.7e-13, 4.617e-13, 5.949e-13, 7.473e-13, 9.378e-13, 1.191e-12, 1.481e-12, 1.813e-12, 2.232e-12, 2.722e-12, 3.254e-12, 3.845e-12, 4.458e-12, 5.048e-12, 5.511e-12, 5.898e-12, 6.204e-12, 6.293e-12,
02735
02736
02737
02738
02739
            6.386e-12, 6.467e-12, 6.507e-12, 6.466e-12, 6.443e-12, 6.598e-12,
02740
02741
             6.873e-12, 7.3e-12, 7.816e-12, 8.368e-12, 8.643e-12, 8.466e-12,
02742
            7.871e-12, 6.853e-12, 5.714e-12, 4.482e-12, 3.392e-12, 2.613e-12,
02743
            2.008e-12, 1.562e-12, 1.228e-12, 9.888e-13, 7.646e-13, 5.769e-13,
02744
            4.368e-13, 3.324e-13, 2.508e-13, 1.916e-13
02745
02746
02747
          static double xfcrev[15] =
02748
           { 1.003, 1.009, 1.015, 1.023, 1.029, 1.033, 1.037,
02749
            1.039, 1.04, 1.046, 1.036, 1.027, 1.01, 1.002, 1.
02750
02751
02752
         double a1, a2, a3, dw, ew, dx, xw, xx, vf2, vf6, cw260, cw296,
02753
           sfac, fscal, cwfrn, ctmpth, ctwfrn, ctwslf;
02754
         int iw, ix;
02755
02756
02757
          /* Get H2O continuum absorption... */
02758
          xw = nu / 10 + 1;
          if (xw >= 1 && xw < 2001) {
02759
02760
            iw = (int) xw;
            dw = xw - iw;

ew = 1 - dw;
02761
02762
            cw296 = ew * h2o296[iw - 1] + dw * h2o296[iw];
cw260 = ew * h2o260[iw - 1] + dw * h2o260[iw];
cwfrn = ew * h2ofrn[iw - 1] + dw * h2ofrn[iw];
02763
02764
02765
02766
            if (nu <= 820 || nu >= 960) {
02767
              sfac = 1;
02768
            } else {
              xx = (nu - 820) / 10;
02769
02770
               ix = (int) xx;
              dx = xx - ix;
02771
02772
              sfac = (1 - dx) * xfcrev[ix] + dx * xfcrev[ix + 1];
02773
02774
            ctwslf = sfac * cw296 * pow(cw260 / cw296, (296 - t) / (296 - 260));
            vf2 = POW2 (nu - 370);
02775
            vf6 = POW3(vf2);
02776
02777
            fscal = 36100 / (vf2 + vf6 * 1e-8 + 36100) * -.25 + 1;
02778
            ctwfrn = cwfrn * fscal;
02779
            a1 = nu * u * tanh(.7193876 / t * nu);
            a2 = 296 / t;
a3 = p / P0 * (q * ctwslf + (1 - q) * ctwfrn) * le-20;
02780
02781
02782
            ctmpth = a1 * a2 * a3;
02783
          } else
02784
           ctmpth = 0;
02785
          return ctmpth;
02786 }
02787
02789
02790 double ctmn2(
02791
         double nu.
          double p,
02792
02793
          double t)
02794
          static double ba[98] = { 0., 4.45e-8, 5.22e-8, 6.46e-8, 7.75e-8, 9.03e-8, 1.06e-7, 1.21e-7, 1.37e-7, 1.57e-7, 1.75e-7, 2.01e-7, 2.3e-7,
02795
02797
             2.59e-7, 2.95e-7, 3.26e-7, 3.66e-7, 4.05e-7, 4.47e-7, 4.92e-7,
02798
            5.34e-7, 5.84e-7, 6.24e-7, 6.67e-7, 7.14e-7, 7.26e-7, 7.54e-7,
02799
            7.84e-7, 8.09e-7, 8.42e-7, 8.62e-7, 8.87e-7, 9.11e-7, 9.36e-7,
02800
            9.76e-7, 1.03e-6, 1.11e-6, 1.23e-6, 1.39e-6, 1.61e-6, 1.76e-6,
            1.94e-6, 1.97e-6, 1.87e-6, 1.75e-6, 1.56e-6, 1.42e-6, 1.35e-6,
02801
            1.32e-6, 1.29e-6, 1.29e-6, 1.3e-6, 1.3e-6, 1.32e-6, 1.33e-6,
02802
            1.34e-6, 1.35e-6, 1.33e-6, 1.31e-6, 1.29e-6, 1.24e-6, 1.2e-6,
            1.16e-6, 1.1e-6, 1.04e-6, 9.96e-7, 9.38e-7, 8.63e-7, 7.98e-7, 7.26e-7, 6.55e-7, 5.94e-7, 5.35e-7, 4.74e-7, 4.24e-7, 3.77e-7,
02804
02805
            3.33e-7, 2.96e-7, 2.63e-7, 2.34e-7, 2.08e-7, 1.85e-7, 1.67e-7, 1.47e-7, 1.32e-7, 1.2e-7, 1.09e-7, 9.85e-8, 9.08e-8, 8.18e-8,
02806
02807
            7.56e-8, 6.85e-8, 6.14e-8, 5.83e-8, 5.77e-8, 5e-8, 4.32e-8, 0.
02808
02809
02810
02811
          static double betaa[98] = { 802., 802., 761., 722., 679., 646., 609., 562.,
            511., 472., 436., 406., 377., 355., 338., 319., 299., 278., 255., 233., 208., 184., 149., 107., 66., 25., -13., -49., -82., -104., -119., -130., -139., -144., -146., -146., -147., -148., -150., -153., -160., -169., -181., -189., -195., -200., -205., -209.,
02812
02813
02814
02815
            -211, -210, -210, -209, -205, -199, -190, -180, -180, -181, -157, -143, -126, -108, -89, -63, -32, 1, 35, 65, 95
02816
02817
02818
            121., 141., 152., 161., 164., 164., 161., 155., 148., 143., 137.,
            133., 131., 133., 139., 150., 165., 187., 213., 248., 284., 321., 372., 449., 514., 569., 609., 642., 673., 673.
02819
02820
```

```
02821
02822
          static double nua[98] = { 2120., 2125., 2130., 2135., 2140., 2145., 2150., 2155., 2160., 2165., 2170., 2175., 2180., 2185., 2190., 2195.,
02823
02824
02825
              2200., 2205., 2210., 2215., 2220., 2225., 2230., 2235., 2240.,
             2245., 2250., 2255., 2260., 2265., 2270., 2275., 2280., 2285., 2290., 2295., 2300., 2305., 2310., 2315., 2320., 2325., 2330.,
02826
              2335., 2340., 2345., 2350., 2355., 2360., 2365., 2370., 2375.,
02828
02829
              2380., 2385., 2390., 2395., 2400., 2405., 2410., 2415., 2420.,
02830
             2425., 2430., 2435., 2440., 2445., 2450., 2455., 2460., 2465.,
             2470., 2475., 2480., 2485., 2490., 2495., 2500., 2505., 2510.,
02831
             2515., 2520., 2525., 2530., 2535., 2540., 2545., 2550., 2555., 2560., 2565., 2570., 2575., 2580., 2585., 2590., 2595., 2600., 2605.
02832
02833
02834
02835
02836
          double b, beta, q_n2 = 0.79, t0 = 273, tr = 296;
02837
02838
          int idx;
02839
02840
           /* Check wavenumber range...
02841
          if (nu < nua[0] || nu > nua[97])
02842
             return 0;
02843
02844
           /* Interpolate B and beta... */
02845
          idx = locate_reg(nua, 98, nu);
b = LIN(nua[idx], ba[idx], nua[idx + 1], ba[idx + 1], nu);
02847
          beta = LIN(nua[idx], betaa[idx], nua[idx + 1], betaa[idx + 1], nu);
02848
02849
           /* Compute absorption coefficient... */
          return 0.1 * POW2(p / P0 * t0 / t) * exp(beta * (1 / tr - 1 / t))
 * q_n2 * b * (q_n2 + (1 - q_n2) * (1.294 - 0.4545 * t / tr));
02850
02851
02852 }
02853
02855
02856 double ctmo2(
02857
          double nu,
          double p,
02859
          double t) {
02860
02861
           static double ba[90] = { 0., .061, .074, .084, .096, .12, .162, .208, .246,
             .285, .314, .38, .444, .5, .571, .673, .768, .853, .966, 1.097, 1.214, 1.333, 1.466, 1.591, 1.693, 1.796, 1.922, 2.037, 2.154, 2.264, 2.375, 2.508, 2.671, 2.847, 3.066, 3.417, 3.828, 4.204,
02862
02863
02864
             4.453, 4.599, 4.528, 4.284, 3.955, 3.678, 3.477, 3.346, 3.29, 3.251, 3.231, 3.226, 3.212, 3.192, 3.108, 3.033, 2.911, 2.798
02866
             2.646, 2.508, 2.322, 2.13, 1.928, 1.757, 1.588, 1.417, 1.253, 1.109, .99, .888, .791, .678, .587, .524, .464, .403, .357, .32, .29, .267, .242, .215, .182, .16, .146, .128, .103, .087, .081, .071, .064, 0.
02867
02868
02869
02870
02871
02872
02873
           static double betaa[90] = { 467., 467., 400., 315., 379., 368., 475., 521.,
             531., 512., 442., 444., 430., 381., 335., 324., 296., 248., 215., 193., 158., 127., 101., 71., 31., -6., -26., -47., -63., -79., -88., -88., -87., -90., -98., -99., -109., -134., -160., -167., -164., -158., -153., -151., -156., -166., -168., -173., -170., -161., -145., -126., -108., -84., -59., -29., 4., 41., 73., 97.,
02874
02875
02876
02878
             123., 159., 198., 220., 242., 256., 281., 311., 334., 319., 313., 321., 323., 310., 315., 320., 335., 361., 378., 373., 338., 319., 346., 322., 291., 290., 350., 371., 504., 504.
02879
02880
02881
02882
02883
02884
           static double nua[90] = { 1360., 1365., 1370., 1375., 1380., 1385., 1390.,
02885
             1395., 1400., 1405., 1410., 1415., 1420., 1425., 1430., 1435.,
02886
             1440., 1445., 1450., 1455., 1460., 1465., 1470., 1475., 1480.,
02887
              1485., 1490., 1495., 1500., 1505., 1510., 1515., 1520., 1525.,
             1530., 1535., 1540., 1545., 1550., 1555., 1560., 1565., 1570., 1575., 1580., 1585., 1590., 1595., 1600., 1605., 1610., 1615.,
02888
02889
              1620., 1625., 1630., 1635., 1640., 1645., 1650., 1655., 1660.,
02891
              1665., 1670., 1675., 1680., 1685., 1690., 1695., 1700., 1705.,
02892
              1710., 1715., 1720., 1725., 1730., 1735., 1740., 1745., 1750.,
02893
             1755., 1760., 1765., 1770., 1775., 1780., 1785., 1790., 1795.,
02894
             1800., 1805.
02895
02896
          double b, beta, q_02 = 0.21, t0 = 273, tr = 296;
02897
02898
02899
          int idx:
02900
02901
           /* Check wavenumber range...
02902
           if (nu < nua[0] || nu > nua[89])
02903
             return 0;
02904
02905
           /\star Interpolate B and beta... \star/
02906
          idx = locate_reg(nua, 90, nu);
          b = LIN(nua[idx], ba[idx], nua[idx + 1], ba[idx + 1], nu);
02907
```

```
beta = LIN(nua[idx], betaa[idx], nua[idx + 1], betaa[idx + 1], nu);
02909
02910
        /* Compute absorption coefficient... */
02911
       return 0.1 * POW2(p / P0 * t0 / t) * exp(beta * (1 / tr - 1 / t)) * q_o2 *
02912
02913 }
02914
02916
02917 void copy_atm(
02918
       ctl_t * ctl,
atm_t * atm_dest,
atm_t * atm_src,
02919
02920
02921
       int init) {
02922
02923
       int ig, ip, iw;
02924
02925
       size t s;
02926
02927
       /* Data size... */
02928
       s = (size_t) atm_src->np * sizeof(double);
02929
       /* Copy data... */
atm_dest->np = atm_src->np;
02930
02931
02932
       memcpy(atm_dest->time, atm_src->time, s);
       memcpy(atm_dest->z, atm_src->z, s);
02933
02934
       memcpy(atm_dest->lon, atm_src->lon, s);
02935
       memcpy(atm_dest->lat, atm_src->lat, s);
02936
       memcpy(atm_dest->p, atm_src->p, s);
02937
       memcpy(atm_dest->t, atm_src->t, s);
       for (ig = 0; ig < ctl->ng; ig++)
02938
       memcpy(atm_dest->q[ig], atm_src->q[ig], s);
for (iw = 0; iw < ctl->nw; iw++)
02939
02940
02941
         memcpy(atm_dest->k[iw], atm_src->k[iw], s);
02942
02943
       /* Initialize... */
02944
       if (init)
         for (ip = 0; ip < atm_dest->np; ip++) {
02945
02946
           atm_dest->p[ip] = 0;
02947
            atm_dest->t[ip] = 0;
02948
            for (ig = 0; ig < ctl->ng; ig++)
             atm_dest->q[ig][ip] = 0;
02949
            for (iw = 0: iw < ctl->nw: iw++)
02950
02951
             atm_dest->k[iw][ip] = 0;
02952
02953 }
02954
02956
02957 void copy_obs(
       ctl_t * ctl,
obs_t * obs_dest,
02958
02959
02960
       obs_t * obs_src,
02961
       int init) {
02962
02963
       int id, ir;
02964
02965
       size_t s;
02966
02967
       /* Data size... */
       s = (size_t) obs_src->nr * sizeof(double);
02968
02969
02970
       /* Copy data... */
02971
       obs_dest->nr = obs_src->nr;
02972
       memcpy(obs_dest->time, obs_src->time, s);
02973
       memcpy(obs_dest->obsz, obs_src->obsz, s);
02974
       memcpy(obs_dest->obslon, obs_src->obslon, s);
02975
       memcpy(obs_dest->obslat, obs_src->obslat, s);
02976
       memcpy(obs_dest->vpz, obs_src->vpz, s);
       memcpy(obs_dest->vplon, obs_src->vplon, s);
02978
       memcpy(obs_dest->vplat, obs_src->vplat, s);
02979
       memcpy(obs_dest->tpz, obs_src->tpz, s);
       memcpy(obs_dest->tplon, obs_src->tplon, s);
02980
       memcpy(obs_dest->tplat, obs_src->tplat, s);
for (id = 0; id < ctl->nd; id++)
02981
02982
02983
         memcpy(obs_dest->rad[id], obs_src->rad[id], s);
02984
       for (id = 0; id < ctl->nd; id++)
02985
         memcpy(obs_dest->tau[id], obs_src->tau[id], s);
02986
       /* Initialize... */
02987
02988
       if (init)
         for (id = 0; id < ctl->nd; id++)
02990
            for (ir = 0; ir < obs_dest->nr; ir++)
02991
              if (gsl_finite(obs_dest->rad[id][ir])) {
02992
               obs_dest->rad[id][ir] = 0;
02993
               obs_dest->tau[id][ir] = 0;
02994
              }
```

```
02996
02998
02999 int find emitter(
03000
       ctl t * ctl.
       const char *emitter) {
03002
03003
03004
03005
       for (ig = 0; ig < ctl->ng; ig++)
03006
        if (strcasecmp(ctl->emitter[ig], emitter) == 0)
03007
           return iq;
03008
03009
       return -1;
03010 }
03011
03013
03014 void formod(
      ctl_t * ctl,
atm_t * atm,
03015
03016
       obs_t * obs) {
03017
03018
03019
       int id, ir, *mask;
03020
       /* Allocate... */
03021
03022
       ALLOC(mask, int,
03023
             ND * NR);
03024
03025
       /* Save observation mask... */
03026
       for (id = 0; id < ctl->nd; id++)
03027
        for (ir = 0; ir < obs->nr; ir++)
03028
           mask[id * NR + ir] = !gsl_finite(obs->rad[id][ir]);
03029
       /* Hydrostatic equilibrium... */
03030
03031
       hydrostatic(ctl, atm);
03032
03033
        /* Calculate pencil beams... */
03034
       for (ir = 0; ir < obs->nr; ir++)
03035
         formod_pencil(ctl, atm, obs, ir);
03036
       /* Apply field-of-view convolution... */
03037
03038
       formod_fov(ctl, obs);
03039
03040
        /* Convert radiance to brightness temperature... */
03041
       if (ctl->write_bbt)
         for (id = 0; id < ctl->nd; id+)
  for (ir = 0; ir < obs->nr; ir++)
   obs->rad[id][ir] = brightness(obs->rad[id][ir], ctl->nu[id]);
03042
03043
03044
03045
03046
        /* Apply observation mask...
       for (id = 0; id < ctl->nd; id++)
  for (ir = 0; ir < obs->nr; ir++)
    if (mask[id * NR + ir])
03047
03048
03049
03050
             obs->rad[id][ir] = GSL_NAN;
03051
03052
03053
       free(mask);
03054 }
03055
03057
03058 void formod_continua(
       ctl_t * ctl,
los_t * los,
03059
03060
03061
       int ip,
03062
       double *beta) {
03063
03064
       static int ig_co2 = -999, ig_h2o = -999;
03065
03066
       int id;
03067
       /* Extinction... */
for (id = 0; id < ctl->nd; id++)
  beta[id] = los->k[ctl->window[id]][ip];
03068
03069
03070
03071
03072
        /* CO2 continuum...
03073
       if (ctl->ctm_co2) {
03074
         if (ig_co2 == -999)
03075
           ig_co2 = find_emitter(ct1, "CO2");
03076
          if (ig_co2 >= 0)
           for (id = 0; id < ctl->nd; id++)
beta[id] += ctmco2(ctl->nu[id], los->p[ip], los->t[ip],
03077
03078
03079
                                los->u[ig_co2][ip]) / los->ds[ip];
03080
03081
```

```
/* H2O continuum... */
03083
       if (ctl->ctm_h2o) {
03084
         if (ig_h2o == -999)
           ig_h2o = find_emitter(ctl, "H2O");
03085
          if (ig_h2o >= 0)
  for (id = 0; id < ctl->nd; id++)
03086
03087
             beta[id] += ctmh2o(ctl->nu[id], los->p[ip], los->t[ip],
03088
03089
                                  los->q[ig_h2o][ip],
03090
                                  los \rightarrow u[ig_h2o][ip]) / los \rightarrow ds[ip];
03091
03092
03093
        /* N2 continuum... */
03094
        if (ctl->ctm_n2)
03095
         for (id = 0; id < ctl->nd; id++)
03096
            beta[id] += ctmn2(ctl->nu[id], los->p[ip], los->t[ip]);
03097
03098
        /* 02 continuum... */
03099
        if (ctl->ctm o2)
         for (id = 0; id < ctl->nd; id++)
03100
03101
            beta[id] += ctmo2(ctl->nu[id], los->p[ip], los->t[ip]);
03102 }
03103
03105
03106 void formod_fov(
03107
       ctl_t * ctl,
03108
       obs_t * obs) {
03109
0.3110
       static double dz[NSHAPE], w[NSHAPE];
03111
03112
       static int init = 0, n;
03113
03114
       obs_t *obs2;
03115
0.3116
       double rad[ND][NR], tau[ND][NR], wsum, z[NR], zfov;
03117
03118
       int i, id, idx, ir, ir2, nz;
03119
03120
       /* Do not take into account FOV... */
03121
       if (ctl->fov[0] == '-')
03122
          return;
03123
       /* Initialize FOV data... */
03124
03125
       if (!init) {
        init = 1;
03126
03127
          read_shape(ctl->fov, dz, w, &n);
03128
03129
        /* Allocate... */
03130
03131
       ALLOC(obs2, obs t, 1);
03132
03133
        /* Copy observation data... */
03134
       copy_obs(ct1, obs2, obs, 0);
0.3135
       /* Loop over ray paths... */
for (ir = 0; ir < obs->nr; ir++) {
03136
03137
03138
03139
          /* Get radiance and transmittance profiles... */
03140
          nz = 0;
03141
          for (ir2 = GSL_MAX(ir - NFOV, 0); ir2 < GSL_MIN(ir + 1 + NFOV, obs->nr);
0.3142
               ir2++)
            if (obs->time[ir2] == obs->time[ir]) {
03143
03144
              z[nz] = obs2->vpz[ir2];
03145
              for (id = 0; id < ctl->nd; id++)
03146
                rad[id][nz] = obs2->rad[id][ir2];
03147
               tau[id][nz] = obs2->tau[id][ir2];
03148
              }
03149
             nz++;
03150
03151
03152
            ERRMSG("Cannot apply FOV convolution!");
03153
03154
          /\star Convolute profiles with FOV... \star/
          wsum = 0;
for (id = 0; id < ctl->nd; id++) {
03155
03156
03157
           obs->rad[id][ir] = 0;
03158
            obs->tau[id][ir] = 0;
03159
03160
          for (i = 0; i < n; i++)
            zfov = obs->vpz[ir] + dz[i];
03161
            idx = locate_irr(z, nz, zfov);
for (id = 0; id < ctl->nd; id++) {
03162
03163
03164
             obs->rad[id][ir] += w[i]
03165
                * LIN(z[idx], rad[id][idx], z[idx + 1], rad[id][idx + 1], zfov);
03166
              obs->tau[id][ir] += w[i]
                * LIN(z[idx], tau[id][idx], z[idx + 1], tau[id][idx + 1], zfov);
0.3167
03168
```

```
03169
           wsum += w[i];
03170
03171
          for (id = 0; id < ctl->nd; id++) {
           obs->rad[id][ir] /= wsum;
obs->tau[id][ir] /= wsum;
03172
03173
03174
03175
03176
03177
        /* Free... */
03178
       free(obs2);
03179 }
03180
03182
03183 void formod_pencil(
       ctl_t * ctl,
atm_t * atm,
03184
03185
        obs t * obs,
03186
03187
       int ir) {
03188
03189
       static tbl_t *tbl;
03190
0.3191
       static int init = 0;
03192
03193
       los_t *los;
03194
03195
       double beta_ctm[ND], eps, src_planck[ND], tau_path[NG][ND], tau_gas[ND];
03196
03197
       int id, ip;
03198
03199
        /* Initialize look-up tables... */
03200
        if (!init) {
03201
         init = 1;
03202
          ALLOC(tbl, tbl_t, 1);
03203
          init_tbl(ctl, tbl);
03204
03205
03206
        /* Allocate... */
03207
       ALLOC(los, los_t, 1);
03208
        /* Initialize... */
03209
       for (id = 0; id < ctl->nd; id++) {
03210
        obs->rad[id][ir] = 0;
03211
03212
         obs->tau[id][ir] = 1;
03213
03214
03215
       /* Raytracing... */
03216
       raytrace(ctl, atm, obs, los, ir);
03217
       /* Loop over LOS points... */
03218
03219
        for (ip = 0; ip < los->np; ip++) {
03220
03221
          /* Get trace gas transmittance... */
03222
         intpol_tbl(ctl, tbl, los, ip, tau_path, tau_gas);
03223
03224
          /* Get continuum absorption... */
03225
          formod_continua(ctl, los, ip, beta_ctm);
03226
03227
          /* Compute Planck function... */
03228
          formod_srcfunc(ctl, tbl, los->t[ip], src_planck);
03229
         /* Loop over channels... */
for (id = 0; id < ctl->nd; id++)
03230
03231
03232
           if (tau_gas[id] > 0) {
03233
03234
              /* Get segment emissivity... */
03235
              eps = 1 - tau_gas[id] * exp(-beta_ctm[id] * los->ds[ip]);
03236
03237
              /* Compute radiance... */
03238
              obs->rad[id][ir] += src_planck[id] * eps * obs->tau[id][ir];
03239
03240
              /\star Compute path transmittance... \star/
03241
              obs \rightarrow tau[id][ir] *= (1 - eps);
03242
03243
       }
03244
03245
        /* Add surface...
03246
       if (los->tsurf > 0) {
         formod_srcfunc(ct1, tb1, los->tsurf, src_planck);
03247
03248
         for (id = 0; id < ctl->nd; id++)
03249
           obs->rad[id][ir] += src_planck[id] * obs->tau[id][ir];
03250
03251
03252
        /* Free... */
03253
       free(los);
03254 }
03255
```

```
03257
03258 void formod_srcfunc(
       ctl_t * ctl,
tbl_t * tbl,
03259
03260
       double t,
03261
03262
       double *src) {
03263
03264
       int id, it;
03265
03266
        /* Determine index in temperature array... */
03267
       it = locate_reg(tbl->st, TBLNS, t);
03268
03269
        /* Interpolate Planck function value... */
03270
        for (id = 0; id < ctl->nd; id++)
         03271
03272
03273 }
03274
03276
03277 void geo2cart(
03278
       double z,
03279
       double lon,
03280
        double lat,
03281
       double *x) {
03282
03283
       double radius;
03284
03285
       radius = z + RE;
       x[0] = radius * cos(lat / 180 * M_PI) * cos(lon / 180 * M_PI);
x[1] = radius * cos(lat / 180 * M_PI) * sin(lon / 180 * M_PI);
x[2] = radius * sin(lat / 180 * M_PI);
03286
03287
03288
03289 }
03290
03292
03293 void hydrostatic(
03294
       ctl_t * ctl,
03295
       atm_t * atm)
03296
03297
       static int ig_h2o = -999;
03298
03299
       double dzmin = 1e99, e = 0, mean, mmair = 28.96456e-3, mmh2o = 18.0153e-3;
03300
03301
        int i, ip, ipref = 0, ipts = 20;
03302
       /* Check reference height... */
if (ctl->hydz < 0)</pre>
03303
03304
03305
         return:
03306
03307
        /* Determine emitter index of H2O... */
03308
        if (ig_h2o == -999)
03309
         ig_h2o = find_emitter(ctl, "H2O");
03310
03311
        /* Find air parcel next to reference height... */
03312
        for (ip = 0; ip < atm->np; ip++)
03313
         if (fabs(atm->z[ip] - ctl->hydz) < dzmin) {</pre>
           dzmin = fabs(atm->z[ip] - ctl->hydz);
ipref = ip;
03314
03315
03316
03317
03318
        /\star Upper part of profile... \star/
03319
        for (ip = ipref + 1; ip < atm->np; ip++) {
03320
          mean = 0;
03321
          for (i = 0; i < ipts; i++) {</pre>
            if (ig_h2o >= 0)
03322
             e = LIN(0.0, atm->q[ig_h2o][ip - 1],
ipts - 1.0, atm->q[ig_h2o][ip], (double) i);
03323
03324
            mean += (e * mmh2o + (1 - e) * mmair)
03325
             * G0 / RI
03326
              / LIN(0.0, atm->t[ip - 1], ipts - 1.0, atm->t[ip], (double) i) / ipts;
03327
03328
         }
03329
03330
          /* Compute p(z,T) \dots */
03331
         atm->p[ip]
03332
            \exp(\log(\text{atm->p[ip - 1]}) - \text{mean} * 1000 * (\text{atm->z[ip] - atm->z[ip - 1]}));
03333
03334
        /* Lower part of profile... */
for (ip = ipref - 1; ip >= 0; ip--) {
03335
03336
03337
         mean = 0;
03338
          for (i = 0; i < ipts; i++) {</pre>
03339
            if (ig_h2o >= 0)
03340
             e = LIN(0.0, atm->q[ig_h2o][ip + 1],
           ipts - 1.0, atm \rightarrow q[ig_h2o][ip], (double) i); mean += (e * mmh2o + (1 - e) * mmair)
03341
03342
```

```
03343
             * G0 / RI
03344
             / LIN(0.0, atm->t[ip + 1], ipts - 1.0, atm->t[ip], (double) i) / ipts;
03345
03346
03347
         /* Compute p(z,T) ... */
03348
         atm->p[ip] :
03349
           \exp(\log(atm-p[ip + 1]) - mean * 1000 * (atm-z[ip] - atm-z[ip + 1]));
03350
03351 }
03352
03354
03355 void idx2name(
03356
      ctl_t * ctl,
03357
       int idx,
03358
       char *quantity) {
03359
03360
       int iq, iw;
03361
03362
       if (idx == IDXP)
03363
         sprintf(quantity, "PRESSURE");
03364
       if (idx == TDXT)
03365
         sprintf(quantity, "TEMPERATURE");
03366
03367
03368
       for (ig = 0; ig < ctl->ng; ig++)
         if (idx == IDXQ(ig))
03369
           sprintf(quantity, "%s", ctl->emitter[ig]);
03370
03371
03372
       for (iw = 0; iw < ctl->nw; iw++)
03373
        if (idx == IDXK(iw))
03374
           sprintf(quantity, "EXTINCT_WINDOW%d", iw);
03375 }
03376
03378
03379 void init_tbl(
03380 ctl_t * ctl,
03381
       tbl_t * tbl) {
03382
03383
       FILE *in;
03384
       char filename[2 * LEN]. line[LEN]:
03385
03386
03387
       double eps, eps_old, press, press_old, temp, temp_old, u, u_old,
03388
         f[NSHAPE], fsum, nu[NSHAPE];
03389
03390
       int i, id, ig, ip, it, n;
03391
03392
       /* Loop over trace gases and channels... */
       for (ig = 0; ig < ctl->ng; ig++)
03393
03394 #pragma omp parallel for default (none) shared(ctl,tbl,ig) private(in,filename,line,eps,eps_old,press,
     press_old,temp,temp_old,u,u_old,id,ip,it)
03395
         for (id = 0; id < ctl->nd; id++) {
03396
03397
            /* Initialize... */
03398
           tbl->np[ig][id] = -1;
03399
           eps_old = -999;
           press_old = -999;
temp_old = -999;
03400
03401
           u_old = -999;
03402
03403
03404
            /* Try to open file... */
03405
           sprintf(filename, "%s_%.4f_%s.tab",
03406
                   ctl->tblbase, ctl->nu[id], ctl->emitter[ig]);
           if (!(in = fopen(filename, "r"))) {
  printf("Missing emissivity table: %s\n", filename);
03407
03408
03409
             continue:
03410
03411
           printf("Read emissivity table: %s\n", filename);
03412
03413
            /* Read data... */
03414
           while (fgets(line, LEN, in)) {
03415
             /* Parse line... */
if (sscanf(line, "%lg %lg %lg %lg", &press, &temp, &u, &eps) != 4)
03416
03417
03418
03419
03420
             /* Determine pressure index... */
             if (press != press_old) {
  press_old = press;
03421
03422
               if ((++tbl->np[ig][id]) >= TBLNP)
03423
03424
                 ERRMSG("Too many pressure levels!");
03425
               tbl->nt[ig][id][tbl->np[ig][id]] = -1;
03426
             }
03427
03428
             /* Determine temperature index... */
```

```
if (temp != temp_old) {
03430
                temp_old = temp;
03431
                if ((++tbl->nt[ig][id][tbl->np[ig][id]]) >= TBLNT)
                ERRMSG("Too many temperatures!");
tbl->nu[ig][id][tbl->np[ig][id]]
03432
03433
                  [tbl->nt[ig][id][tbl->np[ig][id]]] = -1;
03434
03435
03436
              03437
03438
03439
                eps_old = eps;
03440
03441
                u\_old = u;
03442
                if ((++tbl->nu[ig][id][tbl->np[ig][id]]
03443
                     [tbl->nt[ig][id][tbl->np[ig][id]]]) >= TBLNU) {
03444
                  tbl->nu[ig][id][tbl->np[ig][id]]
03445
                    [tbl->nt[ig][id][tbl->np[ig][id]]]--;
03446
                  continue:
03447
03448
              }
03449
              /* Store data... */
tbl->p[ig][id][tbl->np[ig][id]] = press;
03450
03451
              \label{tbl-} t[ig][id][tbl->np[ig][id]][tbl->nt[ig][id][tbl->np[ig][id]]]
03452
03453
                = temp;
              tbl->u[ig][id][tbl->np[ig][id]][tbl->nt[ig][id][tbl->np[ig][id]]]
03454
03455
                [tbl->nu[ig][id][tbl->np[ig][id]]
03456
                  [tbl->nt[ig][id][tbl->np[ig][id]]] = (float) u;
03457
              \label{locality} $$ tbl->eps[ig][id][tbl->np[ig][id]][tbl->nt[ig][id][tbl->np[ig][id]]] $$ $$ $$
                [tbl->nu[ig][id][tbl->np[ig][id]]
03458
03459
                 [tbl->nt[ig][id][tbl->np[ig][id]]] = (float) eps;
03460
03461
03462
            /* Increment counters... */
03463
            tbl->np[ig][id]++;
            for (ip = 0; ip < tbl->np[ig][id]; ip++) {
03464
              tbl->nt[ig][id][ip]++;
for (it = 0; it < tbl->nt[ig][id][ip]; it++)
03465
03466
03467
                tbl->nu[ig][id][ip][it]++;
03468
03469
            /* Close file... */
03470
03471
            fclose(in);
03472
03473
03474
        /* Write info... */
03475
       printf("Initialize source function table...\n");
03476
03477
        /* Loop over channels... */
03478 #pragma omp parallel for default(none) shared(ctl,tbl,ig) private(filename,it,i,n,f,fsum,nu)
        for (id = 0; id < ctl->nd; id++) {
03480
03481
          /* Read filter function... */
          sprintf(filename, "%s_%.4f.filt", ctl->tblbase, ctl->nu[id]);
03482
03483
          read_shape(filename, nu, f, &n);
03484
03485
          /* Compute source function table... */
03486
          for (it = 0; it < TBLNS; it++) {</pre>
03487
03488
            /* Set temperature...
03489
            tbl->st[it] = LIN(0.0, TMIN, TBLNS - 1.0, TMAX, (double) it);
03490
03491
            /* Integrate Planck function... */
03492
            fsum = 0;
03493
            tbl->sr[id][it] = 0;
03494
            for (i = 0; i < n; i++) {</pre>
03495
              fsum += f[i]:
              tbl \rightarrow sr[id][it] += f[i] * planck(tbl \rightarrow st[it], nu[i]);
03496
03497
03498
            tbl->sr[id][it] /= fsum;
03499
03500
       }
03501 }
03502
03503 /
       *****************************
03504
03505 void intpol_atm(
       ctl_t * ctl,
atm_t * atm,
03506
03507
03508
        double z.
03509
        double *p,
        double *t,
03510
03511
        double *q,
03512
        double *k)
03513
03514
        int ig, ip, iw;
03515
```

```
/* Get array index... */
03517
        ip = locate_irr(atm->z, atm->np, z);
03518
        /* Interpolate... */
03519
03520
        *p = EXP(atm->z[ip], atm->p[ip], atm->z[ip + 1], atm->p[ip + 1], z);
         *t = LIN(atm->z[ip], atm->t[ip], atm->z[ip + 1], atm->t[ip + 1], z);
03521
        for (ig = 0; ig < ctl->ng; ig++)
03523
          q[ig] =
03524
            \label{eq:linear} LIN(atm->z[ip], atm->q[ig][ip], atm->z[ip+1], atm->q[ig][ip+1], z);
03525
         for (iw = 0; iw < ctl->nw; iw++)
          k[iw] =
03526
03527
            LIN(atm->z[ip], atm->k[iw][ip], atm->z[ip + 1], atm->k[iw][ip + 1], z);
03528 }
03529
03531
03532 void intpol_tbl(
03533
        ctl_t * ctl,
tbl_t * tbl,
03534
        los_t * los,
03535
03536
        int ip,
03537
        double tau_path[NG][ND],
03538
        double tau_seg[ND]) {
03539
03540
        double eps, eps00, eps01, eps10, eps11, u;
03541
03542
        int id, ig, ipr, it0, it1;
03543
03544
        /* Initialize... */
        if (ip <= 0)
  for (ig = 0; ig < ctl->ng; ig++)
    for (id = 0; id < ctl->nd; id++)
03545
03546
03547
03548
              tau_path[ig][id] = 1;
03549
        /* Loop over channels... */
for (id = 0; id < ctl->nd; id++) {
03550
03551
03552
03553
           /* Initialize... */
03554
          tau_seg[id] = 1;
03555
03556
           /* Loop over emitters.... */
03557
          for (ig = 0; ig < ctl->ng; ig++) {
03558
03559
             /* Check size of table (pressure)... */
03560
            if (tbl->np[ig][id] < 2)</pre>
               eps = 0;
03561
03562
            /* Check transmittance... */
else if (tau_path[ig][id] < 1e-9)</pre>
03563
03564
03565
              eps = 1;
03566
03567
             /* Interpolate... */
03568
             else {
03569
               /\!\star Determine pressure and temperature indices... \star/
03570
               ipr = locate_irr(tbl->p[ig][id], tbl->np[ig][id], los->p[ip]);
03571
03572
03573
                 locate_irr(tbl->t[ig][id][ipr], tbl->nt[ig][id][ipr], los->
      t[ip]);
03574
              it1 =
03575
                locate_reg(tbl->t[ig][id][ipr + 1], tbl->nt[ig][id][ipr + 1],
03576
                             los->t[ip]);
03578
               /\star Check size of table (temperature and column density)... \star/
03579
               if (tbl->nt[ig][id][ipr] < 2 || tbl->nt[ig][id][ipr + 1] < 2</pre>
03580
                   || tbl->nu[ig][id][ipr][it0] < 2
                   03581
                   || tbl->nu[ig][id][ipr + 1][it1] < 2
|| tbl->nu[ig][id][ipr + 1][it1 + 1] < 2)
03582
03583
03584
                 eps = 0;
03585
03586
               else {
03587
                 /* Get emissivities of extended path... */
u = intpol_tbl_u(tbl, ig, id, ipr, it0, 1 - tau_path[ig][id]);
eps00 = intpol_tbl_eps(tbl, ig, id, ipr, it0, u + los->u[ig][ip]);
03588
03589
03590
03591
03592
                 u = intpol_tbl_u(tbl, ig, id, ipr, it0 + 1, 1 - tau_path[ig][id]);
                 eps01 =
03593
03594
                   intpol tbl eps(tbl, iq, id, ipr, it0 + 1, u + los->u[iq][ip]);
03595
03596
                 u = intpol_tbl_u(tbl, ig, id, ipr + 1, it1, 1 - tau_path[ig][id]);
03597
03598
                   intpol_tbl_eps(tbl, ig, id, ipr + 1, it1, u + los->u[ig][ip]);
03599
03600
03601
                   intpol tbl u(tbl, ig, id, ipr + 1, it1 + 1, 1 - tau path[igl[idl]);
```

```
03602
               eps11 =
                intpol_tbl_eps(tbl, ig, id, ipr + 1, it1 + 1, u + los->
     u[ig][ip]);
03604
03605
               /* Interpolate with respect to temperature... */
              03606
03607
03608
03609
                           tbl->t[ig][id][ipr + 1][it1 + 1], eps11, los->t[ip]);
03610
03611
               /* Interpolate with respect to pressure... */
               03612
03613
03614
               /* Check emssivity range... */
03615
03616
               eps00 = GSL_MAX(GSL_MIN(eps00, 1), 0);
03617
03618
               /* Determine segment emissivity... */
eps = 1 - (1 - eps00) / tau_path[ig][id];
03619
03620
             }
03621
03622
03623
           /\star Get transmittance of extended path... \star/
03624
           tau_path[ig][id] *= (1 - eps);
03625
03626
           /* Get segment transmittance... */
03627
           tau_seg[id] *= (1 - eps);
03628
03629
       }
03630 }
03631
03632 /
       *****************************
03633
03634 double intpol_tbl_eps(
03635
       tbl_t * tbl,
03636
       int iq,
03637
       int id,
03638
       int ip,
03639
       int it,
03640
       double u) {
03641
03642
       int idx;
03643
03644
       /* Lower boundary... */
       if (u < tbl->u[ig][id][ip][it][0])
03645
03646
         return LIN(0, 0, tbl->u[ig][id][ip][it][0], tbl->eps[ig][id][ip][it][0],
03647
                    u);
03648
03649
       /* Upper boundary... */
       else if (u > tbl->u[ig][id][ip][it][tbl->nu[ig][id][ip][it] - 1])
03650
        return LIN(tbl->u[ig][id][ip][it][tbl->nu[ig][id][ip][it] - 1],
03651
03652
                    tbl->eps[ig][id][ip][it][tbl->nu[ig][id][ip][it] - 1],
03653
                    1e30, 1, u);
03654
03655
       /* Interpolation... */
03656
       else {
03657
03658
          /* Get index... */
03659
         idx = locate_tbl(tbl->u[ig][id][ip][it], tbl->nu[ig][id][ip][it], u);
03660
03661
         /* Interpolate... */
03662
           LIN(tbl->u[ig][id][ip][it][idx], tbl->eps[ig][id][ip][it][idx], tbl->u[ig][id][ip][it][idx + 1], tbl->eps[ig][id][ip][it][idx + 1],
03663
03664
03665
03666
03667 }
03668
03670
03671 double intpol_tbl_u(
03672
       tbl_t * tbl,
03673
       int ig,
03674
       int id.
03675
       int ip,
03676
       int it,
03677
       double eps) {
03678
03679
       int idx:
03680
03681
       /* Lower boundary... */
03682
       if (eps < tbl->eps[ig][id][ip][it][0])
03683
        return LIN(0, 0, tbl->eps[ig][id][ip][it][0], tbl->u[ig][id][ip][it][0],
03684
                    eps);
03685
       /* Upper boundary... */
else if (eps > tbl->eps[ig][id][ip][it][tbl->nu[ig][id][ip][it] - 1])
03686
03687
```

```
return LIN(tbl->eps[ig][id][ip][it][tbl->nu[ig][id][ip][it] - 1],
03689
                    tbl->u[ig][id][ip][it][tbl->nu[ig][id][ip][it] - 1],
03690
                    1, 1e30, eps);
03691
03692
       /* Interpolation... */
03693
       else {
03694
03695
03696
         idx = locate\_tbl(tbl->eps[ig][id][ip][it], tbl->nu[ig][id][ip][it], eps);
03697
03698
         /* Interpolate... */
03699
           LIN(tbl->eps[ig][id][ip][it][idx], tbl->u[ig][id][ip][it][idx], tbl->eps[ig][id][ip][it][idx + 1], tbl->u[ig][id][ip][it][idx + 1],
03700
03701
03702
03703
03704 }
03705
03708 void jsec2time(
03709
       double jsec,
03710
       int *year,
03711
       int *mon,
03712
       int *day,
03713
       int *hour,
03714
       int *min,
       int *sec,
03715
03716
       double *remain) {
03717
03718
       struct tm t0, *t1;
03719
03720
       time_t jsec0;
03721
       t0.tm_year = 100;
t0.tm_mon = 0;
03722
03723
03724
       t0.tm_mday = 1;
03725
       t0.tm\_hour = 0;
03726
       t0.tm_min = 0;
03727
       t0.tm\_sec = 0;
03728
       jsec0 = (time_t) jsec + timegm(&t0);
03729
03730
       t1 = gmtime(&jsec0);
03731
03732
       *year = t1->tm_year + 1900;
03733
        *mon = t1->tm_mon + 1;
03734
       *day = t1->tm_mday;
03735
       *hour = t1->tm_hour;
       *min = t1->tm_min;
03736
03737
       *sec = t1->tm_sec;
03738
       *remain = jsec - floor(jsec);
03739 }
03740
03742
03743 void kernel(
      ctl_t * ctl,
atm_t * atm,
03744
03745
       obs_t * obs,
03746
03747
       gsl_matrix * k) {
03748
03749
       atm_t *atm1;
03750
       obs_t *obs1;
03751
03752
       gsl_vector *x0, *x1, *yy0, *yy1;
03753
03754
       int *iqa, j;
03755
03756
       double h:
03757
03758
       size_t i, n, m;
03759
03760
       /* Get sizes... */
03761
       m = k -> size1:
       n = k \rightarrow size2;
03762
03763
03764
        /* Allocate... */
03765
       x0 = gsl\_vector\_alloc(n);
        yy0 = gsl_vector_alloc(m);
03766
03767
       ALLOC(iqa, int,
03768
             N);
03769
03770
        /\star Compute radiance for undisturbed atmospheric data... \star/
03771
       formod(ctl, atm, obs);
03772
03773
       /* Compose vectors... */
03774
       atm2x(ctl, atm, x0, iqa, NULL);
```

```
obs2y(ctl, obs, yy0, NULL, NULL);
03776
03777
        /* Initialize kernel matrix... */
03778
        gsl_matrix_set_zero(k);
03779
03780 /* Loop over state vector elements... */
03781 #pragma omp parallel for default(none) shared(ctl,atm,obs,k,x0,yy0,n,m,iqa) private(i, j, h, x1, yy1, atm1,
03782
        for (j = 0; j < (int) n; j++) {
03783
03784
           /* Allocate... */
          x1 = gsl_vector_alloc(n);
yy1 = gsl_vector_alloc(m);
ALLOC(atm1, atm_t, 1);
03785
03786
03787
03788
           ALLOC(obs1, obs_t, 1);
03789
03790
           /* Set perturbation size... */
03791
           if (iqa[j] == IDXP)
03792
            h = GSL_MAX(fabs(0.01 * gsl_vector_get(x0, (size_t) j)), 1e-7);
03793
           else if (iqa[j] == IDXT)
03794
            h = 1;
03795
           else if (iqa[j] \geq= IDXQ(0) && iqa[j] < IDXQ(ctl-\geqng))
03796
            h = GSL\_MAX(fabs(0.01 * gsl\_vector\_get(x0, (size\_t) j)), 1e-15);
03797
           else if (iqa[j] \geq= IDXK(0) && iqa[j] < IDXK(ctl->nw))
03798
            h = 1e-4;
03799
03800
             ERRMSG("Cannot set perturbation size!");
03801
03802
           /* Disturb state vector element... */
          gsl_vector_memcpy(x1, x0);
gsl_vector_set(x1, (size_t) j, gsl_vector_get(x1, (size_t) j) + h);
copy_atm(ctl, atm1, atm, 0);
copy_obs(ctl, obs1, obs, 0);
03803
03804
03805
03806
03807
           x2atm(ctl, x1, atm1);
03808
03809
           /* Compute radiance for disturbed atmospheric data... */
03810
           formod(ctl, atml, obs1);
03811
03812
           /* Compose measurement vector for disturbed radiance data... */
03813
           obs2y(ctl, obs1, yy1, NULL, NULL);
03814
03815
           /* Compute derivatives... */
           for (i = 0; i < m; i++)
  gsl_matrix_set(k, i, (size_t) j,</pre>
03816
03817
                             (gsl_vector_get(yy1, i) - gsl_vector_get(yy0, i)) / h);
03818
03819
03820
           /* Free... */
03821
           gsl_vector_free(x1);
03822
           gsl_vector_free(yy1);
03823
           free(atm1);
03824
          free (obs1);
03825
03826
03827
        /* Free... */
        gsl_vector_free(x0);
03828
03829
        gsl_vector_free(yy0);
        free(iqa);
03831 }
03832
03834
03835 int locate_irr(
03836
        double *xx,
03837
        int n,
03838
        double x) {
03839
03840
        int i, ilo, ihi;
03841
03842
        ilo = 0;
        ihi = n - 1;
03843
03844
        i = (ihi + ilo) >> 1;
03845
        if (xx[i] < xx[i + 1])
while (ihi > ilo + 1) {
  i = (ihi + ilo) >> 1;
03846
03847
03848
03849
             if (xx[i] > x)
03850
               ihi = i;
03851
             else
03852
               ilo = i;
03853
        } else
          while (ihi > ilo + 1) {
03854
             i = (ihi + ilo) >> 1;
03855
03856
             if (xx[i] \le x)
03857
               ihi = i;
03858
             else
               ilo = i;
03859
03860
           }
```

```
03862
      return ilo;
03863 }
03864
03866
03867 int locate_reg(
03868
      double *xx,
03869
      int n,
03870
      double x) {
03871
03872
      int i:
03873
03874
      /* Calculate index... */
      i = (int) ((x - xx[0]) / (xx[1] - xx[0]));
03875
03876
03877
      /* Check range... */
03878
      if (i < 0)</pre>
03879
       i = 0;
03880
      else if (i >= n - 2)
03881
       i = n - 2;
03882
03883
      return i;
03884 }
03885
03887
03888 int locate_tbl(
03889
      float *xx,
03890
      int n.
03891
      double x) {
03892
03893
      int i, ilo, ihi;
03894
      ilo = 0;
ihi = n - 1;
03895
03896
      i = (ihi + ilo) >> 1;
03897
03898
03899
      while (ihi > ilo + 1)
      i = (ihi + ilo) >> 1;
if (xx[i] > x)
03900
03901
         ihi = i;
03902
        else
03903
03904
         ilo = i;
03905
      }
03906
03907
      return ilo;
03908 }
03909
03911
03912 size_t obs2y(
      ctl_t * ctl,
obs_t * obs,
03913
0.3914
03915
      gsl_vector * y,
      int *ida,
int *ira) {
03916
03917
03918
03919
      int id, ir;
03920
03921
      size t m = 0;
03922
03923
      /* Determine measurement vector... */
03924
      for (ir = 0; ir < obs->nr; ir++)
03925
        for (id = 0; id < ctl->nd; id++)
03926
         if (gsl_finite(obs->rad[id][ir])) {
03927
           if (y != NULL)
  gsl_vector_set(y, m, obs->rad[id][ir]);
03928
           if (ida != NULL)
03929
03930
             ida[m] = id;
03931
           if (ira != NULL)
03932
             ira[m] = ir;
03933
           m++;
03934
03935
03936
      return m;
03937 }
03938
03940
03941 double planck(
03942
      double t,
03943
03944
03945
      return C1 * POW3(nu) / gsl_expm1(C2 * nu / t);
03946 }
03947
```

```
03949
03950 void raytrace(
03951
        ctl_t * ctl,
        atm_t * atm,
03952
03953
        obs t * obs.
        los_t * los,
03954
03955
        int ir) {
03956
03957
        double cosa, d, dmax, dmin = 0, ds, ex0[3], ex1[3], frac, h = 0.02, k[NW],
03958
          lat, lon, n, naux, ng[3], norm, p, q[NG], t, x[3], xh[3],
03959
          xobs[3], xvp[3], z = 1e99, zmax, zmin, zrefrac = 60;
03960
03961
        int i, ig, ip, iw, stop = 0;
03962
03963
        /* Initialize... */
03964
        los->np = 0;
        los->tsurf = -999;
03965
        obs->tpz[ir] = obs->vpz[ir];
03966
        obs->tplon[ir] = obs->vplon[ir];
03967
03968
        obs->tplat[ir] = obs->vplat[ir];
03969
03970
        /* Get altitude range of atmospheric data... */
03971
        gsl_stats_minmax(&zmin, &zmax, atm->z, 1, (size_t) atm->np);
03972
03973
        /* Check observer altitude... */
03974
        if (obs->obsz[ir] < zmin)</pre>
03975
          ERRMSG("Observer below surface!");
03976
03977
        /* Check view point altitude... */
03978
        if (obs->vpz[ir] > zmax)
03979
          return;
03980
03981
        /\star Determine Cartesian coordinates for observer and view point... \star/
03982
        geo2cart(obs->obsz[ir], obs->obslon[ir], obs->obslat[ir], xobs);
03983
        geo2cart(obs->vpz[ir], obs->vplon[ir], obs->vplat[ir], xvp);
03984
03985
        /* Determine initial tangent vector... */
03986
        for (i = 0; i < 3; i++)</pre>
03987
         ex0[i] = xvp[i] - xobs[i];
        norm = NORM(ex0);
for (i = 0; i < 3; i++)
  ex0[i] /= norm;</pre>
03988
03989
03990
03991
03992
        /\star Observer within atmosphere... \star/
03993
        for (i = 0; i < 3; i++)
03994
         x[i] = xobs[i];
03995
03996
        /* Observer above atmosphere (search entry point)... */
        if (obs->obsz[ir] > zmax) {
03997
03998
          dmax = norm;
03999
          while (fabs(dmin - dmax) > 0.001) {
04000
            d = (dmax + dmin) / 2;
            for (i = 0; i < 3; i++)
  x[i] = xobs[i] + d * ex0[i];</pre>
04001
04002
04003
            cart2geo(x, &z, &lon, &lat);
if (z <= zmax && z > zmax - 0.001)
04004
04005
              break;
04006
             if (z < zmax - 0.0005)
04007
              dmax = d;
            else
04008
04009
              dmin = d;
04010
          }
04011
04012
04013
        /* Ray-tracing... */
04014
        while (1) {
04015
04016
          /* Set step length... */
          ds = ctl->rayds;
          if (ctl->raydz > 0) {
04018
04019
            norm = NORM(x);
            for (i = 0; i < 3; i++)
  xh[i] = x[i] / norm;</pre>
04020
04021
            cosa = fabs(DOTP(ex0, xh));
04022
04023
            if (cosa != 0)
04024
              ds = GSL_MIN(ctl->rayds, ctl->raydz / cosa);
04025
04026
04027
          /* Determine geolocation... */
04028
          cart2geo(x, &z, &lon, &lat);
04029
04030
           /\star Check if LOS hits the ground or has left atmosphere... \star/
04031
          if (z < zmin \mid \mid z > zmax) {
04032
            stop = (z < zmin ? 2 : 1);
04033
             frac =
               ((z <
04034
```

```
zmin ? zmin : zmax) - los->z[los->np - 1]) / (z - los->z[los->np -
04036
04037
              geo2cart(los->z[los->np - 1], los->lon[los->np - 1],
                         los->lat[los->np - 1], xh);
04038
              for (i = 0; i < 3; i++)
04039
               x[i] = xh[i] + frac * (x[i] - xh[i]);
04040
              cart2geo(x, &z, &lon, &lat);
04041
04042
              los \rightarrow ds[los \rightarrow np - 1] = ds * frac;
04043
              ds = 0;
04044
04045
04046
            /\star Interpolate atmospheric data... \star/
04047
            intpol_atm(ctl, atm, z, &p, &t, q, k);
04048
04049
            /* Save data... */
            los->lon[los->np] = lon;
los->lat[los->np] = lat;
04050
04051
            los \rightarrow z[los \rightarrow np] = z;
04052
            los \rightarrow p[los \rightarrow np] = p;
04053
04054
            los->t[los->np] = t;
04055
            for (ig = 0; ig < ctl->ng; ig++)
04056
              los->q[ig][los->np] = q[ig];
            for (iw = 0; iw < ctl->nw; iw++)
los->k[iw][los->np] = k[iw];
04057
04058
04059
            los \rightarrow ds[los \rightarrow np] = ds;
04060
04061
            /\star Increment and check number of LOS points... \star/
04062
            if ((++los->np) > NLOS)
04063
              ERRMSG("Too many LOS points!");
04064
04065
            /* Check stop flag... */
04066
            if (stop) {
04067
              los->tsurf = (stop == 2 ? t : -999);
04068
              break;
04069
04070
04071
            /* Determine refractivity... */
04072
           if (ctl->refrac && z <= zrefrac)</pre>
04073
             n = 1 + refractivity(p, t);
04074
04075
             n = 1;
04076
04077
            /* Construct new tangent vector (first term)... */
04078
           for (i = 0; i < 3; i++)
04079
              ex1[i] = ex0[i] * n;
04080
04081
            /* Compute gradient of refractivity... */
04082
            if (ctl->refrac && z <= zrefrac) {
              for (i = 0; i < 3; i++)

xh[i] = x[i] + 0.5 * ds * ex0[i];

cart2geo(xh, &z, &lon, &lat);
04083
04084
04086
              intpol_atm(ctl, atm, z, &p, &t, q, k);
              n = refractivity(p, t);
for (i = 0; i < 3; i++) {
   xh[i] += h;</pre>
04087
04088
04089
04090
                cart2geo(xh, &z, &lon, &lat);
04091
                 intpol_atm(ctl, atm, z, &p, &t, q, k);
04092
                naux = refractivity(p, t);
                ng[i] = (naux - n) / h;
xh[i] -= h;
04093
04094
04095
04096
           } else
04097
              for (i = 0; i < 3; i++)</pre>
04098
                ng[i] = 0;
04099
04100
            /\star Construct new tangent vector (second term)... \star/
04101
           for (i = 0; i < 3; i++)
ex1[i] += ds * ng[i];</pre>
04102
04103
04104
            /* Normalize new tangent vector... */
04105
           norm = NORM(ex1);
            for (i = 0; i < 3; i++)</pre>
04106
04107
              ex1[i] /= norm;
04108
04109
            /\star Determine next point of LOS... \star/
04110
           for (i = 0; i < 3; i++)
04111
              x[i] += 0.5 * ds * (ex0[i] + ex1[i]);
04112
           /* Copy tangent vector... */
for (i = 0; i < 3; i++)
  ex0[i] = ex1[i];</pre>
04113
04114
04115
04116
04117
04118
         /* Get tangent point (to be done before changing segment lengths!)... */
04119
        tangent_point(los, &obs->tpz[ir], &obs->tplon[ir], &obs->
       tplat[ir]);
04120
```

```
/\star Change segment lengths according to trapezoid rule... \star/
        for (ip = los->np - 1; ip >= 1; ip--)
los->ds[ip] = 0.5 * (los->ds[ip - 1] + los->ds[ip]);
04122
04123
04124
        los->ds[0] *= 0.5;
04125
        /* Compute column density... */
for (ip = 0; ip < los->np; ip++)
04126
04127
04128
           for (ig = 0; ig < ctl->ng; ig++)
04129
             los \rightarrow u[ig][ip] = 10 * los \rightarrow q[ig][ip] * los \rightarrow p[ip]
04130
                / (KB * los->t[ip]) * los->ds[ip];
04131 }
04132
04134
04135 void read_atm(
        const char *dirname, const char *filename,
04136
04137
        ctl_t * ctl,
04138
        atm_t * atm)
04139
04140
04141
        FILE *in;
04142
04143
        char file[LEN], line[LEN], *tok;
04144
04145
        int iq, iw;
04146
04147
         /* Init... */
04148
        atm->np = 0;
04149
04150
         /* Set filename... */
04151
         if (dirname != NULL)
04152
           sprintf(file, "%s/%s", dirname, filename);
04153
04154
           sprintf(file, "%s", filename);
04155
        /* Write info... */
04156
        printf("Read atmospheric data: %s\n", file);
04157
04158
04159
         /* Open file... *
04160
        if (!(in = fopen(file, "r")))
           ERRMSG("Cannot open file!");
04161
04162
04163
        /* Read line... */
04164
        while (fgets(line, LEN, in)) {
04165
          /* Read data... */

TOK(line, tok, "%lg", atm->time[atm->np]);

TOK(NULL, tok, "%lg", atm->z[atm->np]);

TOK(NULL, tok, "%lg", atm->lon[atm->np]);

TOK(NULL, tok, "%lg", atm->lat[atm->np]);

TOK(NULL, tok, "%lg", atm->p[atm->np]);

TOK(NULL, tok, "%lg", atm->t[atm->np]);

for (ig = 0; ig < ctl->ng; ig++)

TOK(NULL, tok, "%lg", atm->q[ig][atm->np]);

for (iw = 0; iw < ctl->nw; iw++)

TOK(NULL, tok, "%lg", atm->k[iw][atm->np]);
04166
04167
04168
04169
04170
04171
04172
04173
04174
04175
04176
04177
04178
           /* Increment data point counter... */
04179
           if ((++atm->np) > NP)
04180
             ERRMSG("Too many data points!");
04181
04182
04183
         /* Close file... */
04184
         fclose(in);
04185
04186
         /* Check number of points... */
04187
         if (atm->np < 1)
           ERRMSG("Could not read any data!");
04188
04189 }
04190
04192
04193 void read_ctl(
04194
        int argc,
        char *argv[],
ctl_t * ctl) {
04195
04196
04197
04198
        int id, ig, iw;
04199
04200
        /* Write info... */
        04201
04202
04203
                 argv[0], __DATE__, __TIME__);
04204
        /* Emitters... */
04205
        ctl->ng = (int) scan_ctl(argc, argv, "NG", -1, "0", NULL);
if (ctl->ng < 0 || ctl->ng > NG)
04206
04207
```

```
04208
            ERRMSG("Set 0 <= NG <= MAX!");</pre>
          for (ig = 0; ig < ctl->ng; ig++)
    scan_ctl(argc, argv, "EMITTER", ig, "", ctl->emitter[ig]);
04209
04210
04211
04212
          /* Radiance channels... */
          ctl->nd = (int) scan_ctl(argc, argv, "ND", -1, "0", NULL);
if (ctl->nd < 0 || ctl->nd > ND)
04213
04214
04215
             ERRMSG("Set 0 <= ND <= MAX!");</pre>
04216
          for (id = 0; id < ctl->nd; id++)
            ctl->nu[id] = scan_ctl(argc, argv, "NU", id, "", NULL);
04217
04218
04219
          /* Spectral windows... */
          ctl->nw = (int) scan_ctl(argc, argv, "NW", -1, "1", NULL);
04220
04221
          if (ctl->nw < 0 || ctl->nw > NW)
04222
            ERRMSG("Set 0 <= NW <= MAX!");</pre>
04223
          for (id = 0; id < ctl->nd; id++)
            ctl->window[id] = (int) scan_ctl(argc, argv, "WINDOW", id, "0", NULL);
04224
04225
          /* Emissivity look-up tables... */
scan_ctl(argc, argv, "TBLBASE", -1, "-", ctl->tblbase);
04227
04228
04229
          /* Hydrostatic equilibrium... */
          ctl->hydz = scan_ctl(argc, argv, "HYDZ", -1, "-999", NULL);
04230
04231
04232
          /* Continua... */
          /* Continua... */
ctl->ctm_co2 = (int) scan_ctl(argc, argv, "CTM_CO2", -1, "1", NULL);
ctl->ctm_h2o = (int) scan_ctl(argc, argv, "CTM_H2O", -1, "1", NULL);
ctl->ctm_n2 = (int) scan_ctl(argc, argv, "CTM_N2", -1, "1", NULL);
ctl->ctm_o2 = (int) scan_ctl(argc, argv, "CTM_O2", -1, "1", NULL);
04233
04234
04235
04236
04237
04238
          /* Ray-tracing...
          ctl->refrac = (int) scan_ctl(argc, argv, "REFRAC", -1, "1", NULL);
ctl->rayds = scan_ctl(argc, argv, "RAYDS", -1, "10", NULL);
ctl->raydz = scan_ctl(argc, argv, "RAYDZ", -1, "0.5", NULL);
04239
04240
04241
04242
          /* Field of view... */
scan_ctl(argc, argv, "FOV", -1, "-", ctl->fov);
04243
04244
04246
           /* Retrieval interface... */
          /* Retrieval interface... */
ctl->retp_zmin = scan_ctl(argc, argv, "RETP_ZMIN", -1, "-999", NULL);
ctl->retp_zmax = scan_ctl(argc, argv, "RETP_ZMAX", -1, "-999", NULL);
ctl->rett_zmin = scan_ctl(argc, argv, "RETT_ZMIN", -1, "-999", NULL);
ctl->rett_zmax = scan_ctl(argc, argv, "RETT_ZMAX", -1, "-999", NULL);
04247
04248
04249
04250
          for (ig = 0; ig < ctl->ng; ig++) {
04251
           ctl->retq_zmin[ig] = scan_ctl(argc, argv, "RETO_ZMIN", ig, "-999", NULL); ctl->retq_zmax[ig] = scan_ctl(argc, argv, "RETO_ZMAX", ig, "-999", NULL);
04252
04253
04254
04255
          for (iw = 0; iw < ctl->nw; iw++) {
            ctl->retk_zmin[iw] = scan_ctl(argc, argv, "RETK_ZMIN", iw, "-999", NULL);
ctl->retk_zmax[iw] = scan_ctl(argc, argv, "RETK_ZMAX", iw, "-999", NULL);
04256
04257
04258
04259
04260
          /* Output flags... */
04261
          ctl->write_bbt = (int) scan_ctl(argc, argv, "WRITE_BBT", -1, "0", NULL);
          ctl->write_matrix =
04262
             (int) scan_ctl(argc, argv, "WRITE_MATRIX", -1, "0", NULL);
04263
04264 }
04265
04267
04268 void read matrix(
04269 const char *dirname,
          const char *filename,
04271
          gsl_matrix * matrix) {
04272
04273
         FILE *in;
04274
04275
          char dum[LEN], file[LEN], line[LEN];
04276
04277
          double value;
04278
04279
          int i, j;
04280
04281
          /* Set filename... */
04282
          if (dirname != NULL)
            sprintf(file, "%s/%s", dirname, filename);
04283
04284
04285
            sprintf(file, "%s", filename);
04286
04287
          /* Write info... */
04288
          printf("Read matrix: %s\n", file);
04289
04290
          if (!(in = fopen(file, "r")))
04291
            ERRMSG("Cannot open file!");
04292
04293
04294
          /* Read data... */
```

5.4 jurassic.c 143

```
gsl_matrix_set_zero(matrix);
         04296
04297
04298
                        &i, dum, dum, dum, dum, dum,
             &j, dum, dum, dum, dum, dum, &value) == 13)
gsl_matrix_set(matrix, (size_t) i, (size_t) j, value);
04299
04300
04301
04302
         /* Close file... */
04303
        fclose(in);
04304 }
04305
04307
04308 void read_obs(
04309
         const char *dirname,
04310
         const char *filename,
04311
         ctl_t * ctl,
         obs_t * obs) {
04312
04313
04314
         FILE *in;
04315
04316
         char file[LEN], line[LEN], *tok;
04317
04318
         int id:
04319
04320
         /* Init... */
04321
         obs->nr = 0;
04322
04323
         /* Set filename...
04324
         if (dirname != NULL)
04325
           sprintf(file, "%s/%s", dirname, filename);
04326
         else
04327
           sprintf(file, "%s", filename);
04328
04329
         /* Write info... */
         printf("Read observation data: %s\n", file);
04330
04331
04332
         /* Open file... */
04333
         if (!(in = fopen(file, "r")))
04334
           ERRMSG("Cannot open file!");
04335
04336
         /* Read line... */
         while (fgets(line, LEN, in)) {
04337
04338
           /* Read data... */
TOK(line, tok, "%lg", obs->time[obs->nr]);
TOK(NULL, tok, "%lg", obs->obsz[obs->nr]);
TOK(NULL, tok, "%lg", obs->obslon[obs->nr]);
TOK(NULL, tok, "%lg", obs->obslat[obs->nr]);
TOK(NULL, tok, "%lg", obs->vpz[obs->nr]);
04339
04340
04341
04342
04343
04344
           TOK (NULL, tok, "%1g", obs->vpz[obs->nr]);
TOK (NULL, tok, "%1g", obs->vplon[obs->nr]);
TOK (NULL, tok, "%1g", obs->vplat[obs->nr]);
TOK (NULL, tok, "%1g", obs->tpz[obs->nr]);
TOK (NULL, tok, "%1g", obs->tplon[obs->nr]);
TOK (NULL, tok, "%1g", obs->tplat[obs->nr]);
for (id = 0; id < ctl->nd; id+)

TOK (NULL, tok, "%1g", obs->rad[id][obs->nr]);
04345
04346
04347
04348
04349
04350
04351
           for (id = 0; id < ctl->nd; id++)
TOK(NULL, tok, "%lg", obs->tau[id][obs->nr]);
04352
04353
04354
04355
            /* Increment counter... */
           if ((++obs->nr) > NR)
04356
04357
              ERRMSG("Too many rays!");
04358
04359
04360
         /* Close file... */
04361
         fclose(in);
04362
         /\star Check number of points... \star/
04363
         if (obs->nr < 1)</pre>
04364
04365
           ERRMSG("Could not read any data!");
04366 }
04367
04369
04370 void read_shape(
04371
         const char *filename,
04372
         double *x,
         double *y,
04373
04374
         int *n) {
04375
04376
         FILE *in;
04377
04378
         char line[LEN];
04379
         /* Write info... */
04380
04381
        printf("Read shape function: %s\n", filename);
```

```
04382
04383
        /* Open file... */
        if (!(in = fopen(filename, "r")))
04384
          ERRMSG("Cannot open file!");
04385
04386
        /* Read data... */
04387
04388
        *n = 0;
04389
        while (fgets(line, LEN, in))
        if (sscanf(line, "%lg %lg", &x[*n], &y[*n]) == 2)
if ((++(*n)) > NSHAPE)
    ERRMSG("Too many data points!");
04390
04391
04392
04393
04394
        /* Check number of points... */
04395
04396
          ERRMSG("Could not read any data!");
04397
       /* Close file... */
04398
04399
       fclose(in);
04400 }
04401
04403
04404 double refractivity(
04405
       double p,
04406
       double t) {
04407
04408
        /* Refractivity of air at 4 to 15 micron... */
04409
       return 7.753e-05 * p / t;
04410 }
04411
04413
04414 double scan_ctl(
04415
        int argc,
04416
        char *argv[],
        const char *varname.
04417
04418
        int arridx,
04419
        const char *defvalue,
04420
        char *value) {
04421
04422
       FILE *in = NULL;
04423
       char dummy[LEN], fullname1[LEN], fullname2[LEN], line[LEN],
  msg[2 * LEN], rvarname[LEN], rval[LEN];
04424
04425
04426
04427
        int contain = 0, i;
04428
        /* Open file... */
if (argv[1][0] != '-')
04429
04430
04431
         if (!(in = fopen(argv[1], "r")))
            ERRMSG("Cannot open file!");
04432
04433
04434
        /* Set full variable name... */
        if (arridx >= 0) {
   sprintf(fullname1, "%s[%d]", varname, arridx);
   sprintf(fullname2, "%s[*]", varname);
04435
04436
04437
04438
        } else {
         sprintf(fullname1, "%s", varname);
sprintf(fullname2, "%s", varname);
04439
04440
04441
04442
04443
        /* Read data... */
04444
        if (in != NULL)
04445
         while (fgets(line, LEN, in))
            if (sscanf(line, "%s %s %s", rvarname, dummy, rval) == 3)
04446
              if (strcasecmp(rvarname, fullname1) == 0 ||
04447
04448
                  strcasecmp(rvarname, fullname2) == 0) {
04449
                 contain = 1:
04450
                break:
04451
              }
04452
        for (i = 1; i < argc - 1; i++)</pre>
04453
         if (strcasecmp(argv[i], fullname1) == 0 ||
            strcasecmp(argv[i], fullname2) == 0) {
sprintf(rval, "%s", argv[i + 1]);
04454
04455
04456
            contain = 1;
04457
            break;
04458
04459
        /* Close file... */
if (in != NULL)
04460
04461
04462
          fclose(in);
04463
04464
        /* Check for missing variables... */
04465
        if (!contain) {
        if (strlen(defvalue) > 0)
   sprintf(rval, "%s", defvalue);
04466
04467
04468
          else {
```

5.4 jurassic.c 145

```
sprintf(msg, "Missing variable %s!\n", fullname1);
04470
04471
       }
04472
04473
04474
       /* Write info... */
       printf("%s = %s\n", fullname1, rval);
04476
04477
        /* Return values... */
       if (value != NULL)
    sprintf(value, "%s", rval);
04478
04479
04480
       return atof(rval);
04481 }
04482
04484
04485 void tangent_point(
       los_t * los,
double *tpz,
04486
04487
04488
       double *tplon,
       double *tplat) {
04489
04490
04491
       double a, b, c, dummy, v[3], v0[3], v2[3], x, x1, x2, yy0, yy1, yy2;
04492
04493
       size_t i, ip;
04494
04495
       /\star Find minimum altitude... \star/
04496
       ip = gsl_stats_min_index(los->z, 1, (size_t) los->np);
04497
       /* Nadir or zenith... */
if (ip <= 0 || ip >= (size_t) los->np - 1) {
04498
04499
04500
         *tpz = los->z[los->np - 1];
04501
         *tplon = los->lon[los->np - 1];
04502
          *tplat = los->lat[los->np - 1];
04503
04504
04505
       /* Limb... */
       else {
04507
04508
          /* Determine interpolating polynomial y=a*x^2+b*x+c... */
04509
         yy0 = los \rightarrow z[ip - 1];
         yy1 = los \rightarrow z[ip];
04510
          yy2 = los -> z[ip + 1];
04511
04512
         x1 = sqrt(POW2(los->ds[ip]) - POW2(yy1 - yy0));
04513
         x2 = x1 + sqrt(POW2(los->ds[ip + 1]) - POW2(yy2 - yy1));
04514
          a = 1 / (x1 - x2) * (-(yy0 - yy1) / x1 + (yy0 - yy2) / x2);
         b = -(yy0 - yy1) / x1 - a * x1;
04515
         c = yy0;
04516
04517
04518
          /* Get tangent point location... */
         x = -b / (2 * a);
04520
          *tpz = a * x * x + b * x + c;
          geo2cart(los->z[ip - 1], los->lon[ip - 1], los->lat[ip - 1], v0);
geo2cart(los->z[ip + 1], los->lon[ip + 1], los->lat[ip + 1], v2);
04521
04522
          for (i = 0; i < 3; i++)
04523
           v[i] = LIN(0.0, v0[i], x2, v2[i], x);
04524
          cart2geo(v, &dummy, tplon, tplat);
04526
04527 }
04528
04530
04531 void time2jsec(
04532
       int year,
       int mon,
04533
04534
       int day,
04535
       int hour,
04536
       int min.
04537
       int sec.
       double remain,
04539
       double *jsec) {
04540
04541
       struct tm t0, t1;
04542
       t0.tm_year = 100;
04543
04544
       t0.tm\_mon = 0;
04545
       t0.tm_mday = 1;
04546
       t0.tm\_hour = 0;
       t0.tm_min = 0;
04547
       t0.tm_sec = 0;
04548
04549
04550
       t1.tm_year = year - 1900;
04551
       t1.tm_mon = mon - 1;
04552
       t1.tm_mday = day;
04553
       t1.tm_hour = hour;
       t1.tm_min = min;
04554
04555
       t1.tm_sec = sec;
```

```
04557
       *jsec = (double) timegm(&t1) - (double) timegm(&t0) + remain;
04558 }
04559
04561
04562 void timer(
04563
       const char *name,
04564
        const char *file,
04565
       const char *func,
       int line,
04566
04567
       int mode) {
04568
04569
       static double w0[10];
04570
04571
       static int 10[10], nt;
04572
04573
        /* Start new timer... */
04574
       if (mode == 1) {
04575
        w0[nt] = omp_get_wtime();
04576
         10[nt] = line;
         if ((++nt) >= 10)
    ERRMSG("Too many timers!");
04577
04578
04579
04580
04581
        /* Write elapsed time... */
04582
        else {
04583
04584
          /\star Check timer index... \star/
         if (nt - 1 < 0)
04585
04586
           ERRMSG("Coding error!");
04587
04588
          /* Write elapsed time... */
04589
         printf("Timer '%s' (%s, %s, 1%d-%d): %.3f sec\n",
                 name, file, func, 10[nt - 1], line, omp_get_wtime() - w0[nt - 1]);
04590
04591
04592
04593
        /* Stop timer... */
04594
        if (mode == 3)
04595
         nt--;
04596 }
04597
04599
04600 void write_atm(
04601
       const char *dirname,
04602
       const char *filename,
04603
       ctl_t * ctl,
       atm_t * atm) {
04604
04605
04606
       FILE *out;
04607
04608
       char file[LEN];
04609
       int ig, ip, iw, n = 6;
04610
04611
04612
       /* Set filename... */
04613
        if (dirname != NULL)
04614
         sprintf(file, "%s/%s", dirname, filename);
04615
        else
04616
         sprintf(file, "%s", filename);
04617
04618
        /* Write info... */
04619
       printf("Write atmospheric data: %s\n", file);
04620
04621
        /* Create file... */
        if (!(out = fopen(file, "w")))
04622
         ERRMSG("Cannot create file!");
04623
04624
04625
        /* Write header... */
04626
        fprintf(out,
04627
                "# $1 = time (seconds since 2000-01-01T00:00Z) \n"
                "# $2 = altitude [km] \n"
04628
                "# $3 = longitude [deg]\n"
04629
               "# $4 = latitude [deg]\n"
"# $5 = pressure [hPa]\n" "# $6 = temperature [K]\n");
04630
04631
       for (ig = 0; ig < ctl->ng; ig++)
  fprintf(out, "# $%d = %s volume mixing ratio\n", ++n, ctl->emitter[ig]);
04632
04633
        for (iw = 0; iw < ctl->nw; iw++)
  fprintf(out, "# $%d = window %d: extinction [1/km]\n", ++n, iw);
04634
04635
04636
04637
        /* Write data... */
04638
        for (ip = 0; ip < atm->np; ip++) {
04639
         if (ip == 0 || atm->lat[ip] != atm->lat[ip - 1]
         || atm->lon[ip] != atm->lon[ip - 1])
fprintf(out, "\n");
fprintf(out, "%.2f %g %g %g %g", atm->time[ip], atm->z[ip],
04640
04641
04642
```

5.4 jurassic.c 147

```
atm->lon[ip], atm->lat[ip], atm->p[ip], atm->t[ip]);
         for (ig = 0; ig < ctl->ng; ig++)
  fprintf(out, " %g", atm->q[ig][ip]);
for (iw = 0; iw < ctl->nw; iw++)
  fprintf(out, " %g", atm->k[iw][ip]);
fprintf(out, "\n");
04644
04645
04646
04647
04648
04649
04650
04651
        /* Close file... */
04652
       fclose(out);
04653 }
04654
04656
04657 void write_matrix(
       const char *dirname,
const char *filename,
04658
04659
        ctl t * ctl,
04660
04661
        gsl_matrix * matrix,
04662
        atm_t * atm,
        obs_t * obs,
04663
04664
        const char *rowspace,
04665
        const char *colspace,
04666
        const char *sort) {
04667
04668
       FILE *out;
04669
04670
        char file[LEN], quantity[LEN];
04671
04672
        int *cida, *ciqa, *cipa, *cira, *rida, *riqa, *ripa, *rira;
04673
04674
        size t i, j, nc, nr;
04675
04676
        /* Check output flag... */
04677
        if (!ctl->write_matrix)
04678
          return;
04679
        /* Allocate... */
04680
04681
        ALLOC(cida, int, M);
04682
        ALLOC(ciqa, int,
04683
             N);
        ALLOC(cipa, int,
04684
04685
              N):
        ALLOC(cira, int,
04686
04687
              M);
04688
        ALLOC(rida, int,
04689
             M);
        ALLOC(riqa, int,
04690
04691
              N);
04692
        ALLOC(ripa, int,
04693
              N);
04694
        ALLOC(rira, int,
04695
             M);
04696
        /* Set filename... */
04697
04698
        if (dirname != NULL)
04699
         sprintf(file, "%s/%s", dirname, filename);
04700
04701
         sprintf(file, "%s", filename);
04702
04703
        /* Write info... */
04704
        printf("Write matrix: %s\n", file);
04705
04706
        /* Create file... */
        if (!(out = fopen(file, "w")))
04707
         ERRMSG("Cannot create file!");
04708
04709
04710
        /* Write header (row space)... */
        if (rowspace[0] == 'v') {
04711
04712
04713
          fprintf (out,
04714
                   "# $1 = Row: index (measurement space) \n"
                   "# $2 = Row: channel wavenumber [cm^-1]\n"
04715
04716
                   "# $3 = Row: time (seconds since 2000-01-01T00:00Z)\n"
                  "# $4 = Row: view point altitude [km]\n"
"# $5 = Row: view point longitude [deg]\n"
04717
04718
04719
                   "# $6 = Row: view point latitude [deg] n");
04720
04721
          /* Get number of rows... */
         nr = obs2y(ct1, obs, NULL, rida, rira);
04722
04723
04724
        } else {
04725
04726
          fprintf(out,
04727
                   "# $1 = Row: index (state space) \n"
                   "# $2 = Row: name of quantity\n"
04728
04729
                   "# $3 = Row: time (seconds since 2000-01-01T00:00Z)\n"
```

```
"# $4 = Row: altitude [km]\n"
04731
                 "# $5 = Row: longitude [deg]\n" "# $6 = Row: latitude [deg]\n");
04732
04733
         /* Get number of rows... */
04734
         nr = atm2x(ctl, atm, NULL, riqa, ripa);
04735
04736
04737
        /* Write header (column space)... */
04738
       if (colspace[0] == 'y') {
04739
04740
         fprintf(out,
04741
                 "# $7 = Col: index (measurement space) \n"
                 "# $8 = Col: channel wavenumber [cm^-1]\n"
04742
04743
                 "# $9 = Col: time (seconds since 2000-01-01T00:00Z)\n"
04744
                 "# $10 = Col: view point altitude [km]\n"
                 "# $11 = Col: view point longitude [deg] n"
04745
                 "# $12 = Col: view point latitude [deg]\n");
04746
04747
04748
         /* Get number of columns... */
04749
         nc = obs2y(ctl, obs, NULL, cida, cira);
04750
04751
       } else {
04752
04753
         fprintf(out,
    "# $7 = Col: index (state space)\n"
04754
04755
                 "# $8 = Col: name of quantity\n"
04756
                 "# $9 = Col: time (seconds since 2000-01-01T00:00Z)\n"
04757
                 "# $10 = Col: altitude [km] \n"
                 "# $11 = Col: longitude [deg]\n" "# $12 = Col: latitude [deg]\n");
04758
04759
04760
          /* Get number of columns... */
04761
         nc = atm2x(ctl, atm, NULL, ciqa, cipa);
04762
04763
       /* Write header entry... */
fprintf(out, "# $13 = Matrix element\n\n");
04764
04765
04766
04767
        /* Write matrix data... */
04768
       i = j = 0;
04769
       while (i < nr && j < nc) {
04770
04771
         /* Write info about the row... */
         04772
04774
04775
                   obs->time[rira[i]], obs->vpz[rira[i]],
04776
                   obs->vplon[rira[i]], obs->vplat[rira[i]]);
04777
         else {
04778
           04779
04780
04781
                   atm->lon[ripa[i]], atm->lat[ripa[i]]);
04782
04783
         /* Write info about the column... */
if (colspace[0] == 'y')
  fprintf(out, " %d %g %.2f %g %g %g",
04784
04785
04786
04787
                   (int) j, ctl->nu[cida[j]],
04788
                   obs->time[cira[j]], obs->vpz[cira[j]],
04789
                   obs->vplon[cira[j]], obs->vplat[cira[j]]);
04790
         else (
           04791
04793
04794
04795
04796
         04797
04798
04799
04800
          /* Set matrix indices... */
04801
         if (sort[0] == 'r') {
           j++;
if (j >= nc) {
04802
04803
             j = 0;
i++;
04804
04805
04806
             fprintf(out, "\n");
04807
04808
         } else {
04809
           i++:
           if (i >= nr) {
04810
             i = 0;
04811
04812
04813
             fprintf(out, "\n");
04814
           }
      }
04815
04816
```

5.4 jurassic.c 149

```
04817
        /* Close file... */
04818
04819
       fclose(out);
04820
04821
       /* Free... */
04822
       free(cida);
04823
       free(ciqa);
04824
        free(cipa);
04825
       free(cira);
04826
       free (rida);
04827
       free (riga);
04828
       free (ripa):
04829
       free (rira);
04830 }
04831
04833
04834 void write obs(
      const char *dirname,
       const char *filename,
04836
04837
       ctl_t * ctl,
       obs_t * obs)
04838
04839
04840
       FILE *out:
04841
04842
       char file[LEN];
04843
04844
       int id, ir, n = 10;
04845
04846
       /* Set filename... */
04847
       if (dirname != NULL)
04848
         sprintf(file, "%s/%s", dirname, filename);
04849
04850
         sprintf(file, "%s", filename);
04851
       /* Write info... */
04852
       printf("Write observation data: %s\n", file);
04853
04854
04855
       /* Create file... *
04856
       if (!(out = fopen(file, "w")))
04857
         ERRMSG("Cannot create file!");
04858
04859
        /* Write header... */
04860
       fprintf(out,
               "# $1 = time (seconds since 2000-01-01T00:00Z)\n"
04861
04862
               "# $2 = observer altitude [km] \n"
04863
               "# $3 = observer longitude [deg]\n"
                "# $4 = observer latitude [deg] \n"
04864
               "# $5 = view point altitude [km]\n"
"# $6 = view point longitude [deg]\n"
04865
04866
               "# $7 = view point latitude [deg]\n"
04867
04868
               "# $8 = tangent point altitude [km]\n"
04869
               "# $9 = tangent point longitude [deg]\n"
04870
               "# $10 = tangent point latitude [deg]\n");
       for (id = 0; id < ctl->nd; id++)
04871
        fprintf(out, "# \$%d = channel \$g: radiance [W/(m^2 sr cm^-1)]\n",
04872
                 ++n, ctl->nu[id]);
04874
       for (id = 0; id < ctl->nd; id++)
04875
         fprintf(out, "# $%d = channel %g: transmittance\n", ++n, ctl->nu[id]);
04876
04877
       /* Write data... */
       for (ir = 0; ir < obs->nr; ir++) {
04878
04879
         if (ir == 0 || obs->time[ir] != obs->time[ir - 1])
          fprintf(out, "\n");
fprintf(out, "%.2f %g %g %g %g %g %g %g %g %g", obs->time[ir],
04880
04881
04882
                 obs->obsz[ir], obs->obslon[ir], obs->obslat[ir],
04883
                 obs->vpz[ir], obs->vplon[ir], obs->vplat[ir],
                 obs->tpz[ir], obs->tplon[ir], obs->tplat[ir]);
04884
04885
          for (id = 0; id < ctl->nd; id++)
           fprintf(out, " %g", obs->rad[id][ir]);
         for (id = 0; id < otl->nd; id+)
  fprintf(out, " %g", obs->tau[id][ir]);
fprintf(out, "\n");
04887
04888
04889
04890
04891
04892
        /* Close file... */
04893
       fclose(out);
04894 }
04895
04897
04898 void x2atm(
04899
       ctl_t * ctl,
04900
       gsl_vector * x,
04901
       atm_t * atm) {
04902
04903
       int ia, iw:
```

```
04904
04905
        size_t n = 0;
04906
04907
       /* Set pressure... */
p, x, &n);
04909
       x2atm_help(atm, ctl->retp_zmin, ctl->retp_zmax, atm->
04910
       /* Set temperature... */
04911 x2atm_help(atm, ctl->rett_zmin, ctl->rett_zmax, atm->
      t, x, &n);
04912
04913
       /* Set volume mixing ratio... */
       for (ig = 0; ig < ctl->ng; ig++)
04914
04915
        x2atm_help(atm, ctl->retq_zmin[ig], ctl->retq_zmax[ig],
04916
                     atm->q[ig], x, &n);
04917
04918
       /* Set extinction... */
       for (iw = 0; iw < ctl->nw; iw++)
04919
04920
        x2atm_help(atm, ctl->retk_zmin[iw], ctl->retk_zmax[iw],
04921
                     atm->k[iw], x, &n);
04922 }
04923
04925
04926 void x2atm_help(
04927 atm_t * atm,
04928
       double zmin,
04929
       double zmax,
04930
       double *value,
04931
       gsl\_vector * x,
04932
       size t * n) {
04933
04934
04935
        /* Extract state vector elements... */
for (ip = 0; ip < atm->np; ip++)
  if (atm->z[ip] >= zmin && atm->z[ip] <= zmax) {
    value[ip] = gsl_vector_get(x, *n);</pre>
04936
04937
04938
04940
            (*n)++;
04941
04942 }
04943
04945
04946 void y2obs(
04947
       ctl_t * ctl,
04948
       gsl_vector * y,
04949
       obs_t * obs) {
04950
04951
       int id, ir;
04952
04953
       size_t m = 0;
04954
04955
        /\star Decompose measurement vector... \star/
       for (ir = 0; ir < obs->nr; ir++)
  for (id = 0; id < ctl->nd; id++)
    if (gsl_finite(obs->rad[id][ir])) {
04956
04957
04958
04959
             obs->rad[id][ir] = gsl_vector_get(y, m);
04960
             m++;
04961
04962 }
```

5.5 jurassic.h File Reference

JURASSIC library declarations.

Data Structures

• struct atm t

Atmospheric data.

struct ctl_t

Forward model control parameters.

• struct los_t

Line-of-sight data.

```
    struct obs t
```

Observation geometry and radiance data.

struct tbl t

Emissivity look-up tables.

Functions

size_t atm2x (ctl_t *ctl, atm_t *atm, gsl_vector *x, int *iqa, int *ipa)

Compose state vector or parameter vector.

void atm2x_help (atm_t *atm, double zmin, double zmax, double *value, int val_iqa, gsl_vector *x, int *iqa, int *ipa, size_t *n)

Add elements to state vector.

double brightness (double rad, double nu)

Compute brightness temperature.

void cart2geo (double *x, double *z, double *lon, double *lat)

Convert Cartesian coordinates to geolocation.

void climatology (ctl_t *ctl, atm_t *atm_mean)

Interpolate climatological data.

• double ctmco2 (double nu, double p, double t, double u)

Compute carbon dioxide continuum (optical depth).

double ctmh2o (double nu, double p, double t, double q, double u)

Compute water vapor continuum (optical depth).

• double ctmn2 (double nu, double p, double t)

Compute nitrogen continuum (absorption coefficient).

double ctmo2 (double nu, double p, double t)

Compute oxygen continuum (absorption coefficient).

void copy_atm (ctl_t *ctl, atm_t *atm_dest, atm_t *atm_src, int init)

Copy and initialize atmospheric data.

void copy_obs (ctl_t *ctl, obs_t *obs_dest, obs_t *obs_src, int init)

Copy and initialize observation data.

• int find_emitter (ctl_t *ctl, const char *emitter)

Find index of an emitter.

void formod (ctl_t *ctl, atm_t *atm, obs_t *obs)

Determine ray paths and compute radiative transfer.

• void formod_continua (ctl_t *ctl, los_t *los, int ip, double *beta)

Compute absorption coefficient of continua.

void formod_fov (ctl_t *ctl, obs_t *obs)

Apply field of view convolution.

• void formod_pencil (ctl_t *ctl, atm_t *atm, obs_t *obs, int ir)

Compute radiative transfer for a pencil beam.

• void formod_srcfunc (ctl_t *ctl, tbl_t *tbl, double t, double *src)

Compute Planck source function.

void geo2cart (double z, double lon, double lat, double *x)

Convert geolocation to Cartesian coordinates.

void hydrostatic (ctl_t *ctl, atm_t *atm)

Set hydrostatic equilibrium.

void idx2name (ctl_t *ctl, int idx, char *quantity)

Determine name of state vector quantity for given index.

void init_tbl (ctl_t *ctl, tbl_t *tbl)

Initialize look-up tables.

```
    void intpol_atm (ctl_t *ctl, atm_t *atm, double z, double *p, double *t, double *q, double *k)

      Interpolate atmospheric data.

    void intpol tbl (ctl t *ctl, tbl t *tbl, los t *los, int ip, double tau path[NG][ND], double tau seg[ND])

      Get transmittance from look-up tables.
• double intpol_tbl_eps (tbl_t *tbl, int ig, int id, int ip, int it, double u)
      Interpolate emissivity from look-up tables.

    double intpol_tbl_u (tbl_t *tbl, int ig, int id, int ip, int it, double eps)

      Interpolate column density from look-up tables.

    void jsec2time (double jsec, int *year, int *mon, int *day, int *hour, int *min, int *sec, double *remain)

      Convert seconds to date.

    void kernel (ctl t *ctl, atm t *atm, obs t *obs, gsl matrix *k)

      Compute Jacobians.

    int locate_irr (double *xx, int n, double x)

      Find array index for irregular grid.

    int locate_reg (double *xx, int n, double x)

      Find array index for regular grid.

    int locate_tbl (float *xx, int n, double x)

      Find array index in float array.

    size_t obs2y (ctl_t *ctl, obs_t *obs, gsl_vector *y, int *ida, int *ira)

      Compose measurement vector.
• double planck (double t, double nu)
      Compute Planck function.

    void raytrace (ctl t *ctl, atm t *atm, obs t *obs, los t *los, int ir)

      Do ray-tracing to determine LOS.

    void read_atm (const char *dirname, const char *filename, ctl_t *ctl, atm_t *atm)

      Read atmospheric data.
void read_ctl (int argc, char *argv[], ctl_t *ctl)
      Read forward model control parameters.

    void read_matrix (const char *dirname, const char *filename, gsl_matrix *matrix)

      Read matrix.

    void read_obs (const char *dirname, const char *filename, ctl_t *ctl, obs_t *obs)

      Read observation data.

    void read_shape (const char *filename, double *x, double *y, int *n)

      Read shape function.

    double refractivity (double p, double t)

      Compute refractivity (return value is n - 1).
• double scan_ctl (int argc, char *argv[], const char *varname, int arridx, const char *defvalue, char *value)
      Search control parameter file for variable entry.

    void tangent point (los t *los, double *tpz, double *tplon, double *tplat)

      Find tangent point of a given LOS.

    void time2jsec (int year, int mon, int day, int hour, int min, int sec, double remain, double *jsec)

      Convert date to seconds.

    void timer (const char *name, const char *file, const char *func, int line, int mode)

      Measure wall-clock time.
• void write_atm (const char *dirname, const char *filename, ctl_t *ctl, atm_t *atm)
      Write atmospheric data.
• void write_matrix (const char *dirname, const char *filename, ctl_t *ctl, gsl_matrix *matrix, atm_t *atm,
  obs_t *obs, const char *rowspace, const char *colspace, const char *sort)
      Write matrix.

    void write obs (const char *dirname, const char *filename, ctl t *ctl, obs t *obs)

      Write observation data.
```

```
    void x2atm (ctl_t *ctl, gsl_vector *x, atm_t *atm)
    Decompose parameter vector or state vector.
```

• void x2atm_help (atm_t *atm, double zmin, double zmax, double *value, gsl_vector *x, size_t *n)

Extract elements from state vector.

• void y2obs (ctl_t *ctl, gsl_vector *y, obs_t *obs)

Decompose measurement vector.

5.5.1 Detailed Description

JURASSIC library declarations.

Definition in file jurassic.h.

5.5.2 Function Documentation

```
5.5.2.1 size_t atm2x ( ctl_t * ctl, atm_t * atm, gsl_vector * x, int * iqa, int * ipa )
```

Compose state vector or parameter vector.

Definition at line 29 of file jurassic.c.

```
00034
00035
00036
        int ig, iw;
00037
00038
       size_t n = 0;
00039
00040
       /* Add pressure... */
       atm2x_help(atm, ctl->retp_zmin, ctl->retp_zmax,
00041
00042
                   atm->p, IDXP, x, iqa, ipa, &n);
00043
00044
       /* Add temperature... */
00045
       atm2x_help(atm, ctl->rett_zmin, ctl->rett_zmax,
00046
                  atm->t, IDXT, x, iqa, ipa, &n);
00047
00048
        /* Add volume mixing ratios... */
00049
       for (ig = 0; ig < ctl->ng; ig++)
00050
         atm2x_help(atm, ctl->retq_zmin[ig], ctl->retq_zmax[ig],
00051
                     atm->q[ig], IDXQ(ig), x, iqa, ipa, &n);
00052
00053
        /* Add extinction... */
       for (iw = 0; iw < ctl->nw; iw++)
00055
        atm2x_help(atm, ctl->retk_zmin[iw], ctl->retk_zmax[iw],
00056
                    atm->k[iw], IDXK(iw), x, iqa, ipa, &n);
00057
00058
       return n;
00059 }
```

Here is the call graph for this function:



5.5.2.2 void atm2x_help (atm_t * atm, double zmin, double zmax, double * value, int val_iqa, gsl_vector * x, int * iqa, int * ipa, size_t * n)

Add elements to state vector.

Definition at line 63 of file jurassic.c.

```
00072
                             {
00073
00074
           int ip;
00075
           /* Add elements to state vector... */
for (ip = 0; ip < atm->np; ip++)
   if (atm->z[ip] >= zmin && atm->z[ip] <= zmax) {
00076
00077
00079
                if (x != NULL)
                 gsl_vector_set(x, *n, value[ip]);
if (iqa != NULL)
08000
00081
                 iqa[*n] = val_iqa;
if (ipa != NULL)
00082
00083
00084
                   ipa[*n] = ip;
00085
                 (*n)++;
00086
00087 }
```

5.5.2.3 double brightness (double rad, double nu)

Compute brightness temperature.

Definition at line 91 of file jurassic.c.

```
00093 {
00094
00095 return C2 * nu / gsl_log1p(C1 * POW3(nu) / rad);
00096 }
```

5.5.2.4 void cart2geo (double *x, double *z, double *lon, double *lon)

Convert Cartesian coordinates to geolocation.

Definition at line 101 of file jurassic.c.

```
5.5.2.5 void climatology ( ctl_t * ctl, atm_t * atm_mean )
```

Interpolate climatological data.

Definition at line 117 of file jurassic.c.

```
00119
00120
00121
           static double z[121] = {
             0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55,
00122
00123
00124
              56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91,
00125
              92, 93, 94, 95, 96, 97, 98, 99, 100, 101, 102, 103, 104, 105, 106, 107,
00127
00128
             108, 109, 110, 111, 112, 113, 114, 115, 116, 117, 118, 119, 120
00129
00130
00131
           static double pre[121] = {
             1017, 901.083, 796.45, 702.227, 617.614, 541.644, 473.437, 412.288,
              357.603, 308.96, 265.994, 228.348, 195.619, 167.351, 143.039, 122.198,
              104.369, 89.141, 76.1528, 65.0804, 55.641, 47.591, 40.7233, 34.8637,
00134
              29.8633, 25.5956, 21.9534, 18.8445, 16.1909, 13.9258, 11.9913, 10.34, 8.92988, 7.72454, 6.6924, 5.80701, 5.04654, 4.39238, 3.82902,
00135
00136
              3.34337, 2.92413, 2.56128, 2.2464, 1.97258, 1.73384, 1.52519, 1.34242, 1.18197, 1.04086, 0.916546, 0.806832, 0.709875, 0.624101, 0.548176,
00137
00138
              0.480974,\ 0.421507,\ 0.368904,\ 0.322408,\ 0.281386,\ 0.245249,\ 0.213465
00139
00140
              0.185549,\ 0.161072,\ 0.139644,\ 0.120913,\ 0.104568,\ 0.0903249,\ 0.0779269,
              0.0671493, 0.0577962, 0.0496902, 0.0426736, 0.0366093, 0.0313743, 0.0268598, 0.0229699, 0.0196206, 0.0167399, 0.0142646, 0.0121397,
00141
00142
              0.0103181, 0.00875775, 0.00742226, 0.00628076, 0.00530519, 0.00447183,
00143
              0.00376124, 0.00315632, 0.00264248, 0.00220738, 0.00184003, 0.00153095,
00144
              0.00127204, 0.00105608, 0.000876652, 0.00072798, 0.00060492,
00146
              0.000503201, 0.000419226, 0.000349896, 0.000292659, 0.000245421
00147
              0.000206394,\ 0.000174125,\ 0.000147441,\ 0.000125333,\ 0.000106985,
              9.173e-05, 7.90172e-05, 6.84172e-05, 5.95574e-05, 5.21183e-05, 4.58348e-05, 4.05127e-05, 3.59987e-05, 3.21583e-05, 2.88718e-05, 2.60322e-05, 2.35687e-05, 2.14263e-05, 1.95489e-05
00148
00149
00150
00151
00152
00153
           static double tem[121] = {
             285.14, 279.34, 273.91, 268.3, 263.24, 256.55, 250.2, 242.82, 236.17, 229.87, 225.04, 221.19, 218.85, 217.19, 216.2, 215.68, 215.42, 215.55, 215.92, 216.4, 216.93, 217.45, 218, 218.68, 219.39, 220.25, 221.3, 222.41, 223.88, 225.42, 227.2, 229.52, 231.89, 234.51, 236.85, 239.42,
00154
00155
00156
              241.94, 244.57, 247.36, 250.32, 253.34, 255.82, 258.27, 260.39, 262.03, 263.45, 264.2, 264.78, 264.67, 264.38, 263.24, 262.03, 260.02,
00158
00159
             258.09, 255.63, 253.28, 250.43, 247.81, 245.26, 242.77, 240.38, 237.94, 235.79, 233.53, 231.5, 229.53, 227.6, 225.62, 223.77, 222.06, 220.33, 218.69, 217.18, 215.64, 214.13, 212.52, 210.86, 209.25, 207.49, 205.81, 204.11, 202.22, 200.32, 198.39, 195.92, 193.46, 190.94, 188.31, 185.82, 183.57, 181.43, 179.74, 178.64, 178.1, 178.25
00160
00161
00162
00163
00164
                                                                                               178.1, 178.25,
              178.7, 179.41, 180.67, 182.31, 184.18, 186.6, 189.53, 192.66, 196.54, 201.13, 205.93, 211.73, 217.86, 225, 233.53, 242.57, 252.14, 261.48,
00165
00166
00167
             272.97, 285.26, 299.12, 312.2, 324.17, 338.34, 352.56, 365.28
00168
00169
00170
           static double c2h2[121] = {
            1.352e-09, 2.83e-10, 1.269e-10, 6.926e-11, 4.346e-11, 2.909e-11,
00171
             2.014e-11, 1.363e-11, 8.71e-12, 5.237e-12, 2.718e-12, 1.375e-12, 5.786e-13, 2.16e-13, 7.317e-14, 2.551e-14, 1.055e-14, 4.758e-15, 2.056e-15, 7.703e-16, 2.82e-16, 1.035e-16, 4.382e-17, 1.946e-17,
00172
00173
00174
              9.638e-18, 5.2e-18, 2.811e-18, 1.494e-18, 7.925e-19, 4.213e-19,
00175
              1.998e-19, 8.78e-20, 3.877e-20, 1.728e-20, 7.743e-21, 3.536e-21,
00176
00177
              1.623e-21, 7.508e-22, 3.508e-22, 1.65e-22, 7.837e-23, 3.733e-23,
00178
              1.808e-23, 8.77e-24, 4.285e-24, 2.095e-24, 1.032e-24, 5.082e-25,
00179
              2.506 e-25,\ 1.236 e-25,\ 6.088 e-26,\ 2.996 e-26,\ 1.465 e-26,\ 0,\ 0,\ 0,
              00180
00181
00182
              00183
00184
00185
           static double c2h6[121] = {
             2.667e-09, 2.02e-09, 1.658e-09, 1.404e-09, 1.234e-09, 1.109e-09,
00186
              1.012e-09, 9.262e-10, 8.472e-10, 7.71e-10, 6.932e-10, 6.216e-10, 5.503e-10, 4.87e-10, 4.342e-10, 3.861e-10, 3.347e-10, 2.772e-10,
00187
              2.209e-10, 1.672e-10, 1.197e-10, 8.536e-11, 5.783e-11, 3.846e-11
00189
00190
              2.495e-11, 1.592e-11, 1.017e-11, 6.327e-12, 3.895e-12, 2.403e-12,
00191
              1.416e-12, 8.101e-13, 4.649e-13, 2.686e-13, 1.557e-13, 9.14e-14,
              5.386e-14, 3.19e-14, 1.903e-14, 1.14e-14, 6.875e-15, 4.154e-15, 2.538e-15, 1.553e-15, 9.548e-16, 5.872e-16, 3.63e-16, 2.244e-16, 1.388e-16, 8.587e-17, 5.308e-17, 3.279e-17, 2.017e-17, 1.238e-17,
00192
00193
00194
00195
              7.542e-18, 4.585e-18, 2.776e-18, 1.671e-18, 9.985e-19, 5.937e-19,
```

```
3.518e-19, 2.07e-19, 1.215e-19, 7.06e-20, 4.097e-20, 2.37e-20,
                               1.363e-20, 7.802e-21, 4.441e-21, 2.523e-21, 1.424e-21, 8.015e-22, 4.497e-22, 2.505e-22, 1.391e-22, 7.691e-23, 4.238e-23, 2.331e-23,
00197
00198
00199
                               1.274e-23, 6.929e-24, 3.752e-24, 2.02e-24, 1.083e-24, 5.774e-25,
00200
                              00201
                               0, 0, 0, 0, 0, 0, 0, 0
00203
00204
                        static double ccl4[121] = {
   1.075e-10, 1.
00205
00206
                               1.075e-10, 1.075e-10, 1.075e-10, 1.06e-10, 1.024e-10, 9.69e-11, 8.93e-11, 8.078e-11, 7.213e-11, 6.307e-11, 5.383e-11, 4.49e-11,
00207
00208
                              3.609e-11, 2.705e-11, 1.935e-11, 1.385e-11, 8.35e-12, 5.485e-12, 3.853e-12, 2.22e-12, 5.875e-13, 3.445e-13, 1.015e-13, 6.075e-14,
00209
00210
                               4.383e-14, 2.692e-14, 1e-14, 1
00211
00212
                               le-14, le
00213
00215
                               le-14, le-14, le-14, le-14, le-14, le-14, le-14, le-14, le-14, le-14,
00216
                               le-14, le-14,
00217
                               1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14,
00218
                               1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14,
00219
                              1e-14, 1e
00220
00221
00222
00223
                        static double ch4[121] = {
00224
                              1.864e-06, 1.835e-06, 1.819e-06, 1.805e-06, 1.796e-06, 1.788e-06,
00225
                               1.782e-06, 1.776e-06, 1.769e-06, 1.761e-06, 1.749e-06, 1.734e-06,
                               1.716e-06, 1.692e-06, 1.654e-06, 1.61e-06, 1.567e-06, 1.502e-06,
00226
00227
                               1.433e-06, 1.371e-06, 1.323e-06, 1.277e-06, 1.232e-06, 1.188e-06,
                               1.147e-06, 1.108e-06, 1.07e-06, 1.027e-06, 9.854e-07, 9.416e-07,
00228
00229
                               8.933e-07, 8.478e-07, 7.988e-07, 7.515e-07, 7.07e-07, 6.64e-07,
                              6.239e-07, 5.864e-07, 5.512e-07, 5.184e-07, 4.87e-07, 4.571e-07, 4.296e-07, 4.04e-07, 3.802e-07, 3.578e-07, 3.383e-07, 3.203e-07, 3.032e-07, 2.889e-07, 2.76e-07, 2.635e-07, 2.519e-07, 2.409e-07, 2.302e-07, 2.219e-07, 2.144e-07, 2.071e-07, 1.999e-07, 1.93e-07,
00230
00231
00232
00234
                               1.862e-07, 1.795e-07, 1.731e-07, 1.668e-07, 1.607e-07, 1.548e-07,
                              1.49e-07, 1.434e-07, 1.38e-07, 1.328e-07, 1.277e-07, 1.227e-07, 1.18e-07, 1.134e-07, 1.089e-07, 1.046e-07, 1.004e-07, 9.635e-08,
00235
00236
                               7.159e-08, 8.867e-08, 8.502e-08, 8.15e-08, 7.809e-08, 7.48e-08, 7.159e-08, 6.849e-08, 6.55e-08, 6.262e-08, 5.98e-08, 5.708e-08, 5.448e-08, 5.194e-08, 4.951e-08, 4.72e-08, 4.5e-08, 4.291e-08,
00237
00238
00239
                               4.093e-08, 3.905e-08, 3.729e-08, 3.563e-08, 3.408e-08, 3.265e-08,
00240
00241
                               3.128e-08, 2.996e-08, 2.87e-08, 2.76e-08, 2.657e-08, 2.558e-08,
00242
                               2.467e-08, 2.385e-08, 2.307e-08, 2.234e-08, 2.168e-08, 2.108e-08,
00243
                               2.05e-08, 1.998e-08, 1.947e-08, 1.902e-08, 1.86e-08, 1.819e-08,
00244
                               1.782e-08
00245
00247
                         static double clo[121] = {
00248
                               7.419e-15, 1.061e-14, 1.518e-14, 2.195e-14, 3.175e-14, 4.666e-14,
                              6.872e-14, 1.03e-13, 1.553e-13, 2.375e-13, 3.664e-13, 5.684e-13, 8.915e-13, 1.402e-12, 2.269e-12, 4.125e-12, 7.501e-12, 1.257e-11,
00249
00250
00251
                               2.048e-11, 3.338e-11, 5.44e-11, 8.846e-11, 1.008e-10, 1.082e-10,
                               1.157e-10, 1.232e-10, 1.312e-10, 1.539e-10, 1.822e-10, 2.118e-10,
                               2.387e-10, 2.687e-10, 2.875e-10, 3.031e-10, 3.23e-10, 3.648e-10,
00253
00254
                               4.117e-10, 4.477e-10, 4.633e-10, 4.794e-10, 4.95e-10, 5.104e-10,
00255
                               5.259e-10, 5.062e-10, 4.742e-10, 4.443e-10, 4.051e-10, 3.659e-10,
                               3.305e-10, 2.911e-10, 2.54e-10, 2.215e-10, 1.927e-10, 1.675e-10,
00256
                               1.452e-10, 1.259e-10, 1.09e-10, 9.416e-11, 8.119e-11, 6.991e-11,
00257
                               6.015e-11, 5.163e-11, 4.43e-11, 3.789e-11, 3.24e-11, 2.769e-11,
                               2.361e-11, 2.011e-11, 1.71e-11, 1.453e-11, 1.233e-11, 1.045e-11,
00259
00260
                               8.851e-12, 7.48e-12, 6.316e-12, 5.326e-12, 4.487e-12, 3.778e-12,
00261
                              3.176e-12, 2.665e-12, 2.234e-12, 1.87e-12, 1.563e-12, 1.304e-12,
                              1.085e-12, 9.007e-13, 7.468e-13, 6.179e-13, 5.092e-13, 4.188e-13, 3.442e-13, 2.816e-13, 2.304e-13, 1.885e-13, 1.542e-13, 1.263e-13,
00262
00263
00264
                               1.035e-13, 8.5e-14, 7.004e-14, 5.783e-14, 4.795e-14, 4.007e-14,
00265
                               3.345e-14, 2.792e-14, 2.33e-14, 1.978e-14, 1.686e-14, 1.438e-14,
                               1.234e-14, 1.07e-14, 9.312e-15, 8.131e-15, 7.164e-15, 6.367e-15,
00266
00267
                               5.67e-15, 5.088e-15, 4.565e-15, 4.138e-15, 3.769e-15, 3.432e-15,
00268
                              3.148e-15
00269
00270
00271
                         static double clono2[121] = {
                              1.011e-13, 1.515e-13, 2.272e-13, 3.446e-13, 5.231e-13, 8.085e-13,
00272
00273
                               1.253e-12, 1.979e-12, 3.149e-12, 5.092e-12, 8.312e-12, 1.366e-11,
                              2.272e-11, 3.791e-11, 6.209e-11, 9.101e-11, 1.334e-10, 1.951e-10, 2.853e-10, 3.94e-10, 4.771e-10, 5.771e-10, 6.675e-10, 7.665e-10,
00274
00275
                               8.504e-10, 8.924e-10, 9.363e-10, 8.923e-10, 8.411e-10, 7.646e-10, 6.525e-10, 5.576e-10, 4.398e-10, 3.403e-10, 2.612e-10, 1.915e-10,
00276
                               1.407e-10, 1.028e-10, 7.455e-11, 5.42e-11, 3.708e-11, 2.438e-11,
00278
00279
                               1.618e-11, 1.075e-11, 7.17e-12, 4.784e-12, 3.205e-12, 2.147e-12,
00280
                               1.44 e^{-12}, \ 9.654 e^{-13}, \ 6.469 e^{-13}, \ 4.332 e^{-13}, \ 2.891 e^{-13}, \ 1.926 e^{-13},
                               1.274e-13, 8.422e-14, 5.547e-14, 3.636e-14, 2.368e-14, 1.536e-14, 9.937e-15, 6.39e-15, 4.101e-15, 2.61e-15, 1.659e-15, 1.052e-15,
00281
00282
```

```
6.638e-16, 4.172e-16, 2.61e-16, 1.63e-16, 1.013e-16, 6.275e-17,
                     3.879e-17, 2.383e-17, 1.461e-17, 8.918e-18, 5.43e-18, 3.301e-18, 1.997e-18, 1.203e-18, 7.216e-19, 4.311e-19, 2.564e-19, 1.519e-19,
00284
00285
00286
                     8.911e-20, 5.203e-20, 3.026e-20, 1.748e-20, 9.99e-21, 5.673e-21,
                     3.215e-21, 1.799e-21, 1.006e-21, 5.628e-22, 3.146e-22, 1.766e-22, 9.94e-23, 5.614e-23, 3.206e-23, 1.841e-23, 1.071e-23, 6.366e-24,
00287
00288
                     3.776e-24, 2.238e-24, 1.326e-24, 8.253e-25, 5.201e-25, 3.279e-25,
                     2.108e-25, 1.395e-25, 9.326e-26, 6.299e-26, 4.365e-26, 3.104e-26,
00290
00291
                     2.219e-26, 1.621e-26, 1.185e-26, 8.92e-27, 6.804e-27, 5.191e-27,
00292
                     4.041e-27
00293
00294
00295
                static double co[121] = {
                    1.907e-07, 1.553e-07, 1.362e-07, 1.216e-07, 1.114e-07, 1.036e-07,
00296
00297
                     9.737e-08, 9.152e-08, 8.559e-08, 7.966e-08, 7.277e-08, 6.615e-08,
00298
                     5.884e-08, 5.22e-08, 4.699e-08, 4.284e-08, 3.776e-08, 3.274e-08,
                    2.845e-08, 2.479e-08, 2.246e-08, 2.054e-08, 1.991e-08, 1.951e-08, 1.94e-08, 2.009e-08, 2.1e-08, 2.201e-08, 2.322e-08, 2.45e-08, 2.602e-08, 2.73e-08, 2.867e-08, 2.998e-08, 3.135e-08, 3.255e-08,
00299
00300
00302
                     3.352e-08, 3.426e-08, 3.484e-08, 3.53e-08, 3.593e-08, 3.671e-08,
                      3.759e-08, 3.945e-08, 4.192e-08, 4.49e-08, 5.03e-08, 5.703e-08,
00303
00304
                      6.538e-08, 7.878e-08, 9.644e-08, 1.196e-07, 1.498e-07, 1.904e-07,
                     2.422e-07, 3.055e-07, 3.804e-07, 4.747e-07, 5.899e-07, 7.272e-07, 8.91e-07, 1.071e-06, 1.296e-06, 1.546e-06, 1.823e-06, 2.135e-06, 2.44e-06, 2.714e-06, 2.967e-06, 3.189e-06, 3.391e-06, 3.58e-06,
00305
00306
00307
                      3.773e-06, 4.022e-06, 4.346e-06, 4.749e-06, 5.199e-06, 5.668e-06,
00309
                      6.157e-06, 6.688e-06, 7.254e-06, 7.867e-06, 8.539e-06, 9.26e-06,
00310
                     1.009e-05, 1.119e-05, 1.228e-05, 1.365e-05, 1.506e-05, 1.641e-05,
00311
                     1.784e-05, 1.952e-05, 2.132e-05, 2.323e-05, 2.531e-05, 2.754e-05,
00312
                     3.047e-05, 3.459e-05, 3.922e-05, 4.439e-05, 4.825e-05, 5.077e-05,
                     5.34e-05, 5.618e-05, 5.909e-05, 6.207e-05, 6.519e-05, 6.845e-05,
00313
                     6.819e-05, 6.726e-05, 6.622e-05, 6.512e-05, 6.671e-05, 6.862e-05, 7.048e-05, 7.264e-05, 7.3e-05, 7.2e-05, 7.2e-
00314
00315
00316
00317
00318
                static double cof2[121] = {
                     7.5e-14, 1.055e-13, 1.485e-13, 2.111e-13, 3.001e-13, 4.333e-13, 6.269e-13, 9.221e-13, 1.364e-12, 2.046e-12, 3.093e-12, 4.703e-12,
00319
00321
                      7.225e-12, 1.113e-11, 1.66e-11, 2.088e-11, 2.626e-11, 3.433e-11,
                      4.549e-11, 5.886e-11, 7.21e-11, 8.824e-11, 1.015e-10, 1.155e-10,
00322
00323
                     1.288e-10, 1.388e-10, 1.497e-10, 1.554e-10, 1.606e-10, 1.639e-10,
                     1.64e-10, 1.64e-10, 1.596e-10, 1.542e-10, 1.482e-10, 1.382e-10,
00324
                     1.289e-10, 1.198e-10, 1.109e-10, 1.026e-10, 9.484e-11, 8.75e-11, 8.086e-11, 7.49e-11, 6.948e-11, 6.446e-11, 5.961e-11, 5.505e-11,
00325
00326
                     5.085e-11, 4.586e-11, 4.1e-11, 3.665e-11, 3.235e-11, 2.842e-11,
00328
                     2.491e-11, 2.11e-11, 1.769e-11, 1.479e-11, 1.197e-11, 9.631e-12,
                    7.74e-12, 6.201e-12, 4.963e-12, 3.956e-12, 3.151e-12, 2.507e-12, 1.99e-12, 1.576e-12, 1.245e-12, 9.83e-13, 7.742e-13, 6.088e-13,
00329
00330
                     4.782e-13, 3.745e-13, 2.929e-13, 2.286e-13, 1.782e-13, 1.388e-13,
00331
                     1.079e-13, 8.362e-14, 6.471e-14, 4.996e-14, 3.85e-14, 2.96e-14,
00332
                     2.265e-14, 1.729e-14, 1.317e-14, 9.998e-15, 7.549e-15, 5.683e-15,
                      4.273e-15, 3.193e-15, 2.385e-15, 1.782e-15, 1.331e-15, 9.957e-16,
00334
00335
                     7.461e-16, 5.601e-16, 4.228e-16, 3.201e-16, 2.438e-16, 1.878e-16,
                     1.445e-16, 1.111e-16, 8.544e-17, 6.734e-17, 5.341e-17, 4.237e-17, 3.394e-17, 2.759e-17, 2.254e-17, 1.851e-17, 1.54e-17, 1.297e-17, 1.096e-17, 9.365e-18, 8e-18, 6.938e-18, 6.056e-18, 5.287e-18,
00336
00337
00338
                      4.662e-18
00340
00341
00342
                 static double f11[121] = {
                    2.65e-10, 2.65e-
00343
00344
00345
                     2.44e-10, 2.348e-10, 2.258e-10, 2.153e-10, 2.046e-10, 1.929e-10,
                     1.782e-10, 1.648e-10, 1.463e-10, 1.291e-10, 1.1e-10, 8.874e-11,
00346
00347
                     7.165e-11, 5.201e-11, 3.744e-11, 2.577e-11, 1.64e-11, 1.048e-11,
00348
                    5.993e-12, 3.345e-12, 1.839e-12, 9.264e-13, 4.688e-13, 2.329e-13,
00349
                     1.129e-13, 5.505e-14, 2.825e-14, 1.492e-14, 7.997e-15, 5.384e-15,
00350
                     3.988e-15, 2.955e-15, 2.196e-15, 1.632e-15, 1.214e-15, 9.025e-16,
00351
                     6.708e-16, 4.984e-16, 3.693e-16, 2.733e-16, 2.013e-16, 1.481e-16,
                     1.087e-16, 7.945e-17, 5.782e-17, 4.195e-17, 3.038e-17, 2.19e-17,
00353
                      1.577e-17, 1.128e-17, 8.063e-18, 5.753e-18, 4.09e-18, 2.899e-18,
00354
                     2.048e-18, 1.444e-18, 1.015e-18, 7.12e-19, 4.985e-19, 3.474e-19,
00355
                     2.417e-19, 1.677e-19, 1.161e-19, 8.029e-20, 5.533e-20, 3.799e-20,
                     2.602e-20, 1.776e-20, 1.209e-20, 8.202e-21, 5.522e-21, 3.707e-21, 2.48e-21, 1.652e-21, 1.091e-21, 7.174e-22, 4.709e-22, 3.063e-22, 1.991e-22, 1.294e-22, 8.412e-23, 5.483e-23, 3.581e-23, 2.345e-23,
00356
00357
00358
00359
                     1.548e-23, 1.027e-23, 6.869e-24, 4.673e-24, 3.173e-24, 2.153e-24,
00360
                      1.461e-24, 1.028e-24, 7.302e-25, 5.188e-25, 3.739e-25, 2.753e-25,
00361
                     2.043e-25, 1.528e-25, 1.164e-25, 9.041e-26, 7.051e-26, 5.587e-26,
00362
                     4.428e-26. 3.588e-26. 2.936e-26. 2.402e-26. 1.995e-26
00363
00364
00365
                 static double f12[121] =
00366
                     5.45e-10, 5.45e-10, 5.45e-10, 5.45e-10, 5.45e-10, 5.45e-10, 5.45e-10,
                     5.45e-10, 5.45e-10, 5.45e-10, 5.45e-10, 5.45e-10, 5.45e-10, 5.429e-10, 5.291e-10, 5.155e-10, 5.022e-10, 4.893e-10, 4.772e-10, 4.655e-10, 4.497e-10, 4.249e-10, 4.015e-10, 3.632e-10, 3.261e-10, 2.858e-10, 2.408e-10,
00367
00368
00369
```

```
2.03e-10, 1.685e-10, 1.4e-10, 1.163e-10, 9.65e-11, 8.02e-11, 6.705e-11,
                5.624e-11, 4.764e-11, 4.249e-11, 3.792e-11, 3.315e-11, 2.819e-11,
00371
00372
                2.4e-11, 1.999e-11, 1.64e-11, 1.352e-11, 1.14e-11, 9.714e-12,
00373
                8.28e-12, 7.176e-12, 6.251e-12, 5.446e-12, 4.72e-12, 4.081e-12,
                3.528e-12, 3.08e-12, 2.699e-12, 2.359e-12, 2.111e-12, 1.901e-12, 1.709e-12, 1.534e-12, 1.376e-12, 1.233e-12, 1.103e-12, 9.869e-13,
00374
00375
                8.808e-13, 7.859e-13, 7.008e-13, 6.241e-13, 5.553e-13, 4.935e-13,
                4.383e-13, 3.889e-13, 3.447e-13, 3.054e-13, 2.702e-13, 2.389e-13,
00377
00378
                2.11e-13, 1.862e-13, 1.643e-13, 1.448e-13, 1.274e-13, 1.121e-13,
                9.844e-14, 8.638e-14, 7.572e-14, 6.62e-14, 5.782e-14, 5.045e-14, 4.394e-14, 3.817e-14, 3.311e-14, 2.87e-14, 2.48e-14, 2.142e-14,
00379
00380
                1.851e-14, 1.599e-14, 1.383e-14, 1.196e-14, 1.036e-14, 9e-15, 7.828e-15, 6.829e-15, 5.992e-15, 5.254e-15, 4.606e-15, 4.037e-15,
00381
00382
                3.5835-15, 3.19e-15, 2.841e-15, 2.542e-15, 2.291e-15, 2.07e-15, 1.875e-15, 1.71e-15, 1.57e-15, 1.442e-15, 1.333e-15, 1.232e-15,
00383
00384
00385
                1.147e-15, 1.071e-15, 1.001e-15, 9.396e-16
00386
00387
            static double f14[121] = {
                9e-11, 9e-11, 9e-11, 9e-11, 9e-11, 9e-11, 9e-11, 9e-11, 9e-11, 9e-11,
00389
                9e-11, 9e-11, 9e-11, 9e-11, 8.91e-11, 8.73e-11, 8.46e-11
00390
00391
                8.19e-11, 7.92e-11, 7.74e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11,
                                                                                                  7.65e-11,
                7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11,
                                                                                                                   7.65e-11,
00392
00393
                7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11,
00394
                7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11,
                7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11,
00395
00396
                7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11,
                                                                                                                    7.65e-11,
                                                                                                                   7,65e-11,
00397
                7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11,
00398
                7.65e-11, 7.65e-11, 7.65e-11,
                                                                  7.65e-11, 7.65e-11, 7.65e-11,
                                                                                                                    7.65e-11.
                7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11,
00399
                                                                                                                    7.65e-11.
00400
                7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11,
00401
                 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11,
                7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11,
00402
00403
                7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11,
                7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e
00404
00405
00406
00408
            static double f22[121] =
              1.4e-10, 1.4e-10, 1.4e-10, 1.4e-10, 1.4e-10, 1.4e-10, 1.4e-10,
00409
00410
                1.4e-10, 1.4e-10, 1.4e-10, 1.372e-10, 1.317e-10, 1.235e-10, 1.153e-10,
                1.075e-10, 1.002e-10, 9.332e-11, 8.738e-11, 8.194e-11, 7.7e-11,
00411
                7.165e-11, 6.753e-11, 6.341e-11, 5.971e-11, 5.6e-11, 5.229e-11, 4.859e-11, 4.488e-11, 4.118e-11, 3.83e-11, 3.568e-11, 3.308e-11,
00412
00413
                3.047e-11, 2.82e-11, 2.594e-11, 2.409e-11, 2.237e-11, 2.065e-11,
00414
00415
                1.894e-11, 1.771e-11, 1.647e-11, 1.532e-11, 1.416e-11, 1.332e-11,
00416
                1.246e-11, 1.161e-11, 1.087e-11, 1.017e-11, 9.471e-12, 8.853e-12,
               8.235e-12, 7.741e-12, 7.247e-12, 6.836e-12, 6.506e-12, 6.176e-12, 5.913e-12, 5.65e-12, 5.419e-12, 5.221e-12, 5.024e-12, 4.859e-12, 4.694e-12, 4.546e-12, 4.414e-12, 4.282e-12, 4.15e-12, 4.019e-12, 3.903e-12, 3.805e-12, 3.706e-12, 3.607e-12, 3.508e-12, 3.41e-12,
00417
00418
00419
                3.31e-12, 3.212e-12, 3.129e-12, 3.047e-12, 2.964e-12, 2.882e-12,
00421
00422
                2.8e-12, 2.734e-12, 2.668e-12, 2.602e-12, 2.537e-12, 2.471e-12,
                2.421e-12, 2.372e-12, 2.322e-12, 2.273e-12, 2.224e-12, 2.182e-12, 2.141e-12, 2.1e-12, 2.059e-12, 2.018e-12, 1.977e-12, 1.935e-12,
00423
00424
                1.894e-12, 1.853e-12, 1.812e-12, 1.77e-12, 1.73e-12,
                                                                                                       1.688e-12.
00425
                 1.647e-12, 1.606e-12, 1.565e-12, 1.524e-12, 1.483e-12, 1.441e-12,
                1.4e-12, 1.359e-12, 1.317e-12, 1.276e-12, 1.235e-12, 1.194e-12,
00427
00428
                1.153e-12, 1.112e-12, 1.071e-12, 1.029e-12, 9.883e-13
00429
00430
00431
            static double h2o[121] = {
00432
                0.01166, 0.008269, 0.005742, 0.003845, 0.00277, 0.001897, 0.001272,
                0.000827, 0.000539, 0.0003469, 0.0001579, 3.134e-05, 1.341e-05,
00433
00434
                6.764e-06, 4.498e-06, 3.703e-06, 3.724e-06, 3.899e-06, 4.002e-06,
00435
                4.122e-06, 4.277e-06, 4.438e-06, 4.558e-06, 4.673e-06, 4.763e-06,
00436
                4.809e-06, 4.856e-06, 4.936e-06, 5.021e-06, 5.114e-06, 5.222e-06,
00437
                5.331e-06, 5.414e-06, 5.488e-06, 5.563e-06, 5.633e-06, 5.704e-06,
00438
                5.767e-06, 5.819e-06, 5.872e-06, 5.914e-06, 5.949e-06, 5.984e-06,
                6.015e-06, 6.044e-06, 6.073e-06, 6.104e-06, 6.136e-06, 6.167e-06,
00440
                6.189e-06, 6.208e-06, 6.226e-06, 6.212e-06, 6.185e-06, 6.158e-06,
00441
                6.114e-06, 6.066e-06, 6.018e-06, 5.877e-06, 5.728e-06, 5.582e-06,
00442
                5.437e-06, 5.296e-06, 5.156e-06, 5.02e-06, 4.886e-06, 4.754e-06,
                4.625e-06, 4.498e-06, 4.374e-06, 4.242e-06, 4.096e-06, 3.955e-06, 3.817e-06, 3.683e-06, 3.491e-06, 3.204e-06, 2.94e-06, 2.696e-06, 2.47e-06, 2.252e-06, 2.019e-06, 1.808e-06, 1.618e-06, 1.445e-06,
00443
00444
00445
                1.285e-06, 1.105e-06, 9.489e-07, 8.121e-07, 6.938e-07, 5.924e-07,
00446
                5.04e-07, 4.288e-07, 3.648e-07, 3.103e-07, 2.642e-07, 2.252e-07, 1.921e-07, 1.643e-07, 1.408e-07, 1.211e-07, 1.048e-07, 9.063e-08,
00447
00448
                7.835e-08, 6.774e-08, 5.936e-08, 5.221e-08, 4.592e-08, 4.061e-08, 3.62e-08, 3.236e-08, 2.902e-08, 2.62e-08, 2.383e-08, 2.171e-08, 1.989e-08, 1.823e-08, 1.684e-08, 1.562e-08, 1.449e-08, 1.351e-08
00449
00450
00451
00452
00453
00454
            static double h2o2[121] = {
               1.779e-10, 7.938e-10, 8.953e-10, 8.032e-10, 6.564e-10, 5.159e-10, 4.003e-10, 3.026e-10, 2.222e-10, 1.58e-10, 1.044e-10, 6.605e-11,
00455
00456
```

```
3.413e-11, 1.453e-11, 1.062e-11, 1.009e-11, 9.597e-12, 1.175e-11,
            1.572e-11, 2.091e-11, 2.746e-11, 3.603e-11, 4.791e-11, 6.387e-11, 8.239e-11, 1.007e-10, 1.23e-10, 1.363e-10, 1.489e-10, 1.585e-10,
00458
00459
00460
            1.608e-10, 1.632e-10, 1.576e-10, 1.502e-10, 1.423e-10, 1.302e-10,
00461
            1.192e-10, 1.085e-10, 9.795e-11, 8.854e-11, 8.057e-11, 7.36e-11, 6.736e-11, 6.362e-11, 6.087e-11, 5.825e-11, 5.623e-11, 5.443e-11,
00462
            5.27e-11, 5.098e-11, 4.931e-11, 4.769e-11, 4.611e-11, 4.458e-11,
            4.308e-11, 4.102e-11, 3.887e-11, 3.682e-11, 3.521e-11, 3.369e-11,
00464
00465
            3.224e-11, 3.082e-11, 2.946e-11, 2.814e-11, 2.687e-11, 2.566e-11,
00466
            2.449e-11, 2.336e-11, 2.227e-11, 2.123e-11, 2.023e-11, 1.927e-11,
            1.835 e^{-11},\ 1.746 e^{-11},\ 1.661 e^{-11},\ 1.58 e^{-11},\ 1.502 e^{-11},\ 1.428 e^{-11},
00467
            1.357e-11, 1.289e-11, 1.224e-11, 1.161e-11, 1.102e-11, 1.045e-11, 9.895e-12, 9.369e-12, 8.866e-12, 8.386e-12, 7.922e-12, 7.479e-12,
00468
00469
            7.06e-12, 6.656e-12, 6.274e-12, 5.914e-12, 5.575e-12, 5.257e-12,
00470
00471
            4.959e-12, 4.679e-12, 4.42e-12, 4.178e-12, 3.954e-12, 3.75e-12,
            3.557e-12, 3.372e-12, 3.198e-12, 3.047e-12, 2.908e-12, 2.775e-12, 2.653e-12, 2.544e-12, 2.442e-12, 2.346e-12, 2.26e-12, 2.183e-12,
00472
00473
00474
            2.11e-12, 2.044e-12, 1.98e-12, 1.924e-12, 1.871e-12, 1.821e-12,
00476
00477
00478
         static double hcn[121] = {
           5.5e-10, 5.498e-10, 5.495e-10, 5.498e-10, 5.498e-10, 5.498e-10, 3.174e-10, 2.4e-10,
00479
00480
00481
            1.626e-10, 1.619e-10, 1.612e-10, 1.602e-10, 1.593e-10, 1.582e-10
            1.572e-10, 1.56e-10, 1.549e-10, 1.539e-10, 1.53e-10, 1.519e-10,
00483
00484
            1.506e-10, 1.487e-10, 1.467e-10, 1.449e-10, 1.43e-10, 1.413e-10,
00485
            1.397e-10, 1.382e-10, 1.368e-10, 1.354e-10, 1.337e-10, 1.315e-10,
00486
            1.292e-10, 1.267e-10, 1.241e-10, 1.215e-10, 1.19e-10, 1.165e-10,
            1.141e-10, 1.118e-10, 1.096e-10, 1.072e-10, 1.047e-10, 1.021e-10,
00487
00488
            9.968e-11, 9.739e-11, 9.539e-11, 9.339e-11, 9.135e-11, 8.898e-11,
            8.664e-11, 8.439e-11, 8.249e-11, 8.075e-11, 7.904e-11, 7.735e-11,
00489
00490
            7.565e-11, 7.399e-11, 7.245e-11, 7.109e-11, 6.982e-11, 6.863e-11,
00491
            6.755e-11, 6.657e-11, 6.587e-11, 6.527e-11, 6.476e-11, 6.428e-11,
            6.382e-11, 6.343e-11, 6.307e-11, 6.272e-11, 6.238e-11, 6.205e-11,
00492
            6.17e-11, 6.137e-11, 6.102e-11, 6.072e-11, 6.046e-11, 6.03e-11, 6.018e-11, 6.01e-11, 6.001e-11, 5.992e-11, 5.984e-11, 5.975e-11,
00493
00495
            5.967e-11, 5.958e-11, 5.95e-11, 5.941e-11, 5.933e-11, 5.925e-11,
00496
            5.916e-11, 5.908e-11, 5.899e-11, 5.891e-11, 5.883e-11, 5.874e-11,
           5.866e-11, 5.858e-11, 5.85e-11, 5.841e-11, 5.833e-11, 5.825e-11, 5.817e-11, 5.808e-11, 5.8e-11, 5.792e-11, 5.784e-11
00497
00498
00499
00500
00501
         static double hno3[121] =
00502
            1.809e-10, 7.234e-10, 5.899e-10, 4.342e-10, 3.277e-10, 2.661e-10,
00503
            2.35e-10, 2.267e-10, 2.389e-10, 2.651e-10, 3.255e-10, 4.099e-10,
            5.42e-10, 6.978e-10, 8.807e-10, 1.112e-09, 1.405e-09, 2.04e-09, 3.111e-09, 4.5e-09, 5.762e-09, 7.37e-09, 7.852e-09, 8.109e-09,
00504
00505
            8.067e-09, 7.554e-09, 7.076e-09, 6.268e-09, 5.524e-09, 4.749e-09,
00506
            3.909e-09, 3.223e-09, 2.517e-09, 1.942e-09, 1.493e-09, 1.122e-09,
            8.449e-10, 6.361e-10, 4.787e-10, 3.611e-10, 2.804e-10, 2.215e-10,
00508
00509
            1.758e-10, 1.441e-10, 1.197e-10, 9.953e-11, 8.505e-11, 7.334e-11,
           6.325e-11, 5.625e-11, 5.058e-11, 4.548e-11, 4.122e-11, 3.748e-11, 3.402e-11, 3.088e-11, 2.8e-11, 2.536e-11, 2.293e-11, 2.072e-11,
00510
00511
            1.871e-11, 1.687e-11, 1.52e-11, 1.368e-11, 1.23e-11, 1.105e-11,
00512
            9.922e-12, 8.898e-12, 7.972e-12, 7.139e-12, 6.385e-12, 5.708e-12,
            5.099e-12, 4.549e-12, 4.056e-12, 3.613e-12, 3.216e-12, 2.862e-12,
00514
00515
            2.544e-12, 2.259e-12, 2.004e-12, 1.776e-12, 1.572e-12, 1.391e-12,
00516
            1.227e-12, 1.082e-12, 9.528e-13, 8.379e-13, 7.349e-13, 6.436e-13,
            5.634e-13, 4.917e-13, 4.291e-13, 3.745e-13, 3.267e-13, 2.854e-13,
00517
00518
            2.494e-13, 2.181e-13, 1.913e-13, 1.68e-13, 1.479e-13, 1.31e-13,
            1.159e-13, 1.025e-13, 9.067e-14, 8.113e-14, 7.281e-14, 6.535e-14,
            5.892e-14, 5.348e-14, 4.867e-14, 4.439e-14, 4.073e-14, 3.76e-14,
00520
00521
            3.476e-14, 3.229e-14, 3e-14, 2.807e-14, 2.635e-14, 2.473e-14,
00522
           2.332e-14
00523
00524
00525
         static double hno4[121] = {
           6.118e-12, 3.594e-12, 2.807e-12, 3.04e-12, 4.458e-12, 7.986e-12,
            1.509e-11, 2.661e-11, 3.738e-11, 4.652e-11, 4.429e-11, 3.992e-11,
00527
00528
            3.347e-11, 3.005e-11, 3.173e-11, 4.055e-11, 5.812e-11, 8.489e-11,
           1.19e-10, 1.482e-10, 1.766e-10, 2.103e-10, 2.35e-10, 2.598e-10, 2.801e-10, 2.899e-10, 3e-10, 2.817e-10, 2.617e-10, 2.332e-10,
00529
00530
            1.933e-10, 1.605e-10, 1.232e-10, 9.285e-11, 6.941e-11, 4.951e-11, 3.539e-11, 2.402e-11, 1.522e-11, 9.676e-12, 6.056e-12, 3.745e-12, 2.34e-12, 1.463e-12, 9.186e-13, 5.769e-13, 3.322e-13, 1.853e-13,
00531
00533
00534
            1.035e-13, 7.173e-14, 5.382e-14, 4.036e-14, 3.401e-14, 2.997e-14,
            2.635e-14, 2.316e-14, 2.034e-14, 1.783e-14, 1.56e-14, 1.363e-14, 1.19e-14, 1.037e-14, 9.032e-15, 7.846e-15, 6.813e-15, 5.912e-15,
00535
00536
            5.121e-15, 4.431e-15, 3.829e-15, 3.306e-15, 2.851e-15, 2.456e-15,
00537
            2.114e-15, 1.816e-15, 1.559e-15, 1.337e-15, 1.146e-15, 9.811e-16,
            8.389e-16, 7.162e-16, 6.109e-16, 5.203e-16, 4.425e-16, 3.76e-16,
00539
00540
            3.184e-16, 2.692e-16, 2.274e-16, 1.917e-16, 1.61e-16, 1.35e-16,
            1.131e-16, 9.437e-17, 7.874e-17, 6.57e-17, 5.481e-17, 4.579e-17, 3.828e-17, 3.204e-17, 2.691e-17, 2.264e-17, 1.912e-17, 1.626e-17, 1.382e-17, 1.174e-17, 9.972e-18, 8.603e-18, 7.45e-18, 6.453e-18,
00541
00542
00543
```

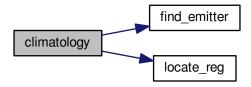
```
5.623e-18, 4.944e-18, 4.361e-18, 3.859e-18, 3.443e-18, 3.096e-18,
                                2.788e-18, 2.528e-18, 2.293e-18, 2.099e-18, 1.929e-18, 1.773e-18,
00545
00546
                               1.64e-18
00547
                         };
00548
00549
                         static double hocl[121] = {
                             1.056e-12, 1.194e-12, 1.35e-12, 1.531e-12, 1.737e-12, 1.982e-12,
                                2.263e-12, 2.599e-12, 2.991e-12, 3.459e-12, 4.012e-12, 4.662e-12,
00551
00552
                                5.438e-12, 6.35e-12, 7.425e-12, 8.686e-12, 1.016e-11, 1.188e-11,
                               1.389e-11, 1.659e-11, 2.087e-11, 2.621e-11, 3.265e-11, 4.064e-11, 4.859e-11, 5.441e-11, 6.09e-11, 6.373e-11, 6.611e-11, 6.94e-11, 7.44e-11, 7.97e-11, 8.775e-11, 9.722e-11, 1.064e-10, 1.089e-10,
00553
00554
00555
                                1.114e-10, 1.106e-10, 1.053e-10, 1.004e-10, 9.006e-11, 7.778e-11,
                                6.739e-11, 5.636e-11, 4.655e-11, 3.845e-11, 3.042e-11, 2.368e-11,
00557
00558
                                1.845e-11, 1.442e-11, 1.127e-11, 8.814e-12, 6.544e-12, 4.763e-12,
00559
                                3.449e-12,\ 2.612e-12,\ 1.999e-12,\ 1.526e-12,\ 1.16e-12,\ 8.793e-13,
00560
                                6.655e-13, 5.017e-13, 3.778e-13, 2.829e-13, 2.117e-13, 1.582e-13,
                                1.178e-13, 8.755e-14, 6.486e-14, 4.799e-14, 3.54e-14, 2.606e-14,
00561
                                1.916e-14, 1.403e-14, 1.026e-14, 7.48e-15, 5.446e-15, 3.961e-15,
00563
                                2.872e-15, 2.076e-15, 1.498e-15, 1.077e-15, 7.726e-16, 5.528e-16,
00564
                                3.929e-16, 2.785e-16, 1.969e-16, 1.386e-16, 9.69e-17, 6.747e-17,
00565
                                4.692e-17, 3.236e-17, 2.232e-17, 1.539e-17, 1.061e-17, 7.332e-18,
                               5.076e-18, 3.522e-18, 2.461e-18, 1.726e-18, 1.22e-18, 8.75e-19,
00566
                                6.264e-19, 4.482e-19, 3.207e-19, 2.368e-19, 1.762e-19, 1.312e-19, 9.891e-20, 7.595e-20, 5.87e-20, 4.567e-20, 3.612e-20, 2.904e-20, 2.343e-20, 1.917e-20, 1.568e-20, 1.308e-20, 1.1e-20, 9.25e-21,
00567
00568
00569
00570
                                7.881e-21
00571
00572
                         static double n2o[121] = {
00573
                             3.17e-07, 3.03e-07,
00574
                                2.984e-07, 2.938e-07, 2.892e-07, 2.847e-07, 2.779e-07, 2.705e-07,
00576
00577
                                2.631e-07, 2.557e-07, 2.484e-07, 2.345e-07, 2.201e-07, 2.01e-07,
00578
                                1.754e-07, 1.532e-07, 1.329e-07, 1.154e-07, 1.003e-07, 8.735e-08,
                               7.617e-08, 6.512e-08, 5.547e-08, 4.709e-08, 3.915e-08, 3.259e-08, 2.738e-08, 2.327e-08, 1.98e-08, 1.711e-08, 1.493e-08, 1.306e-08, 1.165e-08, 1.049e-08, 9.439e-09, 8.375e-09, 7.391e-09, 6.525e-09,
00579
00580
00582
                                5.759e-09, 5.083e-09, 4.485e-09, 3.953e-09, 3.601e-09, 3.27e-09,
                               2.975e-09, 2.757e-09, 2.556e-09, 2.37e-09, 2.195e-09, 2.032e-09, 1.912e-09, 1.79e-09, 1.679e-09, 1.572e-09, 1.482e-09, 1.402e-09,
00583
00584
                                1.326e-09, 1.254e-09, 1.187e-09, 1.127e-09, 1.071e-09, 1.02e-09,
00585
                                9.673e-10, 9.193e-10, 8.752e-10, 8.379e-10, 8.017e-10, 7.66e-10,
00586
                                7.319e-10, 7.004e-10, 6.721e-10, 6.459e-10, 6.199e-10, 5.942e-10,
                                5.703e-10, 5.488e-10, 5.283e-10, 5.082e-10, 4.877e-10, 4.696e-10, 4.52e-10, 4.355e-10, 4.198e-10, 4.039e-10, 3.888e-10, 3.754e-10,
00589
00590
                                3.624e-10, 3.499e-10, 3.381e-10, 3.267e-10, 3.163e-10, 3.058e-10,
00591
                                 2.959 e^{-10}, \ 2.864 e^{-10}, \ 2.77 e^{-10}, \ 2.686 e^{-10}, \ 2.604 e^{-10}, \ 2.534 e^{-10}, \\
                               2.462e-10, 2.386e-10, 2.318e-10, 2.247e-10, 2.189e-10, 2.338e-10, 2.071e-10, 2.189e-10, 1.955e-10, 1.908e-10, 1.816e-10, 1.817e-10
00592
00593
00595
00596
                         static double n2o5[121] = {
                           1.231e-11, 3.035e-12, 1.702e-12, 9.877e-13, 8.081e-13, 9.039e-13, 1.169e-12, 1.474e-12, 1.651e-12, 1.795e-12, 1.998e-12, 2.543e-12, 4.398e-12, 7.698e-12, 1.28e-11, 2.131e-11, 3.548e-11, 5.894e-11,
00597
00598
00599
                                7.645e-11, 1.089e-10, 1.391e-10, 1.886e-10, 2.386e-10, 2.986e-10,
                                3.487e-10, 3.994e-10, 4.5e-10, 4.6e-10, 4.591e-10, 4.1e-10, 3.488e-10,
00601
                               2.846e-10, 2.287e-10, 1.696e-10, 1.011e-10, 6.428e-11, 4.324e-11, 2.225e-11, 6.214e-12, 3.608e-12, 8.793e-13, 4.491e-13, 1.04e-13,
00602
00603
                               6.1e-14, 3.436e-14, 6.671e-15, 1.171e-15, 5.848e-16, 1.212e-16, 1e-16, 1
00604
00605
00606
00607
                                le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16,
00608
                                1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16,
00609
                                le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16,
00610
                                le-16, le
00611
00612
                                1e-16, 1e-16
00614
00615
                         static double nh3[121] = {
00616
                               1e-10, 1e-10, 1e-10, 1e-10, 1e-10, 1e-10, 1e-10, 1e-10, 1e-10, 1e-10,
                                le-10, le-10, le-10, le-10, 9.444e-11, 8.488e-11, 7.241e-11, 5.785e-11,
00617
                                4.178e-11, 3.018e-11, 2.18e-11, 1.574e-11, 1.137e-11, 8.211e-12,
00618
                                5.973e-12, 4.327e-12, 3.118e-12, 2.234e-12, 1.573e-12, 1.04e-12,
                                6.762e-13, 4.202e-13, 2.406e-13, 1.335e-13, 6.938e-14, 3.105e-14,
00620
                                1.609e-14, 1.033e-14, 6.432e-15, 4.031e-15, 2.555e-15, 1.656e-15, 1.115e-15, 7.904e-16, 5.63e-16, 4.048e-16, 2.876e-16, 2.004e-16, 1.356e-16, 9.237e-17, 6.235e-17, 4.223e-17, 3.009e-17, 2.328e-17,
00621
00622
00623
                                2.002e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17,
00624
                                1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17,
                                1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 
00626
00627
00628
                               1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17,
                                1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 
00629
00630
```

```
1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17,
                     1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 
00633
00634
                     1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17,
00635
                     1.914e-17
00636
00638
                static double no[121] = {
00639
                   2.586e-10, 4.143e-11, 1.566e-11, 9.591e-12, 8.088e-12, 8.462e-12,
00640
                     1.013e-11, 1.328e-11, 1.855e-11, 2.678e-11, 3.926e-11, 5.464e-11,
                     7.012e-11, 8.912e-11, 1.127e-10, 1.347e-10, 1.498e-10, 1.544e-10,
00641
                     1.602e-10, 1.824e-10, 2.078e-10, 2.366e-10, 2.691e-10, 5.141e-10,
00642
                     8.259e-10, 1.254e-09, 1.849e-09, 2.473e-09, 3.294e-09, 4.16e-09, 5.095e-09, 6.11e-09, 6.93e-09, 7.888e-09, 8.903e-09, 9.713e-09,
00643
00644
00645
                     1.052e-08, 1.115e-08, 1.173e-08, 1.21e-08, 1.228e-08, 1.239e-08,
00646
                     1.231e-08, 1.213e-08, 1.192e-08, 1.138e-08, 1.085e-08, 1.008e-08, 9.224e-09, 8.389e-09, 7.262e-09, 6.278e-09, 5.335e-09, 4.388e-09,
00647
                     3.589e-09, 2.761e-09, 2.129e-09, 1.633e-09, 1.243e-09, 9.681e-10,
00648
                     8.355e-10, 7.665e-10, 7.442e-10, 8.584e-10, 9.732e-10, 1.063e-09,
                     1.163e-09, 1.286e-09, 1.472e-09, 1.707e-09, 2.032e-09, 2.474e-09,
                     2.977e-09, 3.506e-09, 4.102e-09, 5.013e-09, 6.493e-09, 8.414e-09,
00651
00652
                     1.077e-08, 1.367e-08, 1.777e-08, 2.625e-08, 3.926e-08, 5.545e-08,
00653
                     7.195e-08, 9.464e-08, 1.404e-07, 2.183e-07, 3.329e-07, 4.535e-07,
                     6.158e-07, 8.187e-07, 1.075e-06, 1.422e-06, 1.979e-06, 2.71e-06, 3.58e-06, 4.573e-06, 5.951e-06, 7.999e-06, 1.072e-05, 1.372e-05,
00654
00655
                     1.697e-05, 2.112e-05, 2.643e-05, 3.288e-05, 3.994e-05, 4.794e-05,
                     5.606e-05, 6.383e-05, 7.286e-05, 8.156e-05, 8.883e-05, 9.469e-05,
00657
00658
                     9.848e-05, 0.0001023, 0.0001066, 0.0001115, 0.0001145, 0.0001142,
00659
                    0.0001133
00660
00661
00662
                static double no2[121] = {
                   3.036e-09, 2.945e-10, 9.982e-11, 5.069e-11, 3.485e-11, 2.982e-11,
00663
00664
                     2.947e-11, 3.164e-11, 3.714e-11, 4.586e-11, 6.164e-11, 8.041e-11,
                     9.982e-11, 1.283e-10, 1.73e-10, 2.56e-10, 3.909e-10, 5.959e-10,
00665
                     9.081e-10, 1.384e-09, 1.788e-09, 2.189e-09, 2.686e-09, 3.091e-09,
00666
                     3.49e-09, 3.796e-09, 4.2e-09, 5.103e-09, 6.005e-09, 6.3e-09, 6.706e-09, 7.07e-09, 7.434e-09, 7.663e-09, 7.788e-09, 7.8e-09, 7.597e-09,
00667
                     7.482e-09, 7.227e-09, 6.403e-09, 5.585e-09, 4.606e-09, 3.703e-09,
                     2.984e-09, 2.183e-09, 1.48e-09, 8.441e-10, 5.994e-10, 3.799e-10,
00670
00671
                     2.751e-10, 1.927e-10, 1.507e-10, 1.102e-10, 6.971e-11, 5.839e-11,
                     3.904 e-11, \ 3.087 e-11, \ 2.176 e-11, \ 1.464 e-11, \ 1.209 e-11, \ 8.497 e-12,
00672
                     6.477e-12, 4.371e-12, 2.914e-12, 2.424e-12, 1.753e-12, 1.35e-12, 9.417e-13, 6.622e-13, 5.148e-13, 3.841e-13, 3.446e-13, 3.01e-13,
00673
00674
                     2.551e-13, 2.151e-13, 1.829e-13, 1.64e-13, 1.475e-13, 1.352e-13,
00676
                     1.155e-13, 9.963e-14, 9.771e-14, 9.577e-14, 9.384e-14, 9.186e-14,
00677
                     9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14,
00678
                     9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14,
00679
                     9e-14, 9e-14
00680
00682
                static double o3[121] = {}
00683
                 2.218e-08, 3.394e-08, 3.869e-08, 4.219e-08, 4.501e-08, 4.778e-08, 5.067e-08, 5.402e-08, 5.872e-08, 6.521e-08, 7.709e-08, 9.461e-08, 1.269e-07, 1.853e-07, 2.723e-07, 3.964e-07, 5.773e-07, 8.2e-07, 1.155e-06, 1.59e-06, 2.076e-06, 2.706e-06, 3.249e-06, 3.848e-06,
00684
00685
00686
                     4.459e-06, 4.986e-06, 5.573e-06, 5.958e-06, 6.328e-06, 6.661e-06,
00688
                     6.9e-06, 7.146e-06, 7.276e-06, 7.374e-06, 7.447e-06, 7.383e-06,
00689
00690
                     7.321e-06, 7.161e-06, 6.879e-06, 6.611e-06, 6.216e-06, 5.765e-06,
                     5.355e-06, 4.905e-06, 4.471e-06, 4.075e-06, 3.728e-06, 3.413e-06,
00691
                     3.125e-06, 2.856e-06, 2.607e-06, 2.379e-06, 2.17e-06, 1.978e-06,
00692
00693
                     1.8e-06, 1.646e-06, 1.506e-06, 1.376e-06, 1.233e-06, 1.102e-06,
                     9.839e-07, 8.771e-07, 7.814e-07, 6.947e-07, 6.102e-07, 5.228e-07, 4.509e-07, 3.922e-07, 3.501e-07, 3.183e-07, 2.909e-07, 2.686e-07,
00695
00696
                     2.476e-07, 2.284e-07, 2.109e-07, 2.003e-07, 2.013e-07, 2.022e-07,
                     2.032e-07, 2.042e-07, 2.097e-07, 2.361e-07, 2.656e-07, 2.989e-07, 3.37e-07, 3.826e-07, 4.489e-07, 5.26e-07, 6.189e-07, 7.312e-07, 8.496e-07, 8.444e-07, 8.392e-07, 8.339e-07, 8.286e-07, 8.234e-07,
00697
00698
00699
                     8.181e-07, 8.129e-07, 8.077e-07, 8.026e-07, 6.918e-07, 5.176e-07,
00701
                     3.865e-07, 2.885e-07, 2.156e-07, 1.619e-07, 1.219e-07, 9.161e-08,
00702
                     6.972e-08, 5.399e-08, 3.498e-08, 2.111e-08, 1.322e-08, 8.482e-09,
00703
                     5.527e-09, 3.423e-09, 2.071e-09, 1.314e-09, 8.529e-10, 5.503e-10,
00704
                     3.665e-10
00705
00706
00707
                static double ocs[121] = {
00708
                 6e-10, 6e-10, 6e-10, 6e-10, 6e-10, 6e-10, 6e-10, 6e-10, 6e-10, 5.997e-10,
00709
                     5.989e-10, 5.881e-10, 5.765e-10, 5.433e-10, 5.074e-10, 4.567e-10,
                     4.067e-10, 3.601e-10, 3.093e-10, 2.619e-10, 2.232e-10, 1.805e-10,
00710
                    1.46e-10, 1.187e-10, 8.03e-11, 5.435e-11, 3.686e-11, 2.217e-11, 1.341e-11, 8.756e-12, 4.511e-12, 2.37e-12, 1.264e-12, 8.28e-13,
00711
00713
                     5.263e-13, 3.209e-13, 1.717e-13, 9.068e-14, 4.709e-14, 2.389e-14,
00714
                     1.236e-14, 1.127e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14,
00715
                     1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14,
                     1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 
00716
00717
```

```
1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14,
                        1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 
00719
00720
00721
                         1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14,
00722
                         1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14,
00723
                          1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14,
00724
                         1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14,
00725
                          1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14,
00726
                         1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14,
00727
                        1.091e-14, 1.091e-14, 1.091e-14
00728
00729
00730
                    static double sf6[121] = {
                      4.103e-12, 4.103e-12, 4.103e-12, 4.103e-12, 4.103e-12, 4.103e-12,
00731
00732
                          4.103e-12, 4.103e-12, 4.103e-12, 4.087e-12, 4.064e-12, 4.023e-12,
                         3.988e-12, 3.941e-12, 3.884e-12, 3.755e-12, 3.622e-12, 3.484e-12, 3.32e-12, 3.144e-12, 2.978e-12, 2.811e-12, 2.653e-12, 2.489e-12,
00733
00734
                         2.332e-12, 2.199e-12, 2.089e-12, 2.013e-12, 1.953e-12, 1.898e-12,
00735
                         1.859e-12, 1.826e-12, 1.798e-12, 1.776e-12, 1.757e-12, 1.742e-12,
00737
                          1.728e-12, 1.717e-12, 1.707e-12, 1.698e-12, 1.691e-12, 1.685e-12,
00738
                         1.679e-12, 1.675e-12, 1.671e-12, 1.668e-12, 1.665e-12, 1.663e-12,
00739
                         1.661e-12, 1.659e-12, 1.658e-12, 1.657e-12, 1.656e-12, 1.655e-12,
00740
                         1.654e-12, 1.653e-12, 1.653e-12, 1.652e-12, 1.652e-12, 1.652e-12,
                         1.651e-12, 1.651e-12, 1.651e-12, 1.651e-12, 1.651e-12, 1.651e-12, 1.651e-12, 1.651e-12, 1.65e-12, 1.65e-12
00741
00742
                         1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12,
00743
00744
                          1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12,
00745
                         1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12,
00746
                         1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12,
00747
                         1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12,
                         1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12
00748
00749
00750
00751
                    static double so2[121] = {
00752
                        le-10, le-10, le-10, le-10, le-10, le-10, le-10, le-10, le-10, le-10,
00753
00754
                          1e-10, 1e-10, 9.867e-11, 9.537e-11, 9e-11, 8.404e-11, 7.799e-11,
                          7.205e-11, 6.616e-11, 6.036e-11, 5.475e-11, 5.007e-11, 4.638e-11,
00756
                          4.346e-11, 4.055e-11, 3.763e-11, 3.471e-11, 3.186e-11, 2.905e-11,
00757
                         2.631e-11, 2.358e-11, 2.415e-11, 2.949e-11, 3.952e-11, 5.155e-11,
                          6.76e-11, 8.741e-11, 1.099e-10, 1.278e-10, 1.414e-10, 1.512e-10,
00758
                         1.607e-10, 1.699e-10, 1.774e-10, 1.832e-10, 1.871e-10, 1.907e-10, 1.943e-10, 1.974e-10, 1.993e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10,
00759
00760
                         2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10,
00761
                         2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10,
00762
00763
                         2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10,
00764
                         2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10,
00765
                         2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10,
                         2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e
00766
00767
00768
00769
00770
                   static int ig_co2 = -999;
00771
00772
                   double co2, *q[NG] = { NULL };
00773
00774
                   int ig, ip, iw, iz;
00775
00776
                     /* Find emitter index of CO2... */
                    if (ig_co2 == -999)
ig_co2 = find_emitter(ct1, "CO2");
00777
00778
00779
                    /* Identify variable... */
00781
                    for (ig = 0; ig < ctl->ng; ig++) {
                        q[ig] = NULL;
00782
00783
                         if (strcasecmp(ctl->emitter[ig], "C2H2") == 0)
00784
                              q[ig] = c2h2;
00785
                         if (strcasecmp(ctl->emitter[iq], "C2H6") == 0)
00786
                             q[ig] = c2h6;
00787
                         if
                                (strcasecmp(ctl->emitter[ig], "CCl4") == 0)
00788
                              q[ig] = ccl4;
00789
                         if (strcasecmp(ctl->emitter[ig], "CH4") == 0)
00790
                              q[ig] = ch4;
00791
                         if (strcasecmp(ctl->emitter[ig], "ClO") == 0)
00792
                             q[ig] = clo;
                          if (strcasecmp(ctl->emitter[ig], "ClONO2") == 0)
00793
00794
                              q[ig] = clono2;
00795
                                 (strcasecmp(ctl->emitter[ig], "CO") == 0)
                              q[ig] = co;
00796
00797
                         if (strcasecmp(ctl->emitter[ig], "COF2") == 0)
00798
                             q[ig] = cof2;
                                 (strcasecmp(ctl->emitter[ig], "F11") == 0)
                              q[ig] = f11;
00800
00801
                                 (strcasecmp(ctl->emitter[ig], "F12") == 0)
                         q[ig] = f12;
if (strcasecmp(ctl->emitter[ig], "F14") == 0)
00802
00803
00804
                              q[ig] = f14;
```

```
if (strcasecmp(ctl->emitter[ig], "F22") == 0)
00806
            q[ig] = f22;
          if (strcasecmp(ctl->emitter[ig], "H2O") == 0)
00807
00808
            q[ig] = h2o;
00809
          if (strcasecmp(ctl->emitter[ig], "H2O2") == 0)
00810
            q[ig] = h2o2;
          if (strcasecmp(ctl->emitter[ig], "HCN") == 0)
00811
00812
            q[ig] = hcn;
00813
          if (strcasecmp(ctl->emitter[ig], "HNO3") == 0)
            q[ig] = hno3;
00814
          q[ig] = hno4;
if (street)
          if (strcasecmp(ctl->emitter[ig], "HNO4") == 0)
00815
00816
00817
             (strcasecmp(ctl->emitter[ig], "HOCl") == 0)
            q[ig] = hocl;
00818
00819
          if (strcasecmp(ctl->emitter[ig], "N2O") == 0)
          q[ig] = n2o;
if (strcasecmp(ctl->emitter[ig], "N2O5") == 0)
00820
00821
00822
            q[ig] = n2o5;
00823
          if (strcasecmp(ctl->emitter[ig], "NH3") == 0)
00824
            q[ig] = nh3;
00825
          if (strcasecmp(ctl->emitter[ig], "NO") == 0)
00826
            q[ig] = no;
          if (strcasecmp(ctl->emitter[ig], "NO2") == 0)
00827
            q[ig] = no2;
00828
00829
          if (strcasecmp(ctl->emitter[iq], "03") == 0)
            q[ig] = o3;
00830
00831
             (strcasecmp(ctl->emitter[ig], "OCS") == 0)
            q[ig] = ocs;
00832
          if (strcasecmp(ctl->emitter[ig], "SF6") == 0)
00833
00834
            q[ig] = sf6;
          if (strcasecmp(ctl->emitter[iq], "SO2") == 0)
00835
00836
            q[ig] = so2;
00837
00838
00839
        /\star Loop over atmospheric data points... \star/
00840
        for (ip = 0; ip < atm->np; ip++) {
00841
00842
           /* Get altitude index... */
00843
          iz = locate_reg(z, 121, atm->z[ip]);
00844
00845
          /* Interpolate pressure... */
00846
          atm \rightarrow p[ip] = EXP(z[iz], pre[iz], z[iz + 1], pre[iz + 1], atm \rightarrow z[ip]);
00847
00848
          /* Interpolate temperature... */
          atm \rightarrow t[ip] = LIN(z[iz], tem[iz], z[iz + 1], tem[iz + 1], atm \rightarrow z[ip]);
00849
00850
00851
           /* Interpolate trace gases... */
          for (ig = 0; ig < ctl->ng; ig++)
  if (q[ig] != NULL)
00852
00853
              atm->q[ig][ip] =
00854
00855
                LIN(z[iz], q[ig][iz], z[iz + 1], q[ig][iz + 1], atm->z[ip]);
00856
00857
               atm->q[ig][ip] = 0;
00858
           /* Set CO2... */
00859
          if (ig_co2 >= 0) {
00860
            co2 =
00862
               371.789948e-6 + 2.026214e-6 * (atm->time[ip] - 63158400.) / 31557600.;
00863
             atm->q[ig\_co2][ip] = co2;
00864
00865
          /* Set extinction to zero... */
for (iw = 0; iw < ctl->nw; iw++)
00866
00867
00868
            atm->k[iw][ip] = 0;
00869
00870 }
```

Here is the call graph for this function:



5.5.2.6 double ctmco2 (double nu, double p, double t, double u)

Compute carbon dioxide continuum (optical depth).

Definition at line 874 of file jurassic.c.

```
00878
00880
          static double co2296[2001] = { 9.3388e-5, 9.7711e-5, 1.0224e-4, 1.0697e-4,
00881
            1.1193e-4, 1.1712e-4, 1.2255e-4, 1.2824e-4, 1.3419e-4, 1.4043e-4,
00882
            1.4695e-4, 1.5378e-4, 1.6094e-4, 1.6842e-4, 1.7626e-4, 1.8447e-4,
            1.9307e-4, 2.0207e-4, 2.1149e-4, 2.2136e-4, 2.3169e-4, 2.4251e-4, 2.5384e-4, 2.657e-4, 2.7813e-4, 2.9114e-4, 3.0477e-4, 3.1904e-4,
00883
00884
            3.3399e-4, 3.4965e-4, 3.6604e-4, 3.8322e-4, 4.0121e-4, 4.2006e-4,
00886
            4.398e-4, 4.6047e-4, 4.8214e-4, 5.0483e-4, 5.286e-4, 5.535e-4,
00887
            5.7959e-4, 6.0693e-4, 6.3557e-4, 6.6558e-4, 6.9702e-4, 7.2996e-4,
            7.6449e-4, 8.0066e-4, 8.3856e-4, 8.7829e-4, 9.1991e-4, 9.6354e-4, .0010093, .0010572, .0011074, .00116, .0012152, .001273, .0013336, .0013972, .0014638, .0015336, .0016068, .0016835, .001764, .0018483, .0019367, .0020295, .0021267, .0022286,
00888
00889
00890
00892
            .0023355, .0024476, .0025652, .0026885, .0028178, .0029534
00893
            .0030956, .0032448, .0034012, .0035654, .0037375, .0039181,
00894
            .0041076, .0043063, .0045148, .0047336, .0049632, .005204,
            .0054567, .0057219, .0060002, .0062923, .0065988, .0069204,
00895
            .007258, .0076123, .0079842, .0083746, .0087844, .0092146, .0096663, .01014, .010638, .011161, .01171, .012286, .012891, .013527, .014194, .014895, .015631, .016404, .017217, .01807,
00896
00897
00898
00899
            .018966, .019908, .020897, .021936, .023028, .024176, .025382,
00900
            .026649, .027981, .02938, .030851, .032397, .034023, .035732,
            .037528, .039416, .041402, .04349, .045685, .047994, .050422, .052975, .055661, .058486, .061458, .064584, .067873, .071334, .074975, .078807, .082839, .087082, .091549, .096249, .1012,
00901
00902
00903
            00904
00905
            .23967, .25229, .2656, .27964, .29443, .31004, .3265, .34386,
00906
            .36218, .3815, .40188, .42339, .44609, .47004, .49533, .52202, .5502, .57995, .61137, .64455, .6796, .71663, .75574, .79707, .84075, .88691, .9357, .98728, 1.0418, 1.0995, 1.1605, 1.225,
00907
00908
00909
            1.2932, 1.3654, 1.4418, 1.5227, 1.6083, 1.6989, 1.7948, 1.8964,
00911
            2.004, 2.118, 2.2388, 2.3668, 2.5025, 2.6463, 2.7988, 2.9606,
00912
            3.1321, 3.314, 3.5071, 3.712, 3.9296, 4.1605, 4.4058, 4.6663,
00913
            4.9431, 5.2374, 5.5501, 5.8818, 6.2353, 6.6114, 7.0115, 7.4372,
00914
            7.8905, 8.3731, 8.8871, 9.4349, 10.019, 10.641, 11.305, 12.013,
00915
            12.769, 13.576, 14.437, 15.358, 16.342, 17.39, 18.513, 19.716,
00916
            21.003, 22.379, 23.854, 25.436, 27.126, 28.942, 30.89, 32.973,
            35.219, 37.634, 40.224, 43.021, 46.037, 49.29, 52.803,
00917
00918
            60.418, 64.792, 69.526, 74.637, 80.182, 86.193, 92.713, 99.786
00919
            107.47, 115.84, 124.94, 134.86, 145.69, 157.49, 170.3, 184.39,
            199.83, 216.4, 234.55, 254.72, 276.82, 299.85, 326.16, 354.99, 386.51, 416.68, 449.89, 490.12, 534.35, 578.25, 632.26, 692.61
00920
00921
                                                                        1219.2,
00922
            756.43, 834.75, 924.11, 1016.9, 996.96, 1102.7,
00923
            1494.3, 1654.1, 1826.5, 2027.9, 2249., 2453.8, 2714.4, 2999.4,
00924
            3209.5, 3509., 3840.4, 3907.5, 4190.7, 4533.5, 4648.3, 5059.1,
00925
            5561.6, 6191.4, 6820.8, 7905.9, 9362.2, 2431.3, 2211.3, 2046.8,
00926
            2023.8, 1985.9, 1905.9, 1491.1, 1369.8, 1262.2, 1200.7, 887.74,
00927
            820.25, 885.23, 887.21, 816.73, 1126.9, 1216.2, 1272.4, 1579.5,
00928
            1634.2, 1656.3, 1657.9, 1789.5, 1670.8, 1509.5, 8474.6, 7489.2,
            6793.6, 6117., 5574.1, 5141.2, 5084.6, 4745.1, 4413.2, 4102.8,
```

```
4024.7, 3715., 3398.6, 3100.8, 2900.4, 2629.2, 2374., 2144.7,
                         1955.8, 1760.8, 1591.2, 1435.2, 1296.2, 1174., 1065.1, 967.76, 999.48, 897.45, 809.23, 732.77, 670.26, 611.93, 560.11, 518.77,
00932
                         476.84, 438.8, 408.48, 380.21, 349.24, 322.71, 296.65, 272.85,
00933
                        251.96, 232.04, 213.88, 197.69, 182.41, 168.41, 155.79, 144.05, 133.31, 123.48, 114.5, 106.21, 98.591, 91.612, 85.156, 79.204, 73.719, 68.666, 63.975, 59.637, 56.35, 52.545, 49.042, 45.788, 42.78, 39.992, 37.441, 35.037, 32.8, 30.744, 28.801, 26.986,
00934
00935
00937
00938
                         25.297, 23.731, 22.258, 20.883, 19.603, 18.403, 17.295, 16.249,
                        15.271, 14.356, 13.501, 12.701, 11.954, 11.254, 10.6, 9.9864, 9.4118, 8.8745, 8.3714, 7.8997, 7.4578, 7.0446, 6.6573, 6.2949, 5.9577, 5.6395, 5.3419, 5.063, 4.8037, 4.5608, 4.3452, 4.1364, 3.9413, 3.7394, 3.562, 3.3932, 3.2325, 3.0789, 2.9318, 2.7898, 2.6537, 2.5225, 2.3958, 2.2305, 2.1215, 2.0245, 1.9427, 1.8795,
00939
00940
00941
00942
00943
00944
                         1.8336, 1.7604, 1.7016, 1.6419, 1.5282, 1.4611, 1.3443, 1.27,
00945
                         1.1675, 1.0824, 1.0534, .99833, .95854, .92981, .90887, .89346,
00946
                         .88113, .87068, .86102, .85096, .88262, .86151, .83565, .80518,
                         .77045, .73736, .74744, .74954, .75773, .82267, .83493, .89402, .89725, .93426, .95564, .94045, .94174, .93404, .92035, .90456, .88621, .86673, .78117, .7515, .72056, .68822, .65658, .62764,
00947
                         .55984, .55598, .57407, .60963, .63763, .66198, .61132, .60972,
00950
00951
                          .52496, .50649, .41872, .3964, .32422, .27276, .24048, .23772,
                         .2286, .22711, .23999, .32038, .34371, .36621, .38561, .39953, .40636, .44913, .42716, .3919, .35477, .33935, .3351, .39746, .40993, .49398, .49956, .56157, .54742, .57295, .57386, .55417,
00952
00953
00954
                         .50745, .471, .43446, .39102, .34993, .31269, .27888, .24912, .22291, .19994, .17972, .16197, .14633, .13252, .12029, .10942,
00956
                         .099745, .091118, .083404, .076494, .070292, .064716, .059697, .055173, .051093, .047411, .044089, .041092, .038392, .035965,
00957
00958
                        .033789, .031846, .030122, .028607, .02729, .026169, .025209, .024405, .023766, .023288, .022925, .022716, .022681, .022685, .022768, .023133, .023325, .023486, .024004, .024126, .024083, .023785, .024023, .023029, .021649, .021108, .019454, .017809, .017292, .016635, .017037, .018068, .018977, .018756, .017847, .016675, .016675, .017037, .018068, .018977, .018756, .017847, .016675, .017037, .018068, .018077, .018756, .017037, .018068, .018077, .018756, .017037, .018068, .018077, .018756, .017037, .018068, .018077, .018756, .017037, .018068, .018077, .018756, .017037, .018068, .018077, .018756, .017037, .018068, .018077, .018756, .017037, .018068, .018077, .018756, .017037, .018068, .018077, .018756, .017037, .018068, .018077, .018756, .018077, .018756, .018077, .018756, .018077, .018756, .018077, .018756, .018077, .018756, .018077, .018068, .018077, .018756, .018077, .018068, .018077, .018756, .018077, .018068, .018077, .018756, .018077, .018078, .018077, .018078, .018078, .018077, .018078, .018078, .018077, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018078, .018
00959
00960
00962
00963
                        .016557, .016142, .014459, .012869, .012381, .010875, .0098701, .009285, .0091698, .0091701, .0096145, .010553, .01106, .012613, .014362, .015017, .016507, .017741, .01768, .017784, .0171, .016357, .016172, .017257, .018978, .020935, .021741, .023567, .025183, .025589, .026732, .027648, .028278, .028215, .02856,
00964
00965
00966
00968
                        .023163, .023163, .026732, .027646, .026276, .026213, .026523, .029015, .029062, .028851, .028497, .027825, .027801, .026523, .02487, .022967, .022168, .020194, .018605, .017903, .018439, .019697, .020311, .020855, .020057, .018608, .016738, .015963, .013844, .011801, .011134, .0097573, .0086007, .0086226, .0083721, .0090978, .0097616, .0098426, .011317, .012853, .014
00969
00970
00971
00972
                        .014657, .015771, .016351, .016079, .014829, .013431, .013185, .013207, .01448, .016176, .017971, .018265, .019526, .020455,
00975
                        .013207, .01448, .016176, .017971, .018265, .019326, .020455, .019777, .019802, .01944, .018176, .017505, .016197, .015339, .014401, .013213, .012203, .011186, .010236, .0093288, .0084854, .0076837, .0069375, .0062614, .0056628, .0051153, .0046015, .0041501, .003752, .0033996, .0030865, .0028077, .0025586, .0023355, .0021353, .0019553, .0017931, .0016466, .0015141, .0013941, .0012852, .0011862, .0010962, .0010142, 9.3935e-4, 8.71e-4, 8.0851e-4, 7.5132e-4, 6.9894e-4, 6.5093e-4, 6.0689e-4,
00976
00977
00978
00979
00981
00982
                         5.6647e-4, 5.2935e-4, 4.9525e-4, 4.6391e-4, 4.3509e-4, 4.086e-4, 3.8424e-4, 3.6185e-4, 3.4126e-4, 3.2235e-4, 3.0498e-4, 2.8904e-4,
00983
00984
                         2.7444e-4, 2.6106e-4, 2.4883e-4, 2.3766e-4, 2.275e-4, 2.1827e-4,
00985
                         2.0992e-4, 2.0239e-4, 1.9563e-4, 1.896e-4, 1.8427e-4, 1.796e-4,
                         1.7555e-4, 1.7209e-4, 1.692e-4, 1.6687e-4, 1.6505e-4, 1.6375e-4
00987
                         1.6294e-4, 1.6261e-4, 1.6274e-4, 1.6334e-4, 1.6438e-4, 1.6587e-4,
00988
00989
                         1.678e-4, 1.7017e-4, 1.7297e-4, 1.762e-4, 1.7988e-4, 1.8399e-4,
                         1.8855e-4, 1.9355e-4, 1.9902e-4, 2.0494e-4, 2.1134e-4, 2.1823e-4, 2.2561e-4, 2.335e-4, 2.4192e-4, 2.5088e-4, 2.604e-4, 2.705e-4, 2.8119e-4, 2.9251e-4, 3.0447e-4, 3.171e-4, 3.3042e-4, 3.4447e-4,
00990
00991
                         3.5927e-4, 3.7486e-4, 3.9127e-4, 4.0854e-4, 4.267e-4, 4.4579e-4, 4.6586e-4, 4.8696e-4, 5.0912e-4, 5.324e-4, 5.5685e-4, 5.8253e-4,
00993
00994
00995
                         6.0949e-4, 6.378e-4, 6.6753e-4, 6.9873e-4, 7.3149e-4, 7.6588e-4,
00996
                         8.0198e-4, 8.3987e-4, 8.7964e-4, 9.2139e-4, 9.6522e-4,
                                                                                                                                                                       .0010112.
                         .0010595, .0011102, .0011634, .0012193, .001278, .0013396, .0014043, .0014722, .0015436, .0016185, .0016972, .0017799,
00997
00998
                         .0018668, .001958, .0020539, .0021547, .0022606, .0023719,
                         .002489, .002612, .0027414, .0028775, .0030206, .0031712,
01000
01001
                         .0033295, .0034962, .0036716, .0038563, .0040506, .0042553,
                         .0044709, .004698, .0049373, .0051894, .0054552, .0057354, .006031, .0063427, .0066717, .0070188, .0073854, .007726,
01002
01003
                         .0081816, .0086138, .0090709, .0095543, .010066, .010607,
01004
                         .011181, .011789, .012433, .013116, .013842, .014613, .015432, .016304, .017233, .018224, .019281, .020394, .021574, .022836,
01006
                         01007
01008
01009
01010
                         . 28811, .31399, .34321, .36606, .39675, .42742, .44243, .47197, .49993, .49027, .51147, .52803, .48931, .49729, .5026, .43854,
01012
01013
                         .441, .44766, .43414, .46151, .50029, .55247, .43855, .32115, .32607, .3431, .36119, .38029, .41179, .43996, .47144, .51853, .55362, .59122, .66338, .69877, .74001, .82923, .86907, .90361,
01014
01015
01016
```

```
01017
                   1.0025, 1.031, 1.0559, 1.104, 1.1178, 1.1341, 1.1547, 1.351,
                   1.4772, 1.4812, 1.4907, 1.512, 1.5422, 1.5853, 1.6358, 1.6963, 1.7674, 1.8474, 1.9353, 2.0335, 2.143, 2.2592, 2.3853, 2.5217,
01018
01019
01020
                   2.6686, 2.8273, 2.9998, 3.183, 3.3868, 3.6109, 3.8564, 4.1159,
                   4.4079, 4.7278, 5.0497, 5.3695, 5.758, 6.0834, 6.4976, 6.9312, 7.38, 7.5746, 7.9833, 8.3791, 8.3956, 8.7501, 9.1067, 9.072,
01021
01022
                    9.4649, 9.9112, 10.402, 10.829, 11.605, 12.54, 12.713, 10.443,
                   10.825, 11.375, 11.955, 12.623, 13.326, 14.101, 15.041, 15.547
01024
                   16.461, 17.439, 18.716, 19.84, 21.036, 22.642, 23.901, 25.244, 27.03, 28.411, 29.871, 31.403, 33.147, 34.744, 36.456, 39.239, 43.605, 45.162, 47.004, 49.093, 51.391, 53.946, 56.673, 59.629, 63.167, 66.576, 70.254, 74.222, 78.477, 83.034, 87.914, 93.18, 98.77, 104.74, 111.15, 117.95, 125.23, 133.01, 141.33, 150.21,
01025
01026
01027
01028
                   159.71, 169.89, 180.93, 192.54, 204.99, 218.34, 232.65, 248.,
01030
01031
                   264.47, 282.14, 301.13, 321.53, 343.48, 367.08, 392.5, 419.88,
                   449.4, 481.26, 515.64, 552.79, 592.99, 636.48, 683.61, 734.65, 789.99, 850.02, 915.14, 985.81, 1062.5, 1147.1, 1237.8, 1336.4, 1443.2, 1558.9, 1684.2, 1819.2, 1965.2, 2122.6, 2291.7, 2470.8, 2665.7, 2874.9, 3099.4, 3337.9, 3541., 3813.3, 4111.9, 4439.3, 4798.9, 5196., 5639.2, 6087.5, 6657.7, 7306.7, 8040.7, 8845.5,
01032
01033
01034
01036
                   9702.2, 10670., 11739., 12842., 14141., 15498., 17068., 18729., 20557., 22559., 25248., 27664., 30207., 32915., 35611., 38081.,
01037
01038
                   40715., 43191., 41651., 42750., 43785., 44353., 44366., 44189.,
01039
                   43618., 42862., 41878., 35133., 35215., 36383., 39420., 44055., 44155., 45850., 46853., 39197., 38274., 29942., 28553., 21792.,
01040
01041
                   21228., 17106., 14955., 18181., 19557., 21427., 23728., 26301., 28584., 30775., 32536., 33867., 40089., 39204., 37329., 34452.,
01042
01043
01044
                   31373., 33921., 34800., 36043., 44415., 45162., 52181., 50895.,
01045
                   54140., 50840., 50468., 48302., 44915., 40910., 36754., 32755.,
                   29093., 25860., 22962., 20448., 18247., 16326., 14645., 13165., 11861., 10708., 9686.9, 8779.7, 7971.9, 7250.8, 6605.7, 6027.2, 5507.3, 5039.1, 4616.6, 4234.8, 3889., 3575.4, 3290.5, 3031.3,
01046
01047
                   2795.2, 2579.9, 2383.1, 2203.3, 2038.6, 1887.6, 1749.1, 1621.9,
01049
01050
                   1505., 1397.4, 1298.3, 1207., 1122.8, 1045., 973.1, 906.64,
                   845.16, 788.22, 735.48, 686.57, 641.21, 599.1, 559.99, 523.64, 489.85, 458.42, 429.16, 401.92, 376.54, 352.88, 330.82, 310.24, 291.03, 273.09, 256.34, 240.69, 226.05, 212.37, 199.57, 187.59, 176.37, 165.87, 156.03, 146.82, 138.17, 130.07, 122.47, 115.34,
01051
01052
01053
01055
                   108.65, 102.37, 96.473, 90.934, 85.73, 80.84, 76.243, 71.922,
                   67.858, 64.034, 60.438, 57.052, 53.866, 50.866, 48.04, 45.379,
01056
01057
                   42.872, 40.51, 38.285, 36.188, 34.211, 32.347, 30.588, 28.929,
                   27.362, 25.884, 24.489, 23.171, 21.929, 20.755, 19.646, 18.599, 17.61, 16.677, 15.795, 14.961, 14.174, 13.43, 12.725, 12.06, 11.431, 10.834, 10.27, 9.7361, 9.2302, 8.7518, 8.2997, 7.8724,
01058
01059
01060
                   7.4674, 7.0848, 6.7226, 6.3794, 6.054, 5.745, 5.4525, 5.1752, 4.9121, 4.6625, 4.4259, 4.2015, 3.9888, 3.7872, 3.5961, 3.4149,
01061
01062
01063
                   3.2431, 3.0802, 2.9257, 2.7792, 2.6402, 2.5084, 2.3834, 2.2648,
                  2.1522, 2.0455, 1.9441, 1.848, 1.7567, 1.6701, 1.5878, 1.5097, 1.4356, 1.3651, 1.2981, 1.2345, 1.174, 1.1167, 1.062, 1.0101, .96087, .91414, .86986, .82781, .78777, .74971, .71339, .67882, .64604, .61473, .58507, .55676, .52987, .5044, .48014, .45715,
01064
01065
01066
                   .43527, .41453, .3948, .37609, .35831, .34142, .32524, .30995, .29536, .28142, .26807, .25527, .24311, .23166, .22077, .21053
01068
01069
                   .20081, .19143, .18261, .17407, .16603, .15833, .15089, .14385, .13707, .13065, .12449, .11865, .11306, .10774, .10266, .097818,
01070
01071
                   .093203, .088815, .084641, .080671, .076892, .073296, .069873, .066613, .06351, .060555, .05774, .055058, .052504, .050071, .047752, .045543, .043438, .041432, .039521, .037699, .035962,
01072
01074
                   .034307, .032729, .031225, .029791, .028423, .02712, .025877, .024692, .023563, .022485, .021458, .020478, .019543, .018652,
01075
01076
                  .017802, .016992, .016219, .015481, .014778, .014107, .013467, .012856, .012274, .011718, .011188, .010682, .0102, .0097393, .0093001, .008881, .0084812, .0080997, .0077358, .0073885, .0070571, .0067409, .0064393, .0061514, .0058768, .0056147, .0053647, .0051262, .0048987, .0046816, .0044745, .0042769,
01077
01078
01080
01081
01082
                    .0040884, .0039088, .0037373, .0035739, .003418, .0032693,
                   .0031277, .0029926, .0028639, .0027413, .0026245, .0025133, .0024074, .0023066, .0022108, .0021196, .002033, .0019507, .0018726, .0017985, .0017282, .0016617, .0015988, .0015394, .0014834, .0014306, .0013811, .0013346, .0012911, .0012506,
01083
01084
01085
                   .0012131, .0011784, .0011465, .0011175, .0010912, .0010678, .0010472, .0010295, .0010147, .001003, 9.9428e-4, 9.8883e-4
01087
01088
                  9.8673e-4, 9.8821e-4, 9.9343e-4, .0010027, .0010164, .0010348, .0010586, .0010882, .0011245, .0011685, .0012145, .0012666, .0013095, .0013688, .0014048, .0014663, .0015309, .0015499, .0016144, .0016312, .001705, .0017892, .0018499, .0019715, .0021102, .0022442, .0024284, .0025893, .0027703, .0029445,
01089
01090
01091
01092
01093
                   .0031193, .003346, .0034552, .0036906, .0037584, .0040084, .0041934, .0044587, .0047093, .0049759, .0053421, .0055134,
01094
01095
                    .0059048, .0058663, .0061036, .0063259, .0059657, .0060653,
01096
                   .0060972, .0055539, .0055653, .0055772, .005331, .0054953, .0055919, .0058684, .006183, .0066675, .0069808, .0075142, .0078536, .0084282, .0089454, .0094625, .0093703, .0095857,
01097
01099
01100
                    .0099283,
                                       .010063, .010521, .0097778, .0098175, .010379, .010447,
                   .0105, .010617, .010706, .01078, .011177, .011212, .011304, .011446, .011603, .011816, .012165, .012545, .013069, .013539, .01411, .014776, .016103, .017016, .017994, .018978, .01998,
01101
01102
01103
```

```
.021799, .022745, .023681, .024627, .025562, .026992, .027958,
              .029013, .030154, .031402, .03228, .033651, .035272, .037088, .039021, .041213, .043597, .045977, .04877, .051809, .054943,
01106
01107
               .058064, .061528, .06537, .069309, .071928, .075752, .079589,
              .083352, .084096, .087497, .090817, .091198, .094966, .099045
01108
              .10429, .10867, .11518, .12269, .13126, .14087, .15161, .16388, .16423, .1759, .18721, .19994, .21275, .22513, .23041, .24231,
01109
01110
              .25299, .25396, .26396, .27696, .27929, .2908, .30595,
01111
              3282, 3429, 35944, 37467, 39277, 41245, 43326, 45649, 48152, 51897, 54686, 57877, 61263, 64962, 68983, 73945, 78619, 83537, 89622, 95002, 1.0067, 1.0742, 1.1355, 1.2007, 1.2738, 1.347, 1.4254, 1.5094, 1.6009, 1.6976, 1.8019, 1.9148, 2.0357, 2.166, 2.3066, 2.4579, 2.6208, 2.7966, 2.986, 3.188,
01112
01113
01114
01115
01116
              3.4081, 3.6456, 3.9, 4.1747, 4.4712, 4.7931, 5.1359, 5.5097,
01117
01118
              5.9117, 6.3435, 6.8003, 7.3001, 7.8385, 8.3945, 9.011, 9.6869,
              10.392, 11.18, 12.036, 12.938, 13.944, 14.881, 16.029, 17.2 18.574, 19.945, 21.38, 22.9, 24.477, 26.128, 27.87, 29.037,
01119
01120
              30.988, 33.145, 35.506, 37.76, 40.885, 44.487, 48.505, 52.911, 57.56, 61.964, 67.217, 72.26, 78.343, 85.08, 91.867, 99.435,
01121
              107.68, 116.97, 127.12, 138.32, 150.26, 163.04, 174.81, 189.26,
              205.61, 224.68, 240.98, 261.88, 285.1, 307.58, 334.35, 363.53, 394.68, 427.85, 458.85, 489.25, 472.87, 486.93, 496.27, 501.52,
01124
01125
              501.57, 497.14, 488.09, 476.32, 393.76, 388.51, 393.42, 414.45, 455.12, 514.62, 520.38, 547.42, 562.6, 487.47, 480.83, 391.06, 376.92, 303.7, 295.91, 256.03, 236.73, 280.38, 310.71, 335.53,
01126
01127
01128
              367.88, 401.94, 435.52, 469.13, 497.94, 588.82, 597.94, 597.2,
01129
              588.28, 571.2, 555.75, 603.56, 638.15, 680.75, 801.72, 848.01,
01130
01131
              962.15, 990.06, 1068.1, 1076.2, 1115.3, 1134.2, 1136.6, 1119.1,
01132
              1108.9, 1090.6, 1068.7, 1041.9, 1005.4, 967.98, 927.08, 780.1,
              751.41, 733.12, 742.65, 785.56, 855.16, 852.45, 878.1, 784.59, 777.81, 765.13, 622.93, 498.09, 474.89, 386.9, 378.48, 336.17, 322.04, 329.57, 350.5, 383.38, 420.02, 462.39, 499.71, 531.98,
01133
01134
01135
              654.99, 653.43, 639.99, 605.16, 554.16, 504.42, 540.64, 552.33
01136
01137
              679.46, 699.51, 713.91, 832.17, 919.91, 884.96, 907.57, 846.56,
              818.56, 768.93, 706.71, 642.17, 575.95, 515.38, 459.07, 409.02, 364.61, 325.46, 291.1, 260.89, 234.39, 211.01, 190.38, 172.11,
01138
01139
              155.91, 141.49, 128.63, 117.13, 106.84, 97.584, 89.262, 81.756, 74.975, 68.842, 63.28, 58.232, 53.641, 49.46, 45.649, 42.168,
01140
01142
              38.991, 36.078, 33.409, 30.96, 28.71, 26.642, 24.737, 22.985,
              21.37, 19.882, 18.512, 17.242, 16.073, 14.987, 13.984, 13.05
01143
01144
              12.186, 11.384, 10.637, 9.9436, 9.2988, 8.6991, 8.141, 7.6215,
              7.1378, 6.6872, 6.2671, 5.8754, 5.51, 5.1691, 4.851, 4.5539,
01145
              4.2764, 4.0169, 3.7742, 3.5472, 3.3348, 3.1359, 2.9495, 2.7749, 2.6113, 2.4578, 2.3139, 2.1789, 2.0523, 1.9334, 1.8219, 1.7171,
01146
01147
              1.6188, 1.5263, 1.4395, 1.3579, 1.2812, 1.209, 1.1411, 1.0773,
01148
01149
              1.0171, .96048, .90713, .85684, .80959, .76495, .72282, .68309
01150
               .64563, .61035, .57707, .54573, .51622, .48834, .46199, .43709,
01151
               .41359, .39129, .37034, .35064, .33198, .31442, .29784, .28218,
              .26732, .25337, .24017, .22774, .21601, .20479, .19426
01152
01153
           static double co2260[2001] = { 5.7971e-5, 6.0733e-5, 6.3628e-5, 6.6662e-5,
01155
01156
              6.9843e-5, 7.3176e-5, 7.6671e-5, 8.0334e-5, 8.4175e-5, 8.8201e-5,
01157
              9.2421e-5, 9.6846e-5, 1.0149e-4, 1.0635e-4, 1.1145e-4,
                                                                                               1 16796-4
              1.224e-4, 1.2828e-4, 1.3444e-4, 1.409e-4, 1.4768e-4, 1.5479e-4,
01158
              1.6224e-4, 1.7006e-4, 1.7826e-4, 1.8685e-4, 1.9587e-4, 2.0532e-4,
01159
              2.1524e-4, 2.2565e-4, 2.3656e-4, 2.48e-4, 2.6001e-4, 2.7261e-4,
              2.8582e-4, 2.9968e-4, 3.1422e-4, 3.2948e-4, 3.4548e-4, 3.6228e-4,
01161
              3.799e-4, 3.9838e-4, 4.1778e-4, 4.3814e-4, 4.595e-4, 4.8191e-4,
01162
01163
              5.0543e-4, 5.3012e-4, 5.5603e-4, 5.8321e-4, 6.1175e-4, 6.417e-4,
              8.9745e-4, 9.4162e-4, 9.8798e-4, .0010367, .0010878, .0011415,
01164
01165
              .0011978, .001257, .0013191, .0013844, .001453, .0015249,
01166
              .0016006, .00168, .0017634, .001851, .001943, .0020397, .0021412, .0022479, .00236, .0024778, .0026015, .0027316, .0028682,
01168
               .0030117, .0031626, .0033211, .0034877, .0036628, .0038469,
01169
              .0040403, .0042436, .0044574, .004682, .0049182, .0051665, .0054276, .0057021, .0059907, .0062942, .0066133, .0069489, .0073018, .0076729, .0080632, .0084738, .0089056, .0093599,
01170
01171
01172
              .0098377, .01034, .010869, .011426, .012011, .012627, .013276,
              .013958, .014676, .015431, .016226, .017063, .017944, .018872, .019848, .020876, .021958, .023098, .024298, .025561, .026892,
01174
01175
01176
               .028293, .029769, .031323, .032961, .034686, .036503, .038418,
              .040435, .042561, .044801, .047161, .049649, .052271, .055035, .057948, .061019, .064256, .06767, .07127, .075066, .079069, .083291, .087744, .092441, .097396, .10262, .10814, .11396,
01177
01178
              .1201, .12658, .13342, .14064, .14826, .1563, .1648, .17376,
01180
              .18323, .19324, .2038, .21496, .22674, .23919, .25234, .26624, .28093, .29646, .31287, .33021, .34855, .36794, .38844, .41012, .43305, .45731, .48297, .51011, .53884, .56924, .60141, .63547,
01181
01182
01183
              .67152, .70969, .75012, .79292, .83826, .8863, .93718, .99111, 1.0482, 1.1088, 1.173, 1.2411, 1.3133, 1.3898, 1.471, 1.5571, 1.6485, 1.7455, 1.8485, 1.9577, 2.0737, 2.197, 2.3278, 2.4668, 2.6145, 2.7715, 2.9383, 3.1156, 3.3042, 3.5047, 3.7181, 3.9451
01184
01187
              4.1866, 4.4437, 4.7174, 5.0089, 5.3192, 5.65, 6.0025, 6.3782, 6.7787, 7.206, 7.6617, 8.1479, 8.6669, 9.221, 9.8128, 10.445, 11.12, 11.843, 12.615, 13.441, 14.325, 15.271, 16.283, 17.367,
01188
01189
01190
```

```
18.529, 19.776, 21.111, 22.544, 24.082, 25.731, 27.504, 29.409,
                        31.452, 33.654, 36.024, 38.573, 41.323, 44.29, 47.492, 50.951, 54.608, 58.588, 62.929, 67.629, 72.712, 78.226, 84.207, 90.699,
01192
01193
01194
                        97.749, 105.42, 113.77, 122.86, 132.78, 143.61, 155.44, 168.33,
                        182.48, 198.01, 214.87, 233.39, 253.86, 276.34, 300.3, 327.28, 356.89, 389.48, 422.29, 458.99, 501.39, 548.13, 595.62, 652.74,
01195
01196
                        716.54, 784.57, 866.78, 960.59, 1062.8, 1072.5, 1189.5, 1319.4,
01197
                        1467.6, 1630.2, 1813.7, 2016.9, 2253., 2515.3, 2773.5, 3092.8,
01198
01199
                        3444.4, 3720.4, 4104.3, 4527.5, 4645.9, 5021.7, 5462.2, 5597.,
01200
                        6110.6, 6732.5, 7513.8, 8270.6, 9640.6, 11487., 2796.1, 2680.1,
                        2441.6, 2404.2, 2334.8, 2215.2, 1642.5, 1477.9, 1328.1, 223.5, 843.34, 766.96, 831.65, 834.84, 774.85, 1156.3, 1275.6, 1366.1, 1795.6, 1885., 1936.5, 1953.4, 2154.4, 2002.7, 1789.8, 10381., 9040., 8216.5, 7384.7, 6721.9, 6187.7, 6143.8, 5703.9, 5276.6, 4873.1, 4736., 4325.3, 3927., 3554.1, 3286.1, 2950.1, 2642.4,
01201
01202
01203
01204
01205
01206
                        2368.7, 2138.9, 1914., 1719.6, 1543.9, 1388.6, 1252.1, 1132.2
                        1024.1, 1025.4, 920.58, 829.59, 750.54, 685.01, 624.25, 570.14, 525.81, 481.85, 441.95, 408.71, 377.23, 345.86, 318.51, 292.26, 268.34, 247.04, 227.14, 209.02, 192.69, 177.59, 163.78, 151.26,
01207
01208
                         139.73, 129.19, 119.53, 110.7, 102.57, 95.109, 88.264, 81.948,
01210
                        76.13, 70.768, 65.827, 61.251, 57.022, 53.495, 49.824, 46.443,
01211
                       43.307, 40.405, 37.716, 35.241, 32.923, 30.77, 28.78, 26.915, 25.177, 23.56, 22.059, 20.654, 19.345, 18.126, 16.988, 15.93, 14.939, 14.014, 13.149, 12.343, 11.589, 10.884, 10.225, 9.6093, 9.0327, 8.4934, 7.9889, 7.5166, 7.0744, 6.6604, 6.2727, 5.9098,
01212
01213
01214
                        5.5701, 5.2529, 4.955, 4.676, 4.4148, 4.171, 3.9426, 3.7332,
01216
                        3.5347, 3.3493, 3.1677, 3.0025, 2.8466, 2.6994, 2.5601, 2.4277,
01217
01218
                        2.3016, 2.1814, 2.0664, 1.9564, 1.8279, 1.7311, 1.6427, 1.5645,
                       1.4982, 1.443, 1.374, 1.3146, 1.2562, 1.17, 1.1105, 1.0272, 1.96863, .89718, .83654, .80226, .75908, .72431, .69573, .67174, .65126, .63315, .61693, .60182, .58715, .59554, .57649, .55526, .53177, .50622, .48176, .4813, .47642, .47492, .50273, .50293, .52687, .52239, .53419, .53814, .52626, .52211, .51492, .50622,
01219
01220
01221
01222
01223
01224
                        .49746, .48841, .4792, .43534, .41999, .40349, .38586, .36799,
                        .35108, .31089, .30803, .3171, .33599, .35041, .36149, .32924, .32462, .27309, .25961, .20922, .19504, .15683, .13098, .11588,
01225
01226
                        .11478, .11204, .11363, .12135, .16423, .17785, .19094, .20236, .21084, .2154, .24108, .22848, .20871, .18797, .17963, .17834, .21552, .22284, .26945, .27052, .30108, .28977, .29772, .29224,
01227
01229
                        .27658, .24956, .22777, .20654, .18392, .16338, .1452, .12916, .1152, .10304, .092437, .083163, .075031, .067878, .061564, .055976, .051018, .046609, .042679, .03917, .036032, .033223, .030706, .02845, .026428, .024617, .022998, .021554, .02027, .019136, .018141, .017278, .016541, .015926, .015432, .015058,
01230
01231
01232
01233
                        .014807, .014666, .014635, .014728, .014947, .01527, .015728, .016345, .017026, .017798, .018839, .019752, .020636, .021886,
01235
01236
01237
                         .022695, .02327, .023478, .024292, .023544, .022222, .021932,
                        .022093, .02327, .023476, .024292, .023344, .022222, .021932, .020052, .018143, .017722, .017031, .017782, .01938, .020734, .020476, .019255, .017477, .016878, .014617, .012489, .011765, .0099077, .0086446, .0079446, .0078644, .0079763, .008671, .01001, .0108, .012933, .015349, .016341, .018484, .020254,
01238
01239
01240
                        .020254, .020478, .019591, .018595, .018385, .019913, .022254, .024847, .025809, .028053, .029924, .030212, .031367, .03222,
01242
01243
                       .032739, .032537, .03286, .033344, .033507, .033499, .033339, .032809, .033041, .031723, .029837, .027511, .026603, .024032, .021914, .020948, .021701, .023425, .024259, .024987, .023818, .021768, .019223, .018144, .015282, .012604, .01163, .0097907,
01244
01245
01246
                        .008336, .0082473, .0079582, .0088077, .009779, .010129, .012145,
01248
                        .016336, .016247, .00726, .018997, .019998, .019809, .01819, .016358, .016099, .01617, .017939, .020223, .022521, .02277, .024279, .025247, .024222, .023989, .023224, .021493, .020362, .018596, .017309, .015975, .014466, .013171, .011921, .01078, .0097229, .0087612, .0078729, .0076682, .0063494, .0057156,
01249
01250
01251
01252
                        .0051459, .0046273, .0041712, .0037686, .0034119, .003095, .0028126, .0025603, .0023342, .0021314, .0019489, .0017845
01254
01255
                                                                                                                                                     .0017845
                        .001636, .0015017, .00138, .0012697, .0011694, .0010782, 9.9507e-4, 9.1931e-4, 8.5013e-4, 7.869e-4, 7.2907e-4, 6.7611e-4, 6.2758e-4, 5.8308e-4, 5.4223e-4, 5.0473e-4, 4.7027e-4, 4.3859e-4,
01256
01257
01258
                        4.0946e-4, 3.8265e-4, 3.5798e-4, 3.3526e-4, 3.1436e-4, 2.9511e-4,
01260
                        2.7739e-4, 2.6109e-4, 2.4609e-4, 2.3229e-4, 2.1961e-4, 2.0797e-4,
                        1.9729e-4, 1.875e-4, 1.7855e-4, 1.7038e-4, 1.6294e-4, 1.5619e-4,
01261
01262
                        1.5007e-4, 1.4456e-4, 1.3961e-4, 1.3521e-4, 1.3131e-4, 1.2789e-4,
                        1.2494e-4, 1.2242e-4, 1.2032e-4, 1.1863e-4, 1.1733e-4, 1.1641e-4, 1.1585e-4, 1.1565e-4, 1.158e-4, 1.1629e-4, 1.1712e-4, 1.1827e-4, 1.1976e-4, 1.2158e-4, 1.2373e-4, 1.262e-4, 1.2901e-4, 1.3214e-4,
01263
01264
01265
                         1.3562e-4, 1.3944e-4, 1.4361e-4, 1.4814e-4, 1.5303e-4, 1.5829e-4,
01266
                        1.6394e-4, 1.6999e-4, 1.7644e-4, 1.8332e-4, 1.9063e-4, 1.984e-4,
01267
                        2.0663e-4, 2.1536e-4, 2.246e-4, 2.3436e-4, 2.4468e-4, 2.5558e-4, 2.6708e-4, 2.7921e-4, 2.92e-4, 3.0548e-4, 3.1968e-4, 3.3464e-4, 3.5039e-4, 3.6698e-4, 3.8443e-4, 4.0281e-4, 4.2214e-4, 4.4248e-4,
01268
01269
01270
                        1.0389-4, 1.0389-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281-4, 1.0281
01271
                        8.2644e-4, 8.6783e-4, 9.1137e-4, 9.5721e-4, .0010054, .0010562,
01273
01274
                         .0011096, .0011659, .0012251, .0012875, .0013532, .0014224,
01275
                         .0014953, .001572, .0016529, .0017381, .0018279, .0019226,
                        .0020224, .0021277, .0022386, .0023557, .0024792, .0026095, .002747, .0028921, .0030453, .0032071, .003378, .0035586,
01276
```

```
.0037494, .003951, .0041642, .0043897, .0046282, .0048805,
              .0051476, .0054304, .00573, .0060473, .0063837, .0067404, .0071188, .0075203, .0079466, .0083994, .0088806, .0093922
01279
01280
               .0099366, .010516, .011134, .011792, .012494, .013244, .014046,
01281
              .014898, .015808, .016781, .017822, .018929, .020108, .02138, .022729, .02419, .02576, .027412, .029233, .031198, .033301, .035594, .038092, .040767, .04372, .046918, .050246, .053974, .058009, .061976, .066586, .071537, .076209, .081856, .087998
01282
01283
01285
               .093821, .10113, .10913, .11731, .12724, .13821, .15025, .1639
01286
01287
               .17807, .19472, .21356, .23496, .25758, .28387, .31389,
                                                                                                      .34104.
               .37469, .40989, .43309, .46845, .5042, .5023, .52981, .55275,
01288
               .51075, .51976, .52457, .44779, .44721, .4503, .4243, .45244,
01289
               .49491, .55399, .39021, .24802, .2501, .2618, .27475, .28879, .31317, .33643, .36257, .4018, .43275, .46525, .53333, .56599
01290
01291
01292
               .60557, .70142, .74194, .77736, .88567, .91182, .93294, .98407,
               .98772, .99176, .9995, 1.2405, 1.3602, 1.338, 1.3255, 1.3267, 1.3404, 1.3634, 1.3967, 1.4407, 1.4961, 1.5603, 1.6328, 1.7153, 1.8094, 1.9091, 2.018, 2.1367, 2.264, 2.4035, 2.5562, 2.7179, 2.9017, 3.1052, 3.3304, 3.5731, 3.8488, 4.1553, 4.4769, 4.7818,
01293
01294
01295
               5.1711, 5.5204, 5.9516, 6.4097, 6.8899, 7.1118, 7.5469, 7.9735,
               7.9511, 8.3014, 8.6418, 8.4757, 8.8256, 9.2294, 9.6923, 10.033, 10.842, 11.851, 11.78, 8.8435, 9.1381, 9.5956, 10.076, 10.629,
01298
01299
               11.22, 11.883, 12.69, 13.163, 13.974, 14.846, 16.027, 17.053, 18.148, 19.715, 20.907, 22.163, 23.956, 25.235, 26.566, 27.94, 29.576, 30.956, 32.432, 35.337, 39.911, 41.128, 42.625, 44.386,
01300
01301
01302
               46.369, 48.619, 51.031, 53.674, 56.825, 59.921, 63.286, 66.929,
               70.859, 75.081, 79.618, 84.513, 89.739, 95.335, 101.35, 107.76,
01304
01305
               114.63, 121.98, 129.87, 138.3, 147.34, 157.04, 167.56, 178.67,
01306
               190.61, 203.43, 217.19, 231.99, 247.88, 264.98, 283.37, 303.17,
              324.49, 347.47, 372.25, 398.98, 427.85, 459.06, 492.8, 529.31, 568.89, 611.79, 658.35, 708.91, 763.87, 823.65, 888.72, 959.58,
01307
01308
01309
               1036.8, 1121.8, 1213.9, 1314.3, 1423.8, 1543., 1672.8, 1813.4,
               1966.1, 2131.4, 2309.5, 2499.3, 2705., 2925.7, 3161.6, 3411.3,
01310
               3611.5, 3889.2, 4191.1, 4519.3, 4877.9, 5272.9, 5712.9, 6142.7,
01311
               6719.6, 7385., 8145., 8977.7, 9831.9, 10827., 11934., 13063., 14434., 15878., 17591., 19435., 21510., 23835., 26835., 29740.,
01312
01313
               32878., 36305., 39830., 43273., 46931., 50499., 49586., 51598., 53429., 54619., 55081., 55102., 54485., 53487., 52042., 42689.,
01314
01316
               42607., 44020., 47994., 54169., 53916., 55808., 56642., 46049.,
               44243., 32929., 30658., 21963., 20835., 15962., 13679., 17652.,
01317
01318
               19680., 22388., 25625., 29184., 32520., 35720., 38414., 40523.,
               49228., 48173., 45678., 41768., 37600., 41313., 42654., 44465.,
01319
               55736., 56630., 65409., 63308., 66572., 61845., 60379., 56777., 51920., 46601., 41367., 36529., 32219., 28470., 25192., 22362.,
01320
               19907., 17772., 15907., 14273., 12835., 11567., 10445., 9450.2, 8565.1, 7776., 7070.8, 6439.2, 5872.3, 5362.4, 4903., 4488.3,
01323
01324
               4113.4, 3773.8, 3465.8, 3186.1, 2931.7, 2700.1, 2488.8, 2296.,
               2119.8, 1958.6, 1810.9, 1675.6, 1551.4, 1437.3, 1332.4, 1236., 1147.2, 1065.3, 989.86, 920.22, 855.91, 796.48, 741.53, 690.69, 643.62, 600.02, 559.6, 522.13, 487.35, 455.06, 425.08, 397.21,
01325
01326
01327
               371.3, 347.2, 324.78, 303.9, 284.46, 266.34, 249.45, 233.7,
               219.01, 205.3, 192.5, 180.55, 169.38, 158.95, 149.2, 140.07,
01329
01330
               131.54, 123.56, 116.09, 109.09, 102.54, 96.405, 90.655, 85.266,
01331
               80.213, 75.475, 71.031, 66.861, 62.948, 59.275, 55.827, 52.587,
01332
               49.544, 46.686, 43.998, 41.473, 39.099, 36.867, 34.768, 32.795,
01333
               30.939, 29.192, 27.546, 25.998, 24.539, 23.164, 21.869, 20.65,
               19.501, 18.419, 17.399, 16.438, 15.532, 14.678, 13.874, 13.115
               12.4, 11.726, 11.088, 10.488, 9.921, 9.3846, 8.8784, 8.3996,
01335
               7.9469, 7.5197, 7.1174, 6.738, 6.379, 6.0409, 5.7213, 5.419, 5.1327, 4.8611, 4.6046, 4.3617, 4.1316, 3.9138, 3.7077, 3.5125, 3.3281, 3.1536, 2.9885, 2.8323, 2.6846, 2.5447, 2.4124, 2.2871,
01336
01337
01338
               2.1686, 2.0564, 1.9501, 1.8495, 1.7543, 1.6641, 1.5787, 1.4978, 1.4212, 1.3486, 1.2799, 1.2147, 1.1529, 1.0943, 1.0388, .98602,
01339
              .93596, .8886, .84352, .80078, .76029, .722, .68585, .65161, .61901, .58808, .55854, .53044, .5039, .47853, .45459, .43173
01341
01342
01343
               .41008, .38965, .37021, .35186, .33444, .31797, .30234, .28758,
               .2736, .26036, .24764, .2357, .22431, .21342, .20295, .19288, .18334, .17444, .166, .15815, .15072, .14348, .13674, .13015, .12399, .11807, .11231, .10689, .10164, .096696, .091955,
01344
01345
01346
               .087476, .083183, .079113, .075229, .071536, .068026, .064698,
               .06154, .058544, .055699, .052997, .050431, .047993, .045676, .043475, .041382, .039392, .037501, .035702, .033991, .032364,
01348
01349
01350
               .030817, .029345, .027945, .026613, .025345, .024139, .022991,
               .021899, .02086, .019871, .018929, .018033, .01718, .016368, .015595, .014859, .014158, .013491, .012856, .012251, .011675, .011126, .010604, .010107, .0096331, .009182, .0087523, .0083431,
01351
01352
01353
               .0079533, .0075821, .0072284, .0068915, .0065706, .0062649,
01354
01355
               .0059737, .0056963, .005432, .0051802, .0049404, .0047118,
01356
               .0044941, .0042867, .0040891, .0039009, .0037216, .0035507,
               .003388, .0032329, .0030852, .0029445, .0028105, .0026829, .0025613, .0024455, .0023353, .0022303, .0021304, .0020353, .0019448, .0018587, .0017767, .0016988, .0016247, .0015543,
01357
01358
               .0014874, .0014238, .0013635, .0013062, .0012519, .0012005, .0011517, .0011057, .0010621, .001021, 9.8233e-4, 9.4589e-4
01360
01361
               9.1167e-4, 8.7961e-4, 8.4964e-4, 8.2173e-4, 7.9582e-4, 7.7189e-4, 7.499e-4, 7.2983e-4, 7.1167e-4, 6.9542e-4, 6.8108e-4, 6.6866e-4, 6.5819e-4, 6.4971e-4, 6.4328e-4, 6.3895e-4, 6.3681e-4, 6.3697e-4,
01362
01363
01364
```

```
6.3956e-4, 6.4472e-4, 6.5266e-4, 6.6359e-4, 6.778e-4, 6.9563e-4,
                                    7.1749e-4, 7.4392e-4, 7.7556e-4, 8.1028e-4, 8.4994e-4, 8.8709e-4, 9.3413e-4, 9.6953e-4, .0010202, .0010738, .0010976, .0011507,
01366
01367
                                     .0011686, .0012264, .001291, .0013346, .0014246, .0015293,
01368
                                   .0016359, .0017824, .0019255, .0020854, .002247, .0024148, .0026199, .0027523, .0029704, .0030702, .0033047, .0035013, .0037576, .0040275, .0043089, .0046927, .0049307, .0053486, .0053809, .0056699, .0059325, .0055488, .005634, .0056392,
01369
01372
                                   .0043809, .005699, .005925, .005488, .005634, .0056392, .004946, .0048855, .0048208, .0044386, .0045498, .0046377, .0048939, .0052396, .0057324, .0060859, .0066906, .0071148, .0077224, .0082687, .008769, .0084471, .008572, .0087729, .008775, .0090742, .0080704, .0080288, .0085747, .0086087, .0086408, .0088752, .0089381, .0089757, .0093532, .0092824, .0092566, .0092645, .0092735, .009342, .0095806, .0097991, .0080875, .0081300, .0087591, .0080875, .0081300, .0087591, .0081300, .0081300, .0087591, .0081300, .0081300, .0087591, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081300, .0081
01373
01374
01375
01376
01378
01379
                                    .010213, .010611, .011129, .011756, .013237, .01412, .015034,
                                   .015936, .01682, .018597, .019315, .019995, .020658, .021289, .022363, .022996, .023716, .024512, .025434, .026067, .027118, .028396, .029865, .031442, .033253, .03525, .037296, .039701, .042356, .045154, .048059, .051294, .054893, .058636, .061407, .065172, .068974, .072676, .073379, .076547, .079556, .079134, .033253, .03525, .0372956, .079134, .03250, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .03550, .
01380
01381
01382
01384
                                    .082308, .085739, .090192, .09359, .099599, .10669, .11496,
01385
01386
                                     .1244, .13512, .14752, .14494, .15647, .1668, .17863, .19029,
                                     .20124, .20254, .21179, .21982, .21625, .22364, .23405, .23382,
01387
                                   .2434, .25708, .26406, .27621, .28909, .30395, .31717, .33271, .3496, .36765, .38774, .40949, .446, .46985, .49846, .5287, .56
.59841, .64598, .68834, .7327, .78978, .8373, .88708, .94744, .10006, 1.0574, 1.1215, 1.1856, 1.2546, 1.3292, 1.4107, 1.4974,
01388
01389
01391
                                   1.5913, 1.6931, 1.8028, 1.9212, 2.0492, 2.1874, 2.3365, 2.4978, 2.6718, 2.8588, 3.062, 3.2818, 3.5188, 3.7752, 4.0527, 4.3542, 4.6782, 5.0312, 5.4123, 5.8246, 6.2639, 6.7435, 7.2636, 7.8064, 8.4091, 9.0696, 9.7677, 10.548, 11.4, 12.309, 13.324, 14.284, 15.445, 16.687, 18.019, 19.403, 20.847, 22.366, 23.925, 25.537,
01392
01393
01394
01395
                                     27.213, 28.069, 29.864, 31.829, 33.988, 35.856, 38.829, 42.321,
01397
01398
                                     46.319, 50.606, 55.126, 59.126, 64.162, 68.708, 74.615, 81.176,
                                   87.739, 95.494, 103.83, 113.38, 123.99, 135.8, 148.7, 162.58, 176.32, 192.6, 211.47, 232.7, 252.64, 277.41, 305.38, 333.44, 366.42, 402.66, 442.14, 484.53, 526.42, 568.15, 558.78, 582.6, 600.98, 613.94, 619.44, 618.24, 609.84, 595.96, 484.86, 475.59,
01399
01400
01401
01403
                                     478.49, 501.56, 552.19, 628.44, 630.39, 658.92, 671.96, 562.7,
                                    545.88, 423.43, 400.14, 306.59, 294.13, 246.8, 226.51, 278.21,
01404
                                   314.39, 347.22, 389.13, 433.16, 477.48, 521.67, 560.54, 683.6, 696.37, 695.91, 683.1, 658.24, 634.89, 698.85, 742.87, 796.66, 954.49, 1009.5, 1150.5, 1179.1, 1267.9, 1272.4, 1312.7, 1330.4, 1331.6, 1315.8, 1308.3, 1293.3, 1274.6, 1249.5, 1213.2, 1172.1,
01405
01406
01407
                                    1124.4, 930.33, 893.36, 871.27, 883.54, 940.76, 1036., 1025.6,
01410
                                    1053.1, 914.51, 894.15, 865.03, 670.63, 508.41, 475.15, 370.85
                                   361.06, 319.38, 312.75, 331.87, 367.13, 415., 467.94, 525.49, 578.41, 624.66, 794.82, 796.97, 780.29, 736.49, 670.18, 603.75, 659.67, 679.8, 857.12, 884.05, 900.65, 1046.1, 1141.9, 1083., 1089.2, 1e3, 947.08, 872.31, 787.91, 704.75, 624.93, 553.68,
01411
01412
01413
01414
                                     489.91, 434.21, 385.64, 343.3, 306.42, 274.18, 245.94, 221.11,
                                    199.23, 179.88, 162.73, 147.48, 133.88, 121.73, 110.86, 101.1
01416
                                   92.323, 84.417, 77.281, 70.831, 64.991, 59.694, 54.884, 50.509, 46.526, 42.893, 39.58, 36.549, 33.776, 31.236, 28.907, 26.77, 24.805, 23., 21.339, 19.81, 18.404, 17.105, 15.909, 14.801,
01417
01418
01419
                                    13.778, 12.83, 11.954, 11.142, 10.389, 9.691, 9.0434, 8.4423, 7.8842, 7.3657, 6.8838, 6.4357, 6.0189, 5.6308, 5.2696, 4.9332, 4.6198, 4.3277, 4.0553, 3.8012, 3.5639, 3.3424, 3.1355, 2.9422,
01420
01422
                                   4.6198, 4.3277, 4.0553, 3.8012, 3.5639, 3.3424, 3.1355, 2.9422, 2.7614, 2.5924, 2.4343, 2.2864, 2.148, 2.0184, 1.8971, 1.7835, 1.677, 1.5773, 1.4838, 1.3961, 1.3139, 1.2369, 1.1645, 1.0966, 1.0329, .97309, .91686, .86406, .81439, .76767, .72381, .68252, .64359, .60695, .57247, .54008, .50957, .48092, .45401, .42862, .40465, .38202, .36072, .34052, .3216, .30386, .28711, .27135, .25651, .24252, .2293, .21689, .20517, .19416, .18381, .17396,
01423
01424
01425
01426
01428
                                     .16469
01429
01430
01431
                            static double co2230[2001] = \{ 2.743e-5, 2.8815e-5, 3.027e-5, 3.1798e-5, 3.027e-5, 3.1798e-5, 3.027e-5, 
01432
01433
                                 3.3405e-5, 3.5094e-5, 3.6869e-5, 3.8734e-5, 4.0694e-5, 4.2754e-5,
                                     4.492e-5, 4.7196e-5, 4.9588e-5, 5.2103e-5, 5.4747e-5, 5.7525e-5,
                                     6.0446e-5, 6.3516e-5, 6.6744e-5, 7.0137e-5, 7.3704e-5, 7.7455e-5,
01435
01436
                                    8.1397e-5, 8.5543e-5, 8.9901e-5, 9.4484e-5, 9.9302e-5, 1.0437e-4,
01437
                                    1.097e-4, 1.153e-4, 1.2119e-4, 1.2738e-4, 1.3389e-4, 1.4074e-4,
                                    1.4795e-4, 1.5552e-4, 1.6349e-4, 1.7187e-4, 1.8068e-4, 1.8995e-4, 1.997e-4, 2.0996e-4, 2.2075e-4, 2.321e-4, 2.4403e-4, 2.5659e-4, 2.698e-4, 2.837e-4, 2.9832e-4, 3.137e-4, 3.2988e-4, 3.4691e-4,
01438
01439
                                    3.6483e-4, 3.8368e-4, 4.0351e-4, 4.2439e-4, 4.4635e-4, 4.6947e-4,
01441
01442
                                     4.9379e-4, 5.1939e-4, 5.4633e-4, 5.7468e-4, 6.0452e-4,
01443
                                     6.69e-4, 7.038e-4, 7.4043e-4, 7.79e-4, 8.1959e-4, 8.6233e-4,
                                    9.0732e-4, 9.5469e-4, .0010046, .0010571, .0011124, .0011706,
01444
                                   .0012319, .0012964, .0013644, .001436, .0015114, .0015908, .0016745, .0017625, .0018553, .0019531, .002056, .0021645,
01445
                                    .0022788, .0023992, .002526, .0026596, .0028004, .0029488,
01448
                                     .0031052, .0032699, .0034436, .0036265, .0038194, .0040227
01449
                                     .0042369, \ .0044628, \ .0047008, \ .0049518, \ .0052164, \ .0054953,
                                    .0057894, .0060995, .0064265, .0067713, .007135, .0075184, .0079228, .0083494, .0087993, .0092738, .0097745, .010303,
01450
01451
```

```
.01086, .011448, .012068, .012722, .013413, .014142, .014911,
                    .015723, .01658, .017484, .018439, .019447, .020511, .021635, .022821, .024074, .025397, .026794, .02827, .029829, .031475,
01454
01455
                     .033215, .035052, .036994, .039045, .041213, .043504, .045926,
                     .048485, .05119, .05405, .057074, .060271, .063651, .067225, .071006, .075004, .079233, .083708, .088441, .093449, .098749,
01456
                     .10436, .11029, .11657, .12322, .13026, .13772, .14561, .15397,
                     .16282, .1722, .18214, .19266, .20381, .21563, .22816,
01460
                     .2555, .27043, .28625, .30303, .32082, .3397, .35972, .38097,
                     .2333, .27043, .28023, .30303, .32022, .3337, .3372, .38097, .40352, .42746, .45286, .47983, .50847, .53888, .57119, .6055, .64196, .6807, .72187, .76564, .81217, .86165, .91427, .97025, 1.0298, 1.0932, 1.1606, 1.2324, 1.3088, 1.3902, 1.477, 1.5693, 1.6678, 1.7727, 1.8845, 2.0038, 2.131, 2.2666, 2.4114, 2.5659, 2.7309, 2.907, 3.0951, 3.2961, 3.5109, 3.7405, 3.986, 4.2485,
01461
01462
01463
01464
01466
                     4.5293, 4.8299, 5.1516, 5.4961, 5.8651, 6.2605, 6.6842, 7.1385,
                    7.6256, 8.1481, 8.7089, 9.3109, 9.9573, 10.652, 11.398, 12.2, 13.063, 13.992, 14.99, 16.064, 17.222, 18.469, 19.813, 21.263, 22.828, 24.516, 26.34, 28.31, 30.437, 32.738, 35.226, 37.914, 40.824, 43.974, 47.377, 51.061, 55.011, 59.299, 63.961, 69.013,
01467
01468
01469
                     74.492, 80.444, 86.919, 93.836, 101.23, 109.25, 117.98, 127.47,
                     137.81, 149.07, 161.35, 174.75, 189.42, 205.49, 223.02, 242.26,
01472
                    263.45, 286.75, 311.94, 340.01, 370.86, 404.92, 440.44, 480.27, 525.17, 574.71, 626.22, 686.8, 754.38, 827.07, 913.38, 1011.7,
01473
01474
                     1121.5, 1161.6, 1289.5, 1432.2, 1595.4, 1777., 1983.3, 2216.1, 2485.7, 2788.3, 3101.5, 3481., 3902.1, 4257.1, 4740., 5272.8,
01475
                     5457.9, 5946.2, 6505.3, 6668.4, 7302.4, 8061.6, 9015.8, 9908.3, 11613., 13956., 3249.6, 3243., 2901.5, 2841.3, 2729.6, 2558.2,
01478
01479
                     1797.8, 1583.2, 1386., 1233.5, 787.74, 701.46, 761.66, 767.21,
01480
                     722.83, 1180.6, 1332.1, 1461.6, 2032.9, 2166., 2255.9, 2294.7,
                     2587.2, 2396.5, 2122.4, 12553., 10784., 9832.5, 8827.3, 8029.1, 7377.9, 7347.1, 6783.8, 6239.1, 5721.1, 5503., 4975.1, 4477.8,
01481
01482
                     4021.3, 3676.8, 3275.3, 2914.9, 2597.4, 2328.2, 2075.4, 1857.6,
                     1663.6, 1493.3, 1343.8, 1213.3, 1095.6, 1066.5, 958.91, 865.15,
01484
01485
                     783.31, 714.35, 650.77, 593.98, 546.2, 499.9, 457.87, 421.75,
                     387.61, 355.25, 326.62, 299.7, 275.21, 253.17, 232.83, 214.31, 197.5, 182.08, 167.98, 155.12, 143.32, 132.5, 122.58, 113.48,
01486
01487
                     105.11, 97.415, 90.182, 83.463, 77.281, 71.587, 66.341, 61.493,
01488
                     57.014, 53.062, 49.21, 45.663, 42.38, 39.348, 36.547, 33.967,
01490
                     31.573, 29.357, 27.314, 25.415, 23.658, 22.03, 20.524, 19.125,
                     17.829, 16.627, 15.511, 14.476, 13.514, 12.618, 11.786, 11.013
01491
01492
                     10.294, 9.6246, 9.0018, 8.4218, 7.8816, 7.3783, 6.9092, 6.4719,
                     6.0641, 5.6838, 5.3289, 4.998, 4.6893, 4.4014, 4.1325, 3.8813, 3.6469, 3.4283, 3.2241, 3.035, 2.8576, 2.6922, 2.5348, 2.3896, 2.2535, 2.1258, 2.0059, 1.8929, 1.7862, 1.6854, 1.5898, 1.4992,
01493
01494
01495
                     1.4017, 1.3218, 1.2479, 1.1809, 1.1215, 1.0693, 1.0116, .96016,
01497
                     .9105, .84859, .80105, .74381, .69982, .65127, .60899, .57843,
01498
                     .54592, .51792, .49336, .47155, .45201, .43426, .41807, .40303,
                    01499
01500
01501
01503
01504
                     .092554, .074093, .062159, .055523, .054849, .05401, .05528,
                    .058982, .07952, .08647, .093244, .099285, .10393, .10661, .12072, .11417, .10396, .093265, .089137, .088909, .10902, .11277, .13625, .13565, .14907, .14167, .1428, .13744, .127 .11382, .10244, .091686, .08109, .071739, .063616, .056579,
01505
01506
01507
                     .050504, .045251, .040689, .036715, .033237, .030181, .027488,
01509
                     .025107, .022998, .021125, .01946, .017979, .016661, .015489,
01510
01511
                     .014448, .013526, .012712, .011998, .011375, .010839, .010384,
                    .014448, .013526, .012712, .011998, .011375, .010839, .010384, .010007, .0097053, .0094783, .0093257, .0092489, .0092504, .0093346, .0095077, .0097676, .01012, .01058, .011157, .011844, .012672, .013665, .014766, .015999, .017509, .018972, .020444, .022311, .023742, .0249, .025599, .026981, .026462, .025143, .025066, .022814, .020458, .020026, .019142, .020189, .022371,
01512
01513
01516
01517
                     .024163, .023728, .02199, .019506, .018591, .015576, .012784,
                    .011744, .0094777, .0079148, .0070652, .006986, .0071758, .008086, .0098025, .01087, .013609, .016764, .018137, .021061, .023498, .023576, .023965, .022828, .021519, .021283, .023364, .026457, .029782, .030856, .033486, .035515, .035543, .036558, .036588, .036588, .036588, .036588, .036588, .038564, .038564, .038658, .038668, .038668, .038668, .038668, .038668, .038668, .038668, .038668, .038668, .038668, .038668, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038688, .038888, .038888, .038888, .038888, .038888, .038888, .038888, .038888, .038888, .038888, .038888, .038888, .038888, .038888, .038888, .038888, .038888, .038888, .038888, .038888, .038888, .038888, .038888, .038888, .038888, .038888, .038888, .038888, .038888, .038888, .038888, .038888, .038888, .038888, .038888, .038888, .0388888, .038888, .038888, .038888, .038888, .038888, .038888, .0388888, .038888, .038888, .038888, .0388888, .0388888, .038888, .038
01518
01519
01520
                     .037198, .037472, .037045, .037284, .03777, .038085, .038366, .038526, .038282, .038915, .037697, .035667, .032941, .031959,
01522
01523
01524
                     .028692, .025918, .024596, .025592, .027873, .028935, .02984,
                     .028148, .025305, .021912, .020454, .016732, .013357, .01205,
01525
                     .009731, .0079881, .0077704, .0074387, .0083895, .0096776, .010326, .01293, .015955, .019247, .020145, .02267, .024231, .024184, .022131, .019784, .01955, .01971, .022119, .025116,
01526
01528
                     .027978, .028107, .029808, .030701, .029164, .028551, .027286, .024946, .023259, .020982, .019221, .017471, .015643, .014074,
01529
01530
                     .01261, .011301, .010116, .0090582, .0081036, .0072542, .0065034, .0058436, .0052571, .0047321, .0042697, .0038607, .0034977, .0031747, .0028864, .0026284, .002397, .002189, .0020017,
01531
01532
                     .0018326, .0016798, .0015414, .0014159, .0013019, .0011983,
01535
                      .0011039, .0010177, 9.391e-4, 8.6717e-4, 8.0131e-4, 7.4093e-4,
01536
                     6.8553e-4, 6.3464e-4, 5.8787e-4, 5.4487e-4, 5.0533e-4, 4.69e-4,
                     4.3556e-4, 4.0474e-4, 3.7629e-4, 3.5e-4, 3.2569e-4, 3.032e-4, 2.8239e-4, 2.6314e-4, 2.4535e-4, 2.2891e-4, 2.1374e-4, 1.9975e-4,
01537
```

```
1.8685e-4, 1.7498e-4, 1.6406e-4, 1.5401e-4, 1.4479e-4, 1.3633e-4,
                     1.2858e-4, 1.2148e-4, 1.1499e-4, 1.0907e-4, 1.0369e-4, 9.8791e-5, 9.4359e-5, 9.0359e-5, 8.6766e-5, 8.3555e-5, 8.0703e-5, 7.8192e-5,
01540
01541
01542
                     7.6003e-5, 7.4119e-5, 7.2528e-5, 7.1216e-5, 7.0171e-5, 6.9385e-5,
                     6.8848e-5, 6.8554e-5, 6.8496e-5, 6.8669e-5, 6.9069e-5, 6.9694e-5, 7.054e-5, 7.1608e-5, 7.2896e-5, 7.4406e-5, 7.6139e-5, 7.8097e-5,
01543
01544
                      8.0283e-5, 8.2702e-5, 8.5357e-5, 8.8255e-5, 9.1402e-5, 9.4806e-5
                     9.8473e-5, 1.0241e-4, 1.0664e-4, 1.1115e-4, 1.1598e-4, 1.2112e-4,
01546
01547
                     1.2659e-4, 1.3241e-4, 1.3859e-4, 1.4515e-4, 1.521e-4, 1.5947e-4,
01548
                     1.6728e-4, 1.7555e-4, 1.8429e-4, 1.9355e-4, 2.0334e-4, 2.1369e-4,
                     2.2463e-4, 2.3619e-4, 2.4841e-4, 2.6132e-4, 2.7497e-4, 2.8938e-4,
01549
                     3.0462e-4, 3.2071e-4, 3.3771e-4, 3.5567e-4, 3.7465e-4, 3.947e-4, 4.1588e-4, 4.3828e-4, 4.6194e-4, 4.8695e-4, 5.1338e-4, 5.4133e-4, 5.7087e-4, 6.0211e-4, 6.3515e-4, 6.701e-4, 7.0706e-4, 7.4617e-4,
01550
01552
01553
                     7.8756e-4, 8.3136e-4, 8.7772e-4, 9.2681e-4, 9.788e-4, .0010339,
                     .0010922, .001154, .0012195, .0012889, .0013626, .0014407, .0015235, .0016114, .0017048, .0018038, .001909, .0020207,
01554
01555
                     .0021395, .0022657, .0023998, .0025426, .0026944, .002856, .0030281, .0032114, .0034068, .003615, .0038371, .004074, .004327, .0045971, .0048857, .0051942, .0055239, .0058766, .0062538, .0066573, .0070891, .007551, .0080455, .0085747, .0091412, .0097481, .010397, .011092, .011837, .012638, .013495,
01556
01559
01560
                     .019412, .0097461, .010397, .011092, .011637, .012636, .0134
.014415, .01541, .016475, .017621, .018857, .020175, .02162,
.023185, .024876, .02672, .028732, .030916, .033319, .035939,
.038736, .041847, .04524, .048715, .052678, .056977, .061203,
.066184, .07164, .076952, .083477, .090674, .098049, .10697,
01561
01562
01563
                     .066184, .0/164, .0/6952, .0834//, .0906/4, .098049, .1069/, .1169, .1277, .14011, .15323, .1684, .18601, .20626, .22831, .25417, .28407, .31405, .34957, .38823, .41923, .46026, .50409, .51227, .54805, .57976, .53818, .55056, .557, .46741, .46403, .4636, .42265, .45166, .49852, .56663, .34306, .17779, .17697, .18346, .19129, .20014, .21778, .23604, .25649, .28676, .31238, .33856, .39998, .4288, .46568, .56654, .60786, .64473, .76466, .7897, .80778, .86443, .85736, .84798, .84157, 1.1385, 1.2446, .7897, .80778, .86443, .85736, .84798, .84157, 1.1385, 1.2446, .7897, .80778, .86443, .85736, .84798, .84157, 1.1385, 1.2446, .7897, .80778, .86443, .85736, .84798, .84157, 1.1385, 1.2446, .7897, .80778, .86443, .85736, .84798, .84157, 1.1385, 1.2446, .7897, .80778, .86443, .85736, .84798, .84157, 1.1385, 1.2446, .7897, .80778, .86443, .85736, .84798, .84157, 1.1385, 1.2446, .7897, .86786, .7897, .80778, .86443, .85736, .84798, .84157, 1.1385, 1.2446, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .7897, .789
01565
01566
01567
01568
01569
01571
01572
                     1.1923, 1.1552, 1.1338, 1.1266, 1.1292, 1.1431, 1.1683, 1.2059
                     1.2521, 1.3069, 1.3712, 1.4471, 1.5275, 1.6165, 1.7145, 1.8189, 1.9359, 2.065, 2.2007, 2.3591, 2.5362, 2.7346, 2.9515, 3.2021, 3.4851, 3.7935, 4.0694, 4.4463, 4.807, 5.2443, 5.7178, 6.2231, 6.4796, 6.9461, 7.4099, 7.3652, 7.7182, 8.048, 7.7373, 8.0363,
01573
01574
01575
01577
                     8.3855, 8.8044, 9.0257, 9.8574, 10.948, 10.563, 6.8979, 7.0744,
                     7.4121, 7.7663, 8.1768, 8.6243, 9.1437, 9.7847, 10.182, 10.849, 11.572, 12.602, 13.482, 14.431, 15.907, 16.983, 18.11, 19.884, 21.02, 22.18, 23.355, 24.848, 25.954, 27.13, 30.186, 34.893,
01578
01579
01580
                     35.682, 36.755, 38.111, 39.703, 41.58, 43.606, 45.868, 48.573, 51.298, 54.291, 57.559, 61.116, 64.964, 69.124, 73.628, 78.471,
01581
                     83.683, 89.307, 95.341, 101.84, 108.83, 116.36, 124.46, 133.18,
01584
                     142.57, 152.79, 163.69, 175.43, 188.11, 201.79, 216.55, 232.51,
                     249.74, 268.38, 288.54, 310.35, 333.97, 359.55, 387.26, 417.3, 449.88, 485.2, 523.54, 565.14, 610.28, 659.31, 712.56, 770.43, 833.36, 901.82, 976.36, 1057.6, 1146.8, 1243.8, 1350., 1466.3, 1593.6, 1732.7, 1884.1, 2049.1, 2228.2, 2421.9, 2629.4, 2853.7,
01585
01586
01587
01588
                      3094.4, 3351.1, 3622.3, 3829.8, 4123.1, 4438.3, 4777.2, 5144.1,
                     5545.4, 5990.5, 6404.5, 6996.8, 7687.6, 8482.9, 9349.4, 10203.,
01590
01591
                     11223., 12358., 13493., 14916., 16416., 18236., 20222., 22501.,
01592
                     25102., 28358., 31707., 35404., 39538., 43911., 48391., 53193.,
                     58028., 58082., 61276., 64193., 66294., 67480., 67921., 67423.,
01593
01594
                     66254., 64341., 51737., 51420., 53072., 58145., 66195., 65358.,
                     67377., 67869., 53509., 50553., 35737., 32425., 21704., 19974., 14457., 12142., 16798., 19489., 23049., 27270., 31910., 36457.,
01596
01597
                      40877., 44748., 47876., 59793., 58626., 55454., 50337., 44893.,
01598
                     50228., 52216., 54747., 69541., 70455., 81014., 77694., 80533.,
                      73953., 70927., 65539., 59002., 52281., 45953., 40292., 35360.,
01599
                     31124., 27478., 24346., 21647., 19308., 17271., 15491., 13927., 12550., 11331., 10250., 9288.8, 8431.4, 7664.9, 6978.3, 6361.8,
01600
                     5807.4, 5307.7, 4856.8, 4449., 4079.8, 3744.9, 3440.8, 3164.2, 2912.3, 2682.7, 2473., 2281.4, 2106., 1945.3, 1797.9, 1662.5,
01602
01603
                     1538.1, 1423.6, 1318.1, 1221., 1131.5, 1049., 972.99, 902.87, 838.01, 777.95, 722.2, 670.44, 622.35, 577.68, 536.21, 497.76, 462.12, 429.13, 398.61, 370.39, 344.29, 320.16, 297.85, 277.2, 258.08, 240.38, 223.97, 208.77, 194.66, 181.58, 169.43, 158.15,
01604
01605
01606
01607
                      147.67, 137.92, 128.86, 120.44, 112.6, 105.3, 98.499, 92.166,
                     86.264, 80.763, 75.632, 70.846, 66.381, 62.213, 58.321, 54.685,
01609
01610
                     51.288, 48.114, 45.145, 42.368, 39.772, 37.341, 35.065, 32.937,
01611
                     30.943, 29.077, 27.33, 25.693, 24.158, 22.717, 21.367, 20.099,
                     18.909, 17.792, 16.744, 15.761, 14.838, 13.971, 13.157, 12.393, 11.676, 11.003, 10.369, 9.775, 9.2165, 8.6902, 8.1963, 7.7314, 7.2923, 6.8794, 6.4898, 6.122, 5.7764, 5.4525, 5.1484, 4.8611,
01612
01613
01614
                     4.5918, 4.3379, 4.0982, 3.8716, 3.6567, 3.4545, 3.2634, 3.0828,
01615
01616
                     2.9122, 2.7512, 2.5993, 2.4561, 2.3211, 2.1938, 2.0737, 1.9603,
01617
                     1.8534, 1.7525, 1.6572, 1.5673, 1.4824, 1.4022, 1.3265, 1.2551,
                     1.1876, 1.1239, 1.0637, 1.0069, .9532, .90248, .85454, .80921,
01618
                     .76631, .72569, .6872, .65072, .61635, .5836, .55261, .52336, .49581, .46998, .44559, .42236, .40036, .37929, .35924, .34043,
01619
                     .32238, .30547, .28931, .27405, .25975, .24616, .23341, .22133,
01621
01622
                      .20997, .19924, .18917, .17967, .17075, .16211, .15411, .14646
                     .13912, .13201, .12509, .11857, .11261, .10698, .10186, .097039, .092236, .087844, .083443, .07938, .075452, .071564, .067931, .064389, .061078, .057901, .054921, .052061, .049364, .046789,
01623
01624
01625
```

```
.04435, .042044, .039866, .037808, .035863, .034023, .032282,
                .030634, .029073, .027595, .026194, .024866, .023608, .022415, .021283, .02021, .019193, .018228, .017312, .016443, .015619,
01627
01628
               .014837, .014094, .01339, .012721, .012086, .011483, .010911, .010368, .009852, .0093623, .0088972, .0084556, .0080362, .0076379, .0072596, .0069003, .006559, .0062349, .0059269, .0056344, .0053565, .0050925, .0048417, .0046034, .004377, .0041618, .0039575, .0037633, .0035788, .0034034, .0032368,
01629
01630
01631
01632
                .0030785, .002928, .0027851, .0026492, .0025201, .0023975, .0022809, .0021701, .0020649, .0019649, .0018699, .0017796,
01634
01635
                .0016938, .0016122, .0015348, .0014612, .0013913, .001325, .0012619, .0012021, .0011452, .0010913, .0010401, 9.9149e-4, 9.454e-4, 9.0169e-4, 8.6024e-4, 8.2097e-4, 7.8377e-4, 7.4854e-4,
01636
01637
01638
                7.1522e-4, 6.8371e-4, 6.5393e-4, 6.2582e-4, 5.9932e-4, 5.7435e-4
01639
01640
                5.5087e-4, 5.2882e-4, 5.0814e-4, 4.8881e-4, 4.7076e-4, 4.5398e-4,
                4.3843e-4, 4.2407e-4, 4.109e-4, 3.9888e-4, 3.88e-4, 3.7826e-4, 3.6963e-4, 3.6213e-4, 3.5575e-4, 3.505e-4, 3.464e-4, 3.4346e-4,
01641
01642
                3.4173e-4, 3.4125e-4, 3.4206e-4, 3.4424e-4, 3.4787e-4, 3.5303e-4,
01643
01644
                3.5986e-4, 3.6847e-4, 3.7903e-4, 3.9174e-4, 4.0681e-4, 4.2455e-4,
                4.4527e-4, 4.6942e-4, 4.9637e-4, 5.2698e-4, 5.5808e-4, 5.9514e-4, 6.2757e-4, 6.689e-4, 7.1298e-4, 7.3955e-4, 7.8403e-4, 8.0449e-4,
01646
01647
                8.5131e-4, 9.0256e-4, 9.3692e-4, .0010051, .0010846, .0011678,
                .001282, .0014016, .0015355, .0016764, .0018272, .0020055,
01648
               .0021455, .0023421, .0024615, .0026786, .0028787, .0031259, .0034046, .0036985, .0040917, .0043902, .0048349, .0049531, .0052989, .0056148, .0052452, .0053357, .005333, .0045069,
01649
01650
01651
                .0043851, .004253, .003738, .0038084, .0039013, .0041505
01652
                .0045372, .0050569, .0054507, .0061267, .0066122, .0072449,
01653
01654
                .0078012, .0082651, .0076538, .0076573, .0076806, .0075227,
                .0076269, .0063758, .006254, .0067749, .0067909, .0068231, .0072143, .0072762, .0072954, .007679, .0075107, .0073658, .0072441, .0071074, .0070378, .007176, .0072472, .0075844,
01655
01656
                .0079291, .008412, .0090165, .010688, .011535, .012375, .013166,
01658
01659
                .013895, .015567, .016011, .016392, .016737, .017043, .017731,
                .018031, .018419, .018877, .019474, .019868, .020604, .021538, .022653, .023869, .025288, .026879, .028547, .030524, .03274,
01660
01661
                .035132, .03769, .040567, .043793, .047188, .049962, .053542, .057205, .060776, .061489, .064419, .067124, .065945, .068487, .071209, .074783, .077039, .082444, .08902, .09692, .10617,
01662
01663
                . 11687, .12952, .12362, .13498, .14412, .15492, .16519, .1744, .17096, .17714, .18208, .17363, .17813, .18564, .18295, .19045,
01665
01666
                .20252, .20815, .21844, .22929, .24229, .25321, .26588, .2797,
01667
                .29465, .31136, .32961, .36529, .38486, .41027, .43694, .4667, .49943, .54542, .58348, .62303, .67633, .71755, .76054, .81371, .85934, .90841, .96438, 1.0207, 1.0821, 1.1491, 1.2226, 1.3018, 1.388, 1.4818, 1.5835, 1.6939, 1.8137, 1.9435, 2.0843, 2.237,
01668
01669
01670
01671
01672
                2.4026, 2.5818, 2.7767, 2.9885, 3.2182, 3.4679, 3.7391, 4.0349,
               2.3016, 2.3016, 2.3016, 2.3016, 3.2102, 3.4017, 3.3017, 4.0317, 4.3554, 4.7053, 5.0849, 5.4986, 5.9436, 6.4294, 6.9598, 7.5203, 8.143, 8.8253, 9.5568, 10.371, 11.267, 12.233, 13.31, 14.357, 15.598, 16.93, 18.358, 19.849, 21.408, 23.04, 24.706, 26.409, 28.153, 28.795, 30.549, 32.43, 34.49, 36.027, 38.955, 42.465,
01673
01674
01675
                46.565, 50.875, 55.378, 59.002, 63.882, 67.949, 73.693, 80.095,
01677
01678
                86.403, 94.264, 102.65, 112.37, 123.3, 135.54, 149.14, 163.83,
                179.17, 196.89, 217.91, 240.94, 264.13, 292.39, 324.83, 358.21, 397.16, 440.5, 488.6, 541.04, 595.3, 650.43, 652.03, 688.74,
01679
01680
                719.47, 743.54, 757.68, 762.35, 756.43, 741.42, 595.43, 580.97, 580.83, 605.68, 667.88, 764.49, 759.93, 789.12, 798.17, 645.66,
01681
                615.65, 455.05, 421.09, 306.45, 289.14, 235.7, 215.52, 274.57,
01683
                316.53, 357.73, 409.89, 465.06, 521.84, 579.02, 630.64, 794.46, 813., 813.56, 796.25, 761.57, 727.97, 812.14, 866.75, 932.5, 1132.8, 1194.8, 1362.2, 1387.2, 1482.3, 1479.7, 1517.9, 1533.1,
01684
01685
01686
                1534.2, 1523.3, 1522.5, 1515.5, 1505.2, 1486.5, 1454., 1412., 1358.8, 1107.8, 1060.9, 1033.5, 1048.2, 1122.4, 1248.9, 1227.1,
01687
                1255.4, 1058.9, 1020.7, 970.59, 715.24, 512.56, 468.47, 349.3,
                338.26, 299.22, 301.26, 332.38, 382.08, 445.49, 515.87, 590.85,
01690
01691
                662.3, 726.05, 955.59, 964.11, 945.17, 891.48, 807.11, 720.9,
01692
                803.36, 834.46, 1073.9, 1107.1, 1123.6, 1296., 1393.7, 1303.1, 1284.3, 1161.8, 1078.8, 976.13, 868.72, 767.4, 674.72, 593.73,
01693
                523.12, 462.24, 409.75, 364.34, 325., 290.73, 260.76, 234.46,
01694
                211.28, 190.78, 172.61, 156.44, 142.01, 129.12, 117.57, 107.2, 97.877, 89.47, 81.882, 75.021, 68.807, 63.171, 58.052, 53.396,
01696
01697
                49.155, 45.288, 41.759, 38.531, 35.576, 32.868, 30.384, 28.102
01698
                26.003, 24.071, 22.293, 20.655, 19.147, 17.756, 16.476, 15.292,
                14.198, 13.183, 12.241, 11.367, 10.554, 9.7989, 9.0978, 8.4475, 7.845, 7.2868, 6.7704, 6.2927, 5.8508, 5.4421, 5.064, 4.714, 4.3902, 4.0902, 3.8121, 3.5543, 3.315, 3.093, 2.8869, 2.6953,
01699
01700
01701
                2.5172, 2.3517, 2.1977, 2.0544, 1.9211, 1.7969, 1.6812, 1.5735,
01702
01703
                1.4731, 1.3794, 1.2921, 1.2107, 1.1346, 1.0637, .99744, .93554,
01704
                .87771, .82368, .77313, .72587, .6816, .64014, .60134, .565,
                .53086, .49883, .46881, .44074, .4144, .38979, .36679, .34513,
01705
                .32474, .30552, .28751, .27045, .25458, .23976, .22584, .21278, .20051, .18899, .17815, .16801, .15846, .14954, .14117, .13328,
01706
01708
01709
01710
            double xw, dw, ew, cw296, cw260, cw230, dt230, dt260, dt296, ctw, ctmpth;
01711
01712
```

```
01713
         int iw:
01714
01715
         /* Get CO2 continuum absorption... */
         xw = nu / 2 + 1;

if (xw >= 1 && xw < 2001) {
01716
01717
           iw = (int) xw;
01718
           dw = xw - iw;
01719
            ew = 1 - dw;
01720
           cw296 = ew * co2296[iw - 1] + dw * co2296[iw];
cw260 = ew * co2260[iw - 1] + dw * co2260[iw];
01721
01722
            cw230 = ew * co2230[iw - 1] + dw * co2230[iw];
01723
01724
           dt230 = t - 230;
           dt260 = t - 260;
01725
01726
           dt296 = t - 296;
01727
           ctw = dt260 * 5.050505e-4 * dt296 * cw230 - dt230 * 9.259259e-4
           * dt296 * cw260 + dt230 * 4.208754e-4 * dt260 * cw296;
ctmpth = u / NA / 1000 * p / P0 * ctw;
01728
01729
01730
         } else
01731
           ctmpth = 0;
01732
         return ctmpth;
01733 }
```

5.5.2.7 double ctmh2o (double nu, double p, double t, double q, double u)

Compute water vapor continuum (optical depth).

Definition at line 1737 of file jurassic.c.

```
01742
01743
01744
         static double h2o296[2001] = { .17, .1695, .172, .168, .1687, .1624, .1606,
01745
           .1508, .1447, .1344, .1214, .1133, .1009, .09217, .08297, .06989,
            .06513, .05469, .05056, .04417, .03779, .03484, .02994, .0272,
01746
                                                   .01405.
01747
            .02325, .02063, .01818,
                                         .01592.
                                                             .01251.
                                                                        .0108.
           .008424, .007519, .006555, .00588, .005136, .004511, .003989, .003509, .003114, .00274, .002446, .002144, .001895, .001676,
01748
01749
01750
             .001486, .001312, .001164, .001031, 9.129e-4, 8.106e-4, 7.213e-4,
01751
            6.4e-4, 5.687e-4, 5.063e-4, 4.511e-4, 4.029e-4, 3.596e-4,
           3.22e-4, 2.889e-4, 2.597e-4, 2.337e-4, 2.108e-4, 1.907e-4, 1.728e-4, 1.57e-4, 1.43e-4, 1.305e-4, 1.195e-4, 1.097e-4,
01752
01753
            1.009e-4, 9.307e-5, 8.604e-5, 7.971e-5, 7.407e-5, 6.896e-5,
01754
            6.433e-5, 6.013e-5, 5.631e-5, 5.283e-5, 4.963e-5, 4.669e-5,
01756
            4.398e-5, 4.148e-5, 3.917e-5, 3.702e-5, 3.502e-5, 3.316e-5,
01757
            3.142e-5, 2.978e-5, 2.825e-5, 2.681e-5, 2.546e-5, 2.419e-5,
01758
            2.299e-5, 2.186e-5, 2.079e-5, 1.979e-5, 1.884e-5, 1.795e-5,
            1.711e-5, 1.633e-5, 1.559e-5, 1.49e-5, 1.426e-5, 1.367e-5,
01759
01760
            1.312e-5, 1.263e-5, 1.218e-5, 1.178e-5, 1.143e-5, 1.112e-5,
            1.088e-5, 1.07e-5, 1.057e-5, 1.05e-5, 1.051e-5, 1.059e-5, 1.076e-5, 1.1e-5, 1.133e-5, 1.18e-5, 1.237e-5, 1.308e-5,
01761
01762
01763
            1.393e-5, 1.483e-5, 1.614e-5, 1.758e-5, 1.93e-5, 2.123e-5,
01764
            2.346e-5, 2.647e-5, 2.93e-5, 3.279e-5, 3.745e-5, 4.152e-5,
            4.813e-5, 5.477e-5, 6.203e-5, 7.331e-5, 8.056e-5, 9.882e-5, 1.05e-4, 1.21e-4, 1.341e-4, 1.572e-4, 1.698e-4, 1.968e-4,
01765
01766
            2.175e-4, 2.431e-4, 2.735e-4, 2.867e-4, 3.19e-4, 3.371e-4,
01768
            3.554e-4, 3.726e-4, 3.837e-4, 3.878e-4, 3.864e-4, 3.858e-4,
01769
            3.841e-4, 3.852e-4, 3.815e-4, 3.762e-4, 3.618e-4, 3.579e-4,
01770
            3.45e-4, 3.202e-4, 3.018e-4, 2.785e-4, 2.602e-4, 2.416e-4,
01771
            2.097e-4, 1.939e-4, 1.689e-4, 1.498e-4, 1.308e-4, 1.17e-4,
01772
            1.011e-4, 9.237e-5, 7.909e-5, 7.006e-5, 6.112e-5, 5.401e-5,
01773
            4.914e-5, 4.266e-5, 3.963e-5, 3.316e-5, 3.037e-5, 2.598e-5,
01774
            2.294e-5, 2.066e-5, 1.813e-5, 1.583e-5, 1.423e-5, 1.247e-5,
01775
            1.116e-5, 9.76e-6, 8.596e-6, 7.72e-6, 6.825e-6, 6.108e-6,
01776
            5.366e-6, 4.733e-6, 4.229e-6, 3.731e-6, 3.346e-6, 2.972e-6,
            2.628e-6, 2.356e-6, 2.102e-6, 1.878e-6, 1.678e-6, 1.507e-6, 1.348e-6, 1.21e-6, 1.089e-6, 9.806e-7, 8.857e-7, 8.004e-7, 7.261e-7, 6.599e-7, 6.005e-7, 5.479e-7, 5.011e-7, 4.595e-7,
01777
01778
01779
            4.219e-7, 3.885e-7, 3.583e-7, 3.314e-7, 3.071e-7, 2.852e-7, 2.654e-7, 2.474e-7, 2.311e-7, 2.162e-7, 2.026e-7, 1.902e-7,
01780
01781
01782
            1.788e-7, 1.683e-7, 1.587e-7, 1.497e-7, 1.415e-7,
                                                                         1.338e-7
01783
            1.266e-7, 1.2e-7, 1.138e-7, 1.08e-7, 1.027e-7, 9.764e-8,
            9.296e-8, 8.862e-8, 8.458e-8, 8.087e-8, 7.744e-8, 7.429e-8,
01784
            7.145e-8, 6.893e-8, 6.664e-8, 6.468e-8, 6.322e-8, 6.162e-8, 6.07e-8, 5.992e-8, 5.913e-8, 5.841e-8, 5.796e-8, 5.757e-8,
01785
01787
            5.746e-8, 5.731e-8, 5.679e-8, 5.577e-8, 5.671e-8, 5.656e-8,
01788
            5.594e-8, 5.593e-8, 5.602e-8, 5.62e-8, 5.693e-8, 5.725e-8,
01789
            5.858e-8, 6.037e-8, 6.249e-8, 6.535e-8, 6.899e-8, 7.356e-8,
            7.918e-8, 8.618e-8, 9.385e-8, 1.039e-7, 1.158e-7, 1.29e-7, 1.437e-7, 1.65e-7, 1.871e-7, 2.121e-7, 2.427e-7, 2.773e-7, 3.247e-7, 3.677e-7, 4.037e-7, 4.776e-7, 5.101e-7, 6.214e-7,
01790
01791
01792
            6.936e-7, 7.581e-7, 8.486e-7, 9.355e-7, 9.942e-7, 1.063e-6,
```

```
1.123e-6, 1.191e-6, 1.215e-6, 1.247e-6, 1.26e-6, 1.271e-6,
              1.284e-6, 1.317e-6, 1.323e-6, 1.349e-6, 1.353e-6, 1.362e-6, 1.344e-6, 1.329e-6, 1.336e-6, 1.327e-6, 1.325e-6, 1.359e-6,
01795
01796
01797
              1.374e-6, 1.415e-6, 1.462e-6, 1.526e-6, 1.619e-6, 1.735e-6,
01798
              1.863e-6, 2.034e-6, 2.265e-6, 2.482e-6, 2.756e-6, 3.103e-6,
              3.466e-6, 3.832e-6, 4.378e-6, 4.913e-6, 5.651e-6, 6.311e-6,
01799
              7.169e-6, 8.057e-6, 9.253e-6, 1.047e-5, 1.212e-5, 1.36e-5,
              1.569e-5, 1.776e-5, 2.02e-5, 2.281e-5, 2.683e-5, 2.994e-5,
01801
01802
              3.488e-5, 3.896e-5, 4.499e-5, 5.175e-5, 6.035e-5, 6.34e-5
01803
              7.281e-5, 7.923e-5, 8.348e-5, 9.631e-5, 1.044e-4, 1.102e-4,
              1.176e-4, 1.244e-4, 1.283e-4, 1.326e-4, 1.4e-4, 1.395e-4,
01804
              1.387e-4, 1.363e-4, 1.314e-4, 1.241e-4, 1.228e-4, 1.148e-4,
01805
              1.086e-4, 1.018e-4, 8.89e-5, 8.316e-5, 7.292e-5, 6.452e-5, 5.625e-5, 5.045e-5, 4.38e-5, 3.762e-5, 3.29e-5, 2.836e-5,
01806
01807
01808
              2.485e-5, 2.168e-5, 1.895e-5, 1.659e-5, 1.453e-5, 1.282e-5,
01809
              1.132e-5, 1.001e-5, 8.836e-6, 7.804e-6, 6.922e-6, 6.116e-6,
01810
              5.429e-6, 4.824e-6, 4.278e-6, 3.788e-6, 3.371e-6, 2.985e-6,
              2.649e-6, 2.357e-6, 2.09e-6, 1.858e-6, 1.647e-6, 1.462e-6, 1.299e-6, 1.155e-6, 1.028e-6, 9.142e-7, 8.132e-7, 7.246e-7,
01811
01812
01813
              6.451e-7, 5.764e-7, 5.151e-7, 4.603e-7, 4.121e-7, 3.694e-7,
             3.318e-7, 2.985e-7, 2.69e-7, 2.428e-7, 2.197e-7, 1.992e-7, 1.81e-7, 1.649e-7, 1.506e-7, 1.378e-7, 1.265e-7, 1.163e-7,
01814
01815
              1.073e-7, 9.918e-8, 9.191e-8, 8.538e-8, 7.949e-8, 7.419e-8,
01816
              1.03e-7, 9.316e-8, 9.13fe-8, 5.36e-8, 7.345e-8, 7.345e-8, 6.508e-8, 6.114e-8, 5.761e-8, 5.437e-8, 5.146e-8, 4.89e-8, 4.636e-8, 4.406e-8, 4.201e-8, 4.015e-8, 3.84e-8,
01817
01818
              3.661e-8, 3.51e-8, 3.377e-8, 3.242e-8, 3.13e-8, 3.015e-8,
              2.918e-8, 2.83e-8, 2.758e-8, 2.707e-8, 2.656e-8, 2.619e-8
01820
01821
              2.609e-8, 2.615e-8, 2.63e-8, 2.675e-8, 2.745e-8, 2.842e-8,
             2.066e-8, 3.125e-8, 3.318e-8, 3.565e-8, 3.85e-8, 4.191e-8, 4.59e-8, 5.059e-8, 5.607e-8, 6.239e-8, 6.958e-8, 7.796e-8, 8.773e-8, 9.88e-8, 1.114e-7, 1.258e-7, 1.422e-7, 1.61e-7, 1.822e-7, 2.06e-7, 2.337e-7, 2.645e-7, 2.996e-7, 3.393e-7, 3.843e-7, 4.363e-7, 4.935e-7, 5.607e-7, 6.363e-7, 7.242e-7,
01822
01823
01824
01825
01826
01827
              8.23e-7, 9.411e-7, 1.071e-6, 1.232e-6, 1.402e-6, 1.6e-6, 1.82e-6,
              2.128e-6, 2.386e-6, 2.781e-6, 3.242e-6, 3.653e-6, 4.323e-6, 4.747e-6, 5.321e-6, 5.919e-6, 6.681e-6, 7.101e-6, 7.983e-6,
01828
01829
              8.342e-6, 8.741e-6, 9.431e-6, 9.952e-6, 1.026e-5, 1.055e-5, 1.095e-5, 1.095e-5, 1.087e-5, 1.056e-5, 1.026e-5, 9.715e-6,
01830
01832
              9.252e-6, 8.452e-6, 7.958e-6, 7.268e-6, 6.295e-6, 6.003e-6, 5e-6,
              4.591e-6, 3.983e-6, 3.479e-6, 3.058e-6, 2.667e-6, 2.293e-6,
01833
01834
              1.995e-6, 1.747e-6, 1.517e-6, 1.335e-6, 1.165e-6, 1.028e-6,
              9.007e-7, 7.956e-7, 7.015e-7, 6.192e-7, 5.491e-7, 4.859e-7, 4.297e-7, 3.799e-7, 3.38e-7, 3.002e-7, 2.659e-7, 2.366e-7, 2.103e-7, 1.861e-7, 1.655e-7, 1.469e-7, 1.309e-7, 1.162e-7,
01835
01836
01837
              1.032e-7, 9.198e-8, 8.181e-8, 7.294e-8, 6.516e-8, 5.787e-8,
01839
              5.163e-8, 4.612e-8, 4.119e-8, 3.695e-8, 3.308e-8, 2.976e-8,
             2.67e-8, 2.407e-8, 2.171e-8, 1.965e-8, 1.78e-8, 1.617e-8, 1.47e-8, 1.341e-8, 1.227e-8, 1.125e-8, 1.033e-8, 9.524e-9,
01840
01841
              8.797e-9, 8.162e-9, 7.565e-9, 7.04e-9, 6.56e-9, 6.129e-9,
01842
              5.733e-9, 5.376e-9, 5.043e-9, 4.75e-9, 4.466e-9, 4.211e-9,
01843
              3.977e-9, 3.759e-9, 3.558e-9, 3.373e-9, 3.201e-9, 3.043e-9,
              2.895e-9, 2.76e-9, 2.635e-9, 2.518e-9, 2.411e-9, 2.314e-9, 2.23e-9, 2.151e-9, 2.087e-9, 2.035e-9, 1.988e-9, 1.946e-9,
01845
01846
01847
              1.927e-9, 1.916e-9, 1.916e-9, 1.933e-9, 1.966e-9, 2.018e-9, 2.09e-9, 2.182e-9, 2.299e-9, 2.442e-9, 2.623e-9, 2.832e-9,
01848
              3.079e-9, 3.368e-9, 3.714e-9, 4.104e-9, 4.567e-9, 5.091e-9, 5.701e-9, 6.398e-9, 7.194e-9, 8.127e-9, 9.141e-9, 1.035e-8,
01849
              1.177e-8, 1.338e-8, 1.508e-8, 1.711e-8, 1.955e-8, 2.216e-8,
01851
              2.534e-8, 2.871e-8, 3.291e-8, 3.711e-8, 4.285e-8, 4.868e-8,
01852
              5.509e-8, 6.276e-8, 7.262e-8, 8.252e-8, 9.4e-8, 1.064e-7, 1.247e-7, 1.411e-7, 1.626e-7, 1.827e-7, 2.044e-7, 2.284e-7, 2.452e-7, 2.854e-7, 3.026e-7, 3.278e-7, 3.474e-7, 3.693e-7,
01853
01854
01855
              3.93e-7, 4.104e-7, 4.22e-7, 4.439e-7, 4.545e-7, 4.778e-7,
              4.812e-7, 5.018e-7, 4.899e-7, 5.075e-7, 5.073e-7, 5.171e-7,
01858
              5.131e-7, 5.25e-7, 5.617e-7, 5.846e-7, 6.239e-7, 6.696e-7,
01859
              7.398e-7, 8.073e-7, 9.15e-7, 1.009e-6, 1.116e-6, 1.264e-6,
              1.439e-6, 1.644e-6, 1.856e-6, 2.147e-6, 2.317e-6, 2.713e-6, 2.882e-6, 2.99e-6, 3.489e-6, 3.581e-6, 4.033e-6, 4.26e-6, 4.543e-6, 4.84e-6, 4.826e-6, 5.013e-6, 5.252e-6, 5.277e-6,
01860
01861
01862
              5.306e-6, 5.236e-6, 5.123e-6, 5.171e-6, 4.843e-6, 4.615e-6,
              4.385e-6, 3.97e-6, 3.693e-6, 3.231e-6, 2.915e-6, 2.495e-6, 2.144e-6, 1.91e-6, 1.639e-6, 1.417e-6, 1.226e-6, 1.065e-6,
01864
01865
             9.29e-7, 8.142e-7, 7.161e-7, 6.318e-7, 5.581e-7, 4.943e-7, 4.376e-7, 3.884e-7, 3.449e-7, 3.06e-7, 2.712e-7, 2.412e-7, 2.139e-7, 1.903e-7, 1.689e-7, 1.499e-7, 1.331e-7, 1.183e-7, 1.05e-7, 9.362e-8, 8.306e-8, 7.403e-8, 6.578e-8, 5.853e-8,
01866
01867
01868
              5.216e-8, 4.632e-8, 4.127e-8, 3.678e-8, 3.279e-8, 2.923e-8,
01870
01871
              2.612e-8, 2.339e-8, 2.094e-8, 1.877e-8, 1.686e-8, 1.516e-8,
01872
              1.366e-8, 1.234e-8, 1.114e-8, 1.012e-8, 9.182e-9, 8.362e-9,
              7.634e-9, 6.981e-9, 6.406e-9, 5.888e-9, 5.428e-9, 5.021e-9,
01873
01874
              4.65e-9, 4.326e-9, 4.033e-9, 3.77e-9, 3.536e-9, 3.327e-9,
              3.141e-9, 2.974e-9, 2.825e-9, 2.697e-9, 2.584e-9, 2.488e-9,
              2.406e-9, 2.34e-9, 2.292e-9, 2.259e-9, 2.244e-9, 2.243e-9, 2.272e-9, 2.31e-9, 2.378e-9, 2.454e-9, 2.618e-9, 2.672e-9,
01876
01877
01878
              2.831e-9, 3.05e-9, 3.225e-9, 3.425e-9, 3.677e-9, 3.968e-9,
              4.221e-9, 4.639e-9, 4.96e-9, 5.359e-9, 5.649e-9, 6.23e-9, 6.716e-9, 7.218e-9, 7.746e-9, 7.988e-9, 8.627e-9, 8.999e-9,
01879
01880
```

```
9.442e-9, 9.82e-9, 1.015e-8, 1.06e-8, 1.079e-8, 1.109e-8,
           1.137e-8, 1.186e-8, 1.18e-8, 1.187e-8, 1.194e-8, 1.192e-8,
01882
01883
           1.224e-8, 1.245e-8, 1.246e-8, 1.318e-8, 1.377e-8, 1.471e-8,
01884
           1.582e-8, 1.713e-8, 1.853e-8, 2.063e-8, 2.27e-8, 2.567e-8,
           2.891e-8, 3.264e-8, 3.744e-8, 4.286e-8, 4.915e-8, 5.623e-8, 6.336e-8, 7.293e-8, 8.309e-8, 9.319e-8, 1.091e-7, 1.243e-7,
01885
01886
           1.348e-7, 1.449e-7, 1.62e-7, 1.846e-7, 1.937e-7, 2.04e-7, 2.179e-7, 2.298e-7, 2.433e-7, 2.439e-7, 2.464e-7, 2.611e-7,
01888
01889
           2.617e-7, 2.582e-7, 2.453e-7, 2.401e-7, 2.349e-7, 2.203e-7,
           2.066e-7, 1.939e-7, 1.78e-7, 1.558e-7, 1.391e-7, 1.203e-7, 1.048e-7, 9.464e-8, 8.306e-8, 7.239e-8, 6.317e-8, 5.52e-8, 4.847e-8, 4.282e-8, 3.796e-8, 3.377e-8, 2.996e-8, 2.678e-8,
01890
01891
01892
01893
            2.4e-8, 2.134e-8, 1.904e-8, 1.705e-8, 1.523e-8, 1.35e-8,
            1.204e-8, 1.07e-8, 9.408e-9, 8.476e-9, 7.47e-9, 6.679e-9,
01894
01895
            5.929e-9, 5.267e-9, 4.711e-9, 4.172e-9, 3.761e-9, 3.288e-9,
           2.929e-9, 2.609e-9, 2.315e-9, 2.042e-9, 1.844e-9, 1.64e-9, 1.47e-9, 1.31e-9, 1.176e-9, 1.049e-9, 9.377e-10, 8.462e-10,
01896
01897
01898
            7.616e-10, 6.854e-10, 6.191e-10, 5.596e-10, 5.078e-10, 4.611e-10,
            4.197e-10, 3.83e-10, 3.505e-10, 3.215e-10, 2.956e-10, 2.726e-10,
01900
            2.521e-10, 2.338e-10, 2.173e-10, 2.026e-10, 1.895e-10, 1.777e-10,
            1.672e-10, 1.579e-10, 1.496e-10, 1.423e-10, 1.358e-10, 1.302e-10,
01901
01902
           1.254e-10, 1.216e-10, 1.187e-10, 1.163e-10, 1.147e-10, 1.145e-10,
           1.15e-10, 1.17e-10, 1.192e-10, 1.25e-10, 1.298e-10, 1.345e-10,
01903
           1.405e-10, 1.538e-10, 1.648e-10, 1.721e-10, 1.872e-10, 1.968e-10, 2.089e-10, 2.172e-10, 2.317e-10, 2.389e-10, 2.503e-10, 2.585e-10,
01904
01905
            2.686e-10, 2.8e-10, 2.895e-10, 3.019e-10, 3.037e-10, 3.076e-10,
01906
            3.146e-10, 3.198e-10, 3.332e-10, 3.397e-10, 3.54e-10, 3.667e-10,
01907
01908
           3.895e-10, 4.071e-10, 4.565e-10, 4.983e-10, 5.439e-10, 5.968e-10,
           6.676e-10, 7.456e-10, 8.405e-10, 9.478e-10, 1.064e-9, 1.218e-9, 1.386e-9, 1.581e-9, 1.787e-9, 2.032e-9, 2.347e-9, 2.677e-9,
01909
01910
           3.008e-9, 3.544e-9, 4.056e-9, 4.687e-9, 5.331e-9, 6.227e-9,
01911
            6.854e-9, 8.139e-9, 8.945e-9, 9.865e-9, 1.125e-8, 1.178e-8,
01913
            1.364e-8, 1.436e-8, 1.54e-8, 1.672e-8, 1.793e-8, 1.906e-8,
01914
           2.036e-8, 2.144e-8, 2.292e-8, 2.371e-8, 2.493e-8, 2.606e-8,
           2.706e-8, 2.866e-8, 3.036e-8, 3.136e-8, 3.405e-8, 3.665e-8, 3.837e-8, 4.229e-8, 4.748e-8, 5.32e-8, 5.763e-8, 6.677e-8, 7.216e-8, 7.716e-8, 8.958e-8, 9.419e-8, 1.036e-7, 1.108e-7,
01915
01916
01917
            1.189e-7, 1.246e-7, 1.348e-7, 1.31e-7, 1.361e-7, 1.364e-7,
01919
            1.363e-7, 1.343e-7, 1.293e-7, 1.254e-7, 1.235e-7, 1.158e-7,
01920
            1.107e-7, 9.961e-8, 9.011e-8, 7.91e-8, 6.916e-8, 6.338e-8,
01921
           5.564e-8, 4.827e-8, 4.198e-8, 3.695e-8, 3.276e-8, 2.929e-8,
           2.633e-8, 2.391e-8, 2.192e-8, 2.021e-8, 1.89e-8, 1.772e-8,
01922
           1.667e-8, 1.603e-8, 1.547e-8, 1.537e-8, 1.492e-8, 1.515e-8, 1.479e-8, 1.45e-8, 1.513e-8, 1.495e-8, 1.529e-8, 1.565e-8,
01923
01924
            1.564e-8, 1.553e-8, 1.569e-8, 1.584e-8, 1.57e-8, 1.538e-8,
01925
01926
            1.513e-8, 1.472e-8, 1.425e-8, 1.349e-8, 1.328e-8, 1.249e-8,
01927
           1.17e-8, 1.077e-8, 9.514e-9, 8.614e-9, 7.46e-9, 6.621e-9,
           5.775e-9, 5.006e-9, 4.308e-9, 3.747e-9, 3.24e-9, 2.84e-9, 2.481e-9, 2.184e-9, 1.923e-9, 1.71e-9, 1.504e-9, 1.334e-9,
01928
01929
            1.187e-9, 1.053e-9, 9.367e-10, 8.306e-10, 7.419e-10, 6.63e-10,
01930
            5.918e-10, 5.277e-10, 4.717e-10, 4.222e-10, 3.783e-10, 3.39e-10,
           3.036e-10, 2.729e-10, 2.455e-10, 2.211e-10, 1.995e-10, 1.804e-10,
01932
01933
           1.635e-10, 1.485e-10, 1.355e-10, 1.24e-10, 1.139e-10, 1.051e-10,
           9.757e-11, 9.114e-11, 8.577e-11, 8.139e-11, 7.792e-11, 7.52e-11, 7.39e-11, 7.311e-11, 7.277e-11, 7.482e-11, 7.698e-11, 8.162e-11, 8.517e-11, 8.968e-11, 9.905e-11, 1.075e-10, 1.187e-10, 1.291e-10,
01934
01935
01936
            1.426e-10, 1.573e-10, 1.734e-10, 1.905e-10, 2.097e-10, 2.28e-10,
            2.473e-10, 2.718e-10, 2.922e-10, 3.128e-10, 3.361e-10, 3.641e-10,
01938
01939
           3.91e-10, 4.196e-10, 4.501e-10, 4.932e-10, 5.258e-10, 5.755e-10,
01940
            6.253e-10, 6.664e-10, 7.344e-10, 7.985e-10, 8.877e-10, 1.005e-9,
           1.118e-9, 1.251e-9, 1.428e-9, 1.61e-9, 1.888e-9, 2.077e-9,
01941
           2.331e-9, 2.751e-9, 3.061e-9, 3.522e-9, 3.805e-9, 4.181e-9,
01942
            4.575e-9, 5.167e-9, 5.634e-9, 6.007e-9, 6.501e-9, 6.829e-9,
            7.211e-9, 7.262e-9, 7.696e-9, 7.832e-9, 7.799e-9, 7.651e-9,
01944
01945
            7.304e-9, 7.15e-9, 6.977e-9, 6.603e-9, 6.209e-9, 5.69e-9,
01946
           5.432e-9, 4.764e-9, 4.189e-9, 3.64e-9, 3.203e-9, 2.848e-9,
01947
           2.51 e-9, \ 2.194 e-9, \ 1.946 e-9, \ 1.75 e-9, \ 1.567 e-9, \ 1.426 e-9,
            1.302e-9, 1.197e-9, 1.109e-9, 1.035e-9, 9.719e-10, 9.207e-10,
01948
            8.957e-10, 8.578e-10, 8.262e-10, 8.117e-10, 7.987e-10, 7.875e-10,
01949
            7.741e-10, 7.762e-10, 7.537e-10, 7.424e-10, 7.474e-10, 7.294e-10,
            7.216e-10, 7.233e-10, 7.075e-10, 6.892e-10, 6.618e-10,
01951
                                                                             6.314e-10
01952
            6.208e-10, 5.689e-10, 5.55e-10, 4.984e-10, 4.6e-10, 4.078e-10,
01953
           3.879e-10, 3.459e-10, 2.982e-10, 2.626e-10, 2.329e-10, 1.988e-10,
            1.735e-10, 1.487e-10, 1.297e-10, 1.133e-10, 9.943e-11, 8.736e-11,
01954
            7.726e-11, 6.836e-11, 6.053e-11, 5.384e-11, 4.789e-11, 4.267e-11,
01955
            3.804e-11, 3.398e-11, 3.034e-11, 2.71e-11, 2.425e-11, 2.173e-11,
01956
            1.95e-11, 1.752e-11, 1.574e-11, 1.418e-11, 1.278e-11, 1.154e-11,
01957
01958
            1.044e-11, 9.463e-12, 8.602e-12, 7.841e-12, 7.171e-12, 6.584e-12,
01959
            6.073e-12, 5.631e-12, 5.254e-12, 4.937e-12, 4.679e-12, 4.476e-12,
            4.328e-12, 4.233e-12, 4.194e-12, 4.211e-12, 4.286e-12, 4.424e-12,
01960
            4.628e-12, 4.906e-12, 5.262e-12, 5.708e-12, 6.254e-12, 6.914e-12,
01961
            7.714e-12, 8.677e-12, 9.747e-12, 1.101e-11, 1.256e-11, 1.409e-11,
            1.597e-11, 1.807e-11, 2.034e-11, 2.316e-11, 2.622e-11, 2.962e-11,
01963
01964
            3.369e-11, 3.819e-11, 4.329e-11, 4.932e-11, 5.589e-11, 6.364e-11,
01965
           7.284 e^{-11}, \ 8.236 e^{-11}, \ 9.447 e^{-11}, \ 1.078 e^{-10}, \ 1.229 e^{-10}, \ 1.417 e^{-10},
           1.614e-10, 1.843e-10, 2.107e-10, 2.406e-10, 2.728e-10, 3.195e-10, 3.595e-10, 4.153e-10, 4.736e-10, 5.41e-10, 6.088e-10, 6.769e-10,
01966
01967
```

```
7.691e-10, 8.545e-10, 9.621e-10, 1.047e-9, 1.161e-9, 1.296e-9,
            1.424e-9, 1.576e-9, 1.739e-9, 1.838e-9, 2.08e-9, 2.336e-9, 2.604e-9, 2.76e-9, 3.001e-9, 3.365e-9, 3.55e-9, 3.895e-9,
01969
01970
            4.183e-9, 4.614e-9, 4.846e-9, 5.068e-9, 5.427e-9, 5.541e-9,
01971
            5.864e-9, 5.997e-9, 5.997e-9, 6.061e-9, 5.944e-9, 5.855e-9, 5.661e-9, 5.523e-9, 5.374e-9, 4.94e-9, 4.688e-9, 4.17e-9,
01972
01973
            3.913e-9, 3.423e-9, 2.997e-9, 2.598e-9, 2.253e-9, 1.946e-9,
01974
            1.71e-9, 1.507e-9, 1.336e-9, 1.19e-9, 1.068e-9, 9.623e-10,
01975
01976
            8.772e-10, 8.007e-10, 7.42e-10, 6.884e-10, 6.483e-10, 6.162e-10,
01977
            5.922e-10, 5.688e-10, 5.654e-10, 5.637e-10, 5.701e-10, 5.781e-10,
01978
            5.874e-10, 6.268e-10, 6.357e-10, 6.525e-10, 7.137e-10, 7.441e-10,
01979
            8.024e-10, 8.485e-10, 9.143e-10, 9.536e-10, 9.717e-10, 1.018e-9,
            1.042e-9, 1.054e-9, 1.092e-9, 1.079e-9, 1.064e-9, 1.043e-9, 1.02e-9, 9.687e-10, 9.273e-10, 9.208e-10, 9.068e-10, 7.687e-10,
01980
01981
01982
            7.385e-10, 6.595e-10, 5.87e-10, 5.144e-10, 4.417e-10, 3.804e-10,
            3.301e-10, 2.866e-10, 2.509e-10, 2.202e-10, 1.947e-10, 1.719e-10, 1.525e-10, 1.361e-10, 1.21e-10, 1.084e-10, 9.8e-11, 8.801e-11,
01983
01984
            7.954e-11, 7.124e-11, 6.335e-11, 5.76e-11, 5.132e-11, 4.601e-11,
01985
            4.096e-11, 3.657e-11, 3.25e-11, 2.909e-11, 2.587e-11, 2.297e-11,
01986
01987
            2.05e-11, 1.828e-11, 1.632e-11, 1.462e-11, 1.314e-11, 1.185e-11,
            1.073e-11, 9.76e-12, 8.922e-12, 8.206e-12, 7.602e-12, 7.1e-12,
01988
01989
            6.694e-12, 6.378e-12, 6.149e-12, 6.004e-12, 5.941e-12, 5.962e-12,
01990
            6.069e-12, 6.265e-12, 6.551e-12, 6.935e-12, 7.457e-12, 8.074e-12,
            8.811e-12, 9.852e-12, 1.086e-11, 1.207e-11, 1.361e-11, 1.553e-11, 1.737e-11, 1.93e-11, 2.175e-11, 2.41e-11, 2.706e-11, 3.023e-11,
01991
01992
            3.313e-11, 3.657e-11, 4.118e-11, 4.569e-11, 5.025e-11, 5.66e-11,
01993
            6.231e-11, 6.881e-11, 7.996e-11, 8.526e-11, 9.694e-11, 1.106e-10,
01994
01995
            1.222e-10, 1.355e-10, 1.525e-10, 1.775e-10, 1.924e-10, 2.181e-10,
01996
            2.379e-10, 2.662e-10, 2.907e-10, 3.154e-10, 3.366e-10, 3.579e-10,
            3.858e-10, 4.046e-10, 4.196e-10, 4.166e-10, 4.457e-10, 4.466e-10,
01997
01998
            4.404e-10, 4.337e-10, 4.15e-10, 4.083e-10, 3.91e-10, 3.723e-10,
01999
            3.514e-10, 3.303e-10, 2.847e-10, 2.546e-10, 2.23e-10, 1.994e-10,
            1.733e-10, 1.488e-10, 1.297e-10, 1.144e-10, 1.004e-10, 8.741e-11,
02000
02001
            7.928e-11, 7.034e-11, 6.323e-11, 5.754e-11, 5.25e-11, 4.85e-11,
02002
            4.502e-11, 4.286e-11, 4.028e-11, 3.899e-11, 3.824e-11, 3.761e-11,
02003
            3.804e-11, 3.839e-11, 3.845e-11, 4.244e-11, 4.382e-11, 4.582e-11,
            4.847e-11, 5.209e-11, 5.384e-11, 5.887e-11, 6.371e-11, 6.737e-11,
02004
            7.168e-11, 7.415e-11, 7.827e-11, 8.037e-11, 8.12e-11, 8.071e-11,
02006
            8.008e-11, 7.851e-11, 7.544e-11, 7.377e-11, 7.173e-11, 6.801e-11,
            6.267e-11, 5.727e-11, 5.288e-11, 4.853e-11, 4.082e-11, 3.645e-11,
02007
02008
            3.136e-11, 2.672e-11, 2.304e-11, 1.986e-11, 1.725e-11, 1.503e-11,
            1.315e-11, 1.153e-11, 1.014e-11, 8.942e-12, 7.901e-12, 6.993e-12, 6.199e-12, 5.502e-12, 4.89e-12, 4.351e-12, 3.878e-12, 3.461e-12, 3.094e-12, 2.771e-12, 2.488e-12, 2.241e-12, 2.025e-12, 1.838e-12,
02009
02010
02011
            1.677e-12, 1.541e-12, 1.427e-12, 1.335e-12, 1.262e-12, 1.209e-12,
02012
02013
            1.176e-12, 1.161e-12, 1.165e-12, 1.189e-12, 1.234e-12, 1.3e-12,
02014
            1.389e-12, 1.503e-12, 1.644e-12, 1.814e-12, 2.017e-12, 2.255e-12,
            2.534e-12, 2.858e-12, 3.231e-12, 3.661e-12, 4.153e-12, 4.717e-12, 5.36e-12, 6.094e-12, 6.93e-12, 7.882e-12, 8.966e-12, 1.02e-11, 1.162e-11, 1.324e-11, 1.51e-11, 1.72e-11, 1.965e-11, 2.237e-11,
02015
02016
02017
            2.56e-11, 2.927e-11, 3.371e-11, 3.842e-11, 4.429e-11, 5.139e-11,
            5.798e-11, 6.697e-11, 7.626e-11, 8.647e-11, 1.022e-10, 1.136e-10,
02019
02020
            1.3e-10, 1.481e-10, 1.672e-10, 1.871e-10, 2.126e-10, 2.357e-10,
            2.583e-10, 2.997e-10, 3.289e-10, 3.702e-10, 4.012e-10, 4.319e-10, 4.527e-10, 5.001e-10, 5.448e-10, 5.611e-10, 5.76e-10, 5.965e-10,
02021
02022
            6.079e-10, 6.207e-10, 6.276e-10, 6.222e-10, 6.137e-10, 6e-10,
02023
            5.814e-10, 5.393e-10, 5.35e-10, 4.947e-10, 4.629e-10, 4.117e-10,
            3.712e-10, 3.372e-10, 2.923e-10, 2.55e-10, 2.232e-10, 1.929e-10,
02025
            1.679e-10, 1.46e-10, 1.289e-10, 1.13e-10, 9.953e-11, 8.763e-11,
02026
02027
            7.76e-11, 6.9e-11, 6.16e-11, 5.525e-11, 4.958e-11, 4.489e-11,
            4.072e-11, 3.728e-11, 3.438e-11, 3.205e-11, 3.006e-11, 2.848e-11, 2.766e-11, 2.688e-11, 2.664e-11, 2.67e-11, 2.696e-11, 2.786e-11,
02028
02029
            2.861e-11, 3.009e-11, 3.178e-11, 3.389e-11, 3.587e-11, 3.819e-11,
            4.054e-11, 4.417e-11, 4.703e-11, 5.137e-11, 5.46e-11, 6.055e-11, 6.333e-11, 6.773e-11, 7.219e-11, 7.717e-11, 8.131e-11, 8.491e-11,
02031
02032
            8.574e-11, 9.01e-11, 9.017e-11, 8.999e-11, 8.959e-11, 8.838e-11,
02033
02034
            8.579e-11, 8.162e-11, 8.098e-11, 7.472e-11, 7.108e-11, 6.559e-11,
            5.994e-11, 5.172e-11, 4.424e-11, 3.951e-11, 3.34e-11, 2.902e-11,
02035
            2.541e-11, 2.215e-11, 1.945e-11, 1.716e-11, 1.503e-11, 1.339e-11, 1.185e-11, 1.05e-11, 9.336e-12, 8.307e-12, 7.312e-12, 6.55e-12,
02036
02038
            5.836e-12, 5.178e-12, 4.6e-12, 4.086e-12, 3.639e-12, 3.247e-12,
02039
            2.904e-12, 2.604e-12, 2.341e-12, 2.112e-12, 1.914e-12, 1.744e-12,
02040
            1.598e-12, 1.476e-12, 1.374e-12, 1.293e-12, 1.23e-12, 1.185e-12,
            1.158e-12, 1.147e-12, 1.154e-12, 1.177e-12, 1.219e-12, 1.28e-12,
02041
            1.36e-12, 1.463e-12, 1.591e-12, 1.75e-12, 1.94e-12, 2.156e-12, 2.43e-12, 2.748e-12, 3.052e-12, 3.533e-12, 3.967e-12, 4.471e-12,
02042
02043
            5.041e-12, 5.86e-12, 6.664e-12, 7.522e-12, 8.342e-12, 9.412e-12,
02044
02045
            1.072e-11, 1.213e-11, 1.343e-11, 1.496e-11, 1.664e-11, 1.822e-11,
02046
            2.029e-11, 2.233e-11, 2.457e-11, 2.709e-11, 2.928e-11, 3.115e-11,
02047
            3.356e-11, 3.592e-11, 3.818e-11, 3.936e-11, 4.061e-11, 4.149e-11,
            4.299e-11, 4.223e-11, 4.251e-11, 4.287e-11, 4.177e-11, 4.094e-11, 3.942e-11, 3.772e-11, 3.614e-11, 3.394e-11, 3.222e-11, 2.791e-11,
02048
            2.665e-11, 2.309e-11, 2.032e-11, 1.74e-11, 1.535e-11, 1.323e-11, 1.151e-11, 9.803e-12, 8.65e-12, 7.54e-12, 6.619e-12, 5.832e-12,
02050
02051
02052
            5.113e-12, 4.503e-12, 3.975e-12, 3.52e-12, 3.112e-12, 2.797e-12,
            2.5e-12, 2.24e-12, 2.013e-12, 1.819e-12, 1.653e-12, 1.513e-12, 1.395e-12, 1.299e-12, 1.225e-12, 1.168e-12, 1.124e-12, 1.148e-12,
02053
02054
```

```
1.107e-12, 1.128e-12, 1.169e-12, 1.233e-12, 1.307e-12, 1.359e-12,
             1.543e-12, 1.686e-12, 1.794e-12, 2.028e-12, 2.21e-12, 2.441e-12, 2.653e-12, 2.828e-12, 3.093e-12, 3.28e-12, 3.551e-12, 3.677e-12,
02056
02057
02058
             3.803e-12, 3.844e-12, 4.068e-12, 4.093e-12, 4.002e-12, 3.904e-12,
02059
             3.624e-12, 3.633e-12, 3.622e-12, 3.443e-12, 3.184e-12, 2.934e-12, 2.476e-12, 2.212e-12, 1.867e-12, 1.594e-12, 1.37e-12, 1.192e-12,
02060
             1.045e-12, 9.211e-13, 8.17e-13, 7.29e-13, 6.55e-13, 5.929e-13,
             5.415e-13, 4.995e-13, 4.661e-13, 4.406e-13, 4.225e-13, 4.116e-13,
02062
02063
             4.075e-13, 4.102e-13, 4.198e-13, 4.365e-13, 4.606e-13, 4.925e-13,
02064
             5.326e-13, 5.818e-13, 6.407e-13, 7.104e-13, 7.92e-13, 8.868e-13,
             9.964e-13, 1.123e-12, 1.268e-12, 1.434e-12, 1.626e-12, 1.848e-12,
02065
02066
             2.107e-12, 2.422e-12, 2.772e-12, 3.145e-12, 3.704e-12, 4.27e-12,
02067
             4.721e-12, 5.361e-12, 6.083e-12, 7.095e-12, 7.968e-12, 9.228e-12,
             1.048e-11, 1.187e-11, 1.336e-11, 1.577e-11, 1.772e-11, 2.017e-11,
02068
02069
             2.25e-11, 2.63e-11, 2.911e-11, 3.356e-11, 3.82e-11, 4.173e-11,
02070
             4.811e-11, 5.254e-11, 5.839e-11, 6.187e-11, 6.805e-11, 7.118e-11,
             7.369e-11, 7.664e-11, 7.794e-11, 7.947e-11, 8.036e-11, 7.954e-11,
02071
             7.849e-11, 7.518e-11, 7.462e-11, 6.926e-11, 6.531e-11, 6.197e-11, 5.421e-11, 4.777e-11, 4.111e-11, 3.679e-11, 3.166e-11, 2.786e-11,
02072
02074
             2.436e-11, 2.144e-11, 1.859e-11, 1.628e-11, 1.414e-11, 1.237e-11,
02075
             1.093e-11, 9.558e-12
02076
02077
          static double h2o260[2001] = { .2752, .2732, .2749, .2676, .2667, .2545, .2497, .2327, .2218, .2036, .1825, .1694, .1497, .1353, .121, .1014, .09405, .07848, .07195, .06246, .05306, .04853, .04138,
02078
02079
02080
             .03735, .03171, .02785, .02431, .02111,
                                                                   .01845, .0164,
02081
             .01255, .01098, .009797, .008646, .007779, .006898, .006099,
02082
             .005453, .004909, .004413, .003959, .003581, .003199, .002871, .002583, .00233, .002086, .001874, .001684, .001512, .001361, .001225, .0011, 9.89e-4, 8.916e-4, 8.039e-4, 7.256e-4, 6.545e-4,
02083
02084
02085
             5.918e-4, 5.359e-4, 4.867e-4, 4.426e-4, 4.033e-4, 3.682e-4,
             3.366e-4, 3.085e-4, 2.833e-4, 2.605e-4, 2.403e-4, 2.221e-4
02087
02088
             2.055e-4, 1.908e-4, 1.774e-4, 1.653e-4, 1.544e-4, 1.443e-4,
             1.351e-4, 1.267e-4, 1.19e-4, 1.119e-4, 1.053e-4, 9.922e-5, 9.355e-5, 8.831e-5, 8.339e-5, 7.878e-5, 7.449e-5, 7.043e-5, 6.664e-5, 6.307e-5, 5.969e-5, 5.654e-5, 5.357e-5, 5.075e-5,
02089
02090
02091
             4.81e-5, 4.56e-5, 4.322e-5, 4.102e-5, 3.892e-5, 3.696e-5,
02093
             3.511e-5, 3.339e-5, 3.177e-5, 3.026e-5, 2.886e-5, 2.756e-
02094
             2.636e-5, 2.527e-5, 2.427e-5, 2.337e-5, 2.257e-5, 2.185e-5
             2.127e-5, 2.08e-5, 2.041e-5, 2.013e-5, 2e-5, 1.997e-5, 2.009e-5, 2.031e-5, 2.068e-5, 2.124e-5, 2.189e-5, 2.267e-5, 2.364e-5,
02095
02096
             2.463e-5, 2.618e-5, 2.774e-5, 2.937e-5, 3.144e-5, 3.359e-5, 3.695e-5, 4.002e-5, 4.374e-5, 4.947e-5, 5.431e-5, 6.281e-5,
02097
02098
             7.169e-5, 8.157e-5, 9.728e-5, 1.079e-4, 1.337e-4, 1.442e-4,
02099
02100
             1.683e-4, 1.879e-4, 2.223e-4, 2.425e-4, 2.838e-4, 3.143e-4,
02101
             3.527e-4, 4.012e-4, 4.237e-4, 4.747e-4, 5.057e-4, 5.409e-4,
02102
             5.734e-4, 5.944e-4, 6.077e-4, 6.175e-4, 6.238e-4, 6.226e-4,
             6.248e-4, 6.192e-4, 6.098e-4, 5.818e-4, 5.709e-4, 5.465e-4,
02103
02104
             5.043e-4, 4.699e-4, 4.294e-4, 3.984e-4, 3.672e-4, 3.152e-4,
             2.883e-4, 2.503e-4, 2.211e-4, 1.92e-4, 1.714e-4, 1.485e-4,
             1.358e-4, 1.156e-4, 1.021e-4, 8.887e-5, 7.842e-5, 7.12e-5,
02106
02107
             6.186e-5, 5.73e-5, 4.792e-5, 4.364e-5, 3.72e-5, 3.28e-5,
             2.946e-5, 2.591e-5, 2.261e-5, 2.048e-5, 1.813e-5, 1.63e-5, 1.447e-5, 1.282e-5, 1.167e-5, 1.041e-5, 9.449e-6, 8.51e-6,
02108
02109
             7.596e-6, 6.961e-6, 6.272e-6, 5.728e-6, 5.198e-6, 4.667e-6,
02110
             4.288e-6, 3.897e-6, 3.551e-6, 3.235e-6, 2.952e-6, 2.688e-6,
02111
             2.449e-6, 2.241e-6, 2.05e-6, 1.879e-6, 1.722e-6, 1.582e-6,
02112
             1.456e-6, 1.339e-6, 1.236e-6, 1.144e-6, 1.06e-6, 9.83e-7,
02113
             9.149e-7, 8.535e-7, 7.973e-7, 7.466e-7, 6.999e-7, 6.574e-7, 6.18e-7, 5.821e-7, 5.487e-7, 5.18e-7, 4.896e-7, 4.631e-7, 4.386e-7, 4.16e-7, 3.945e-7, 3.748e-7, 3.562e-7, 3.385e-7, 3.222e-7, 3.068e-7, 2.922e-7, 2.788e-7, 2.659e-7, 2.539e-7,
02114
02115
02116
             2.425e-7, 2.318e-7, 2.219e-7, 2.127e-7, 2.039e-7, 1.958e-7, 1.885e-7, 1.818e-7, 1.758e-7, 1.711e-7, 1.662e-7, 1.63e-7,
02118
02119
             1.605e-7, 1.58e-7, 1.559e-7, 1.545e-7, 1.532e-7, 1.522e-7, 1.51e-7, 1.495e-7, 1.465e-7, 1.483e-7, 1.469e-7, 1.448e-7, 1.444e-7, 1.436e-7, 1.426e-7, 1.431e-7, 1.425e-7, 1.445e-7,
02120
02121
02122
             1.477e-7, 1.515e-7, 1.567e-7, 1.634e-7, 1.712e-7, 1.802e-7,
02123
             1.914e-7, 2.024e-7, 2.159e-7, 2.295e-7, 2.461e-7, 2.621e-7,
02124
             2.868e-7, 3.102e-7, 3.394e-7, 3.784e-7, 4.223e-7, 4.864e-7,
02125
02126
             5.501e-7, 6.039e-7, 7.193e-7, 7.728e-7, 9.514e-7, 1.073e-6,
             1.18e-6, 1.333e-6, 1.472e-6, 1.566e-6, 1.677e-6, 1.784e-6, 1.904e-6, 1.953e-6, 2.02e-6, 2.074e-6, 2.128e-6, 2.162e-6,
02127
02128
             2.219e-6, 2.221e-6, 2.249e-6, 2.239e-6, 2.235e-6, 2.185e-6,
02129
             2.141e-6, 2.124e-6, 2.09e-6, 2.068e-6, 2.1e-6, 2.104e-6,
02130
             2.142e-6, 2.181e-6, 2.257e-6, 2.362e-6, 2.5e-6, 2.664e-6,
02131
             2.884e-6, 3.189e-6, 3.48e-6, 3.847e-6, 4.313e-6, 4.79e-6, 5.25e-6, 5.989e-6, 6.692e-6, 7.668e-6, 8.52e-6, 9.606e-6, 1.073e-5, 1.225e-5, 1.377e-5, 1.582e-5, 1.761e-5, 2.029e-5,
02132
02133
02134
             2.284e-5, 2.602e-5, 2.94e-5, 3.483e-5, 3.928e-5, 4.618e-5, 5.24e-5, 6.132e-5, 7.183e-5, 8.521e-5, 9.111e-5, 1.07e-4,
02135
             1.184e-4, 1.264e-4, 1.475e-4, 1.612e-4, 1.704e-4, 1.818e-4,
02137
02138
             1.924e-4, 1.994e-4, 2.061e-4, 2.18e-4, 2.187e-4, 2.2e-4,
02139
             2.196e-4, 2.131e-4, 2.015e-4, 1.988e-4, 1.847e-4, 1.729e-4,
             1.597e-4, 1.373e-4, 1.262e-4, 1.087e-4, 9.439e-5, 8.061e-5, 7.093e-5, 6.049e-5, 5.12e-5, 4.435e-5, 3.817e-5, 3.34e-5,
02140
02141
```

```
2.927e-5, 2.573e-5, 2.291e-5, 2.04e-5, 1.827e-5, 1.636e-5,
             1.463e-5, 1.309e-5, 1.17e-5, 1.047e-5, 9.315e-6, 8.328e-6, 7.458e-6, 6.665e-6, 5.94e-6, 5.316e-6, 4.752e-6, 4.252e-6,
02143
02144
02145
             3.825e-6, 3.421e-6, 3.064e-6, 2.746e-6, 2.465e-6, 2.216e-6,
             1.99e-6, 1.79e-6, 1.609e-6, 1.449e-6, 1.306e-6, 1.177e-6, 1.063e-6, 9.607e-7, 8.672e-7, 7.855e-7, 7.118e-7, 6.46e-7,
02146
02147
             5.871e-7, 5.34e-7, 4.868e-7, 4.447e-7, 4.068e-7, 3.729e-7, 3.423e-7, 3.151e-7, 2.905e-7, 2.686e-7, 2.484e-7, 2.306e-7,
02149
             2.142e-7, 1.995e-7, 1.86e-7, 1.738e-7, 1.626e-7, 1.522e-7, 1.427e-7, 1.338e-7, 1.258e-7, 1.183e-7, 1.116e-7, 1.056e-7, 9.972e-8, 9.46e-8, 9.007e-8, 8.592e-8, 8.195e-8, 7.816e-8,
02150
02151
02152
             7.483e-8, 7.193e-8, 6.892e-8, 6.642e-8, 6.386e-8, 6.154e-8, 5.949e-8, 5.764e-8, 5.622e-8, 5.479e-8, 5.364e-8, 5.301e-8,
02153
02154
             5.267e-8, 5.263e-8, 5.313e-8, 5.41e-8, 5.55e-8, 5.745e-8,
02155
02156
             6.003e-8, 6.311e-8, 6.713e-8, 7.173e-8, 7.724e-8, 8.368e-8,
             9.121e-8, 9.986e-8, 1.097e-7, 1.209e-7, 1.338e-7, 1.486e-7, 1.651e-7, 1.837e-7, 2.048e-7, 2.289e-7, 2.557e-7, 2.857e-7,
02157
02158
             3.195e-7, 3.587e-7, 4.015e-7, 4.497e-7, 5.049e-7, 5.665e-7,
02159
             6.366e-7, 7.121e-7, 7.996e-7, 8.946e-7, 1.002e-6, 1.117e-6,
02160
02161
             1.262e-6, 1.416e-6, 1.611e-6, 1.807e-6, 2.056e-6, 2.351e-6,
             2.769e-6, 3.138e-6, 3.699e-6, 4.386e-6, 5.041e-6, 6.074e-6,
02162
02163
             6.812e-6, 7.79e-6, 8.855e-6, 1.014e-5, 1.095e-5, 1.245e-5,
02164
             1.316e-5, 1.39e-5, 1.504e-5, 1.583e-5, 1.617e-5, 1.652e-5,
             1.713e-5, 1.724e-5, 1.715e-5, 1.668e-5, 1.629e-5, 1.552e-5, 1.478e-5, 1.34e-5, 1.245e-5, 1.121e-5, 9.575e-6, 8.956e-6,
02165
02166
02167
             7.345e-6, 6.597e-6, 5.612e-6, 4.818e-6, 4.165e-6, 3.579e-6,
             3.041e-6, 2.623e-6, 2.29e-6, 1.984e-6, 1.748e-6, 1.534e-6,
02168
             1.369e-6, 1.219e-6, 1.092e-6, 9.8e-7, 8.762e-7, 7.896e-7, 7.104e-7, 6.364e-7, 5.691e-7, 5.107e-7, 4.575e-7, 4.09e-7
02169
02170
             3.667e-7, 3.287e-7, 2.931e-7, 2.633e-7, 2.356e-7, 2.111e-7,
02171
02172
             1.895e-7, 1.697e-7, 1.525e-7, 1.369e-7, 1.233e-7, 1.114e-7,
02173
             9.988e-8, 9.004e-8, 8.149e-8, 7.352e-8, 6.662e-8, 6.03e-8,
             5.479e-8, 4.974e-8, 4.532e-8, 4.129e-8, 3.781e-8, 3.462e-8
02174
02175
             3.176e-8, 2.919e-8, 2.687e-8, 2.481e-8, 2.292e-8, 2.119e-8,
02176
             1.967e-8, 1.828e-8, 1.706e-8, 1.589e-8, 1.487e-8, 1.393e-8,
             1.307e-8, 1.228e-8, 1.156e-8, 1.089e-8, 1.028e-8, 9.696e-9, 9.159e-9, 8.658e-9, 8.187e-9, 7.746e-9, 7.34e-9, 6.953e-9,
02177
02178
             6.594e-9, 6.259e-9, 5.948e-9, 5.66e-9, 5.386e-9, 5.135e-9,
02180
             4.903e-9, 4.703e-9, 4.515e-9, 4.362e-9, 4.233e-9, 4.117e-9,
             4.017e-9, 3.962e-9, 3.924e-9, 3.905e-9, 3.922e-9, 3.967e-9,
02181
02182
             4.046e-9, 4.165e-9, 4.32e-9, 4.522e-9, 4.769e-9, 5.083e-9, 5.443e-9, 5.872e-9, 6.366e-9, 6.949e-9, 7.601e-9, 8.371e-9,
02183
             9.22e-9, 1.02e-8, 1.129e-8, 1.251e-8, 1.393e-8, 1.542e-8, 1.72e-8, 1.926e-8, 2.152e-8, 2.392e-8, 2.678e-8, 3.028e-8,
02184
02185
             3.39e-8, 3.836e-8, 4.309e-8, 4.9e-8, 5.481e-8, 6.252e-8,
02186
02187
             7.039e-8, 7.883e-8, 8.849e-8, 1.012e-7, 1.142e-7, 1.3e-7,
02188
             1.475e-7, 1.732e-7, 1.978e-7, 2.304e-7, 2.631e-7, 2.988e-7,
             3.392e-7, 3.69e-7, 4.355e-7, 4.672e-7, 5.11e-7, 5.461e-7, 5.828e-7, 6.233e-7, 6.509e-7, 6.672e-7, 6.969e-7, 7.104e-7,
02189
02190
             7.439e-7, 7.463e-7, 7.708e-7, 7.466e-7, 7.668e-7, 7.549e-7,
02191
02192
             7.586e-7, 7.384e-7, 7.439e-7, 7.785e-7, 7.915e-7, 8.31e-7,
             8.745e-7, 9.558e-7, 1.038e-6, 1.173e-6, 1.304e-6, 1.452e-6,
02193
02194
             1.671e-6, 1.931e-6, 2.239e-6, 2.578e-6, 3.032e-6, 3.334e-6,
02195
             3.98e-6, 4.3e-6, 4.518e-6, 5.321e-6, 5.508e-6, 6.211e-6, 6.59e-6,
             7.046e-6, 7.555e-6, 7.558e-6, 7.875e-6, 8.319e-6, 8.433e-6, 8.59e-6, 8.503e-6, 8.304e-6, 8.336e-6, 7.739e-6, 7.301e-6,
02196
02197
             6.827e-6, 6.078e-6, 5.551e-6, 4.762e-6, 4.224e-6, 3.538e-6,
             2.984e-6, 2.619e-6, 2.227e-6, 1.923e-6, 1.669e-6, 1.462e-6,
02199
02200
             1.294e-6, 1.155e-6, 1.033e-6, 9.231e-7, 8.238e-7, 7.36e-7,
             6.564e-7, 5.869e-7, 5.236e-7, 4.673e-7, 4.174e-7, 3.736e-7, 3.33e-7, 2.976e-7, 2.657e-7, 2.367e-7, 2.106e-7, 1.877e-7, 1.671e-7, 1.494e-7, 1.332e-7, 1.192e-7, 1.065e-7, 9.558e-8, 8.586e-8, 7.717e-8, 6.958e-8, 6.278e-8, 5.666e-8, 5.121e-8,
02201
02202
02203
02204
02205
             4.647e-8, 4.213e-8, 3.815e-8, 3.459e-8, 3.146e-8, 2.862e-8,
02206
             2.604e-8, 2.375e-8, 2.162e-8, 1.981e-8, 1.817e-8, 1.67e-8,
02207
             1.537e-8, 1.417e-8, 1.31e-8, 1.215e-8, 1.128e-8, 1.05e-8,
02208
             9.793e-9, 9.158e-9, 8.586e-9, 8.068e-9, 7.595e-9, 7.166e-9, 6.778e-9, 6.427e-9, 6.108e-9, 5.826e-9, 5.571e-9, 5.347e-9,
02209
02210
             5.144e-9, 4.968e-9, 4.822e-9, 4.692e-9, 4.589e-9, 4.506e-9,
             4.467e-9, 4.44e-9, 4.466e-9, 4.515e-9, 4.718e-9, 4.729e-9,
             4.937e-9, 5.249e-9, 5.466e-9, 5.713e-9, 6.03e-9, 6.436e-9, 6.741e-9, 7.33e-9, 7.787e-9, 8.414e-9, 8.908e-9, 9.868e-9,
02212
02213
02214
             1.069e-8, 1.158e-8, 1.253e-8, 1.3e-8, 1.409e-8, 1.47e-8,
             1.548e-8, 1.612e-8, 1.666e-8, 1.736e-8, 1.763e-8, 1.812e-8,
02215
             1.852e-8, 1.923e-8, 1.897e-8, 1.893e-8, 1.888e-8,
02216
                                                                               1.868e-8.
             1.895e-8, 1.899e-8, 1.876e-8, 1.96e-8, 2.02e-8, 2.121e-8,
02217
             2.239e-8, 2.379e-8, 2.526e-8, 2.766e-8, 2.994e-8, 3.332e-8,
02218
02219
             3.703e-8, 4.158e-8, 4.774e-8, 5.499e-8, 6.355e-8, 7.349e-8,
02220
             8.414e-8, 9.846e-8, 1.143e-7, 1.307e-7, 1.562e-7, 1.817e-7,
             3.443e-7, 3.192e-7, 2.485e-7, 2.867e-7, 3.035e-7, 3.223e-7, 3.43e-7, 3.617e-7, 3.793e-7, 3.793e-7, 3.839e-7, 4.081e-7, 4.085e-7, 3.92e-7, 3.851e-7, 3.754e-7, 3.49e-7, 3.229e-7, 2.978e-7, 2.691e-7, 2.312e-7, 2.029e-7, 1.721e-7,
02221
02222
02224
02225
             1.472e-7, 1.308e-7, 1.132e-7, 9.736e-8, 8.458e-8, 7.402e-8,
02226
             6.534e-8, 5.811e-8, 5.235e-8, 4.762e-8, 4.293e-8, 3.896e-8,
             3.526e-8, 3.165e-8, 2.833e-8, 2.551e-8, 2.288e-8, 2.036e-8, 1.82e-8, 1.626e-8, 1.438e-8, 1.299e-8, 1.149e-8, 1.03e-8,
02227
02228
```

```
02229
             9.148e-9, 8.122e-9, 7.264e-9, 6.425e-9, 5.777e-9, 5.06e-9,
            4.502e-9, 4.013e-9, 3.567e-9, 3.145e-9, 2.864e-9, 2.553e-9, 2.311e-9, 2.087e-9, 1.886e-9, 1.716e-9, 1.556e-9, 1.432e-9,
02230
02231
02232
             1.311e-9, 1.202e-9, 1.104e-9, 1.013e-9, 9.293e-10, 8.493e-10,
             7.79e-10, 7.185e-10, 6.642e-10, 6.141e-10, 5.684e-10, 5.346e-10, 5.032e-10, 4.725e-10, 4.439e-10, 4.176e-10, 3.93e-10, 3.714e-10,
02233
02234
             3.515e-10, 3.332e-10, 3.167e-10, 3.02e-10, 2.887e-10, 2.769e-10,
             2.665e-10, 2.578e-10, 2.503e-10, 2.436e-10, 2.377e-10, 2.342e-10,
02236
02237
             2.305e-10, 2.296e-10, 2.278e-10, 2.321e-10, 2.355e-10, 2.402e-10,
             2.478e-10, 2.67e-10, 2.848e-10, 2.982e-10, 3.263e-10, 3.438e-10, 3.649e-10, 3.829e-10, 4.115e-10, 4.264e-10, 4.473e-10, 4.63e-10,
02238
02239
02240
             4.808e-10, 4.995e-10, 5.142e-10, 5.313e-10, 5.318e-10, 5.358e-10,
02241
             5.452e-10, 5.507e-10, 5.698e-10, 5.782e-10, 5.983e-10, 6.164e-10,
             6.532e-10, 6.811e-10, 7.624e-10, 8.302e-10, 9.067e-10,
                                                                                    9.937e-10.
02242
02243
             1.104e-9, 1.221e-9, 1.361e-9, 1.516e-9, 1.675e-9, 1.883e-9,
             2.101e-9, 2.349e-9, 2.614e-9, 2.92e-9, 3.305e-9, 3.724e-9, 4.142e-9, 4.887e-9, 5.614e-9, 6.506e-9, 7.463e-9, 8.817e-9, 9.849e-9, 1.187e-8, 1.321e-8, 1.474e-8, 1.698e-8, 1.794e-8,
02244
02245
02246
             2.09e-8, 2.211e-8, 2.362e-8, 2.556e-8, 2.729e-8, 2.88e-8,
02248
             3.046e-8, 3.167e-8, 3.367e-8, 3.457e-8, 3.59e-8, 3.711e-8,
             3.826e-8, 4.001e-8, 4.211e-8, 4.315e-8, 4.661e-8, 5.01e-8, 5.249e-8, 5.84e-8, 6.628e-8, 7.512e-8, 8.253e-8, 9.722e-8, 1.067e-7, 1.153e-7, 1.347e-7, 1.428e-7, 1.577e-7, 1.694e-7,
02249
02250
02251
             1.833e-7, 1.938e-7, 2.108e-7, 2.059e-7, 2.157e-7, 2.185e-7, 2.208e-7, 2.182e-7, 2.093e-7, 2.014e-7, 1.962e-7, 1.819e-7,
02252
02253
             1.713e-7, 1.51e-7, 1.34e-7, 1.154e-7, 9.89e-8, 8.88e-8, 7.673e-8,
02254
             6.599e-8, 5.73e-8, 5.081e-8, 4.567e-8, 4.147e-8, 3.773e-8,
02255
02256
             3.46e-8, 3.194e-8, 2.953e-8, 2.759e-8, 2.594e-8, 2.442e-8,
            2.355e-8, 2.283e-8, 2.279e-8, 2.231e-8, 2.279e-8, 2.239e-8, 2.21e-8, 2.309e-8, 2.293e-8, 2.352e-8, 2.415e-8, 2.43e-8, 2.426e-8, 2.465e-8, 2.5e-8, 2.496e-8, 2.465e-8, 2.445e-8,
02257
02258
02259
02260
             2.383e-8, 2.299e-8, 2.165e-8, 2.113e-8, 1.968e-8, 1.819e-8,
             1.644e-8, 1.427e-8, 1.27e-8, 1.082e-8, 9.428e-9, 8.091e-9,
02261
02262
             6.958e-9, 5.988e-9, 5.246e-9, 4.601e-9, 4.098e-9, 3.664e-9,
             3.287e-9, 2.942e-9, 2.656e-9, 2.364e-9, 2.118e-9, 1.903e-9, 1.703e-9, 1.525e-9, 1.365e-9, 1.229e-9, 1.107e-9, 9.96e-10,
02263
02264
             8.945e-10, 8.08e-10, 7.308e-10, 6.616e-10, 5.994e-10, 5.422e-10, 4.929e-10, 4.478e-10, 4.07e-10, 3.707e-10, 3.379e-10, 3.087e-10,
02265
02267
             2.823e-10, 2.592e-10, 2.385e-10, 2.201e-10, 2.038e-10, 1.897e-10,
             1.774e-10, 1.667e-10, 1.577e-10, 1.502e-10, 1.437e-10, 1.394e-10,
02268
02269
             1.358e-10, 1.324e-10, 1.329e-10, 1.324e-10, 1.36e-10, 1.39e-10,
             1.424e-10, 1.544e-10, 1.651e-10, 1.817e-10, 1.984e-10, 2.195e-10,
02270
             2.438e-10, 2.7e-10, 2.991e-10, 3.322e-10, 3.632e-10, 3.957e-10, 4.36e-10, 4.701e-10, 5.03e-10, 5.381e-10, 5.793e-10, 6.19e-10,
02271
             6.596e-10, 7.004e-10, 7.561e-10, 7.934e-10, 8.552e-10, 9.142e-10,
02273
02274
             9.57e-10, 1.027e-9, 1.097e-9, 1.193e-9, 1.334e-9, 1.47e-9,
02275
             1.636e-9, 1.871e-9, 2.122e-9, 2.519e-9, 2.806e-9, 3.203e-9,
             3.846e-9, 4.362e-9, 5.114e-9, 5.643e-9, 6.305e-9, 6.981e-9, 7.983e-9, 8.783e-9, 9.419e-9, 1.017e-8, 1.063e-8, 1.121e-8,
02276
02277
             1.13e-8, 1.201e-8, 1.225e-8, 1.232e-8, 1.223e-8, 1.177e-8,
02278
             1.151e-8, 1.116e-8, 1.047e-8, 9.698e-9, 8.734e-9, 8.202e-9,
             7.041e-9, 6.074e-9, 5.172e-9, 4.468e-9, 3.913e-9, 3.414e-9,
02280
02281
             2.975e-9, 2.65e-9, 2.406e-9, 2.173e-9, 2.009e-9, 1.861e-9,
             1.727e-9, 1.612e-9, 1.514e-9, 1.43e-9, 1.362e-9, 1.333e-9, 1.288e-9, 1.249e-9, 1.238e-9, 1.228e-9, 1.217e-9, 1.202e-9,
02282
02283
             1.209e-9, 1.177e-9, 1.157e-9, 1.165e-9, 1.142e-9, 1.131e-9,
02284
             1.138e-9, 1.117e-9, 1.1e-9, 1.069e-9, 1.023e-9, 1.005e-9,
             9.159e-10, 8.863e-10, 7.865e-10, 7.153e-10, 6.247e-10, 5.846e-10,
02286
             5.133e-10, 4.36e-10, 3.789e-10, 3.335e-10, 2.833e-10, 2.483e-10,
02287
02288
             2.155e-10, 1.918e-10, 1.709e-10, 1.529e-10, 1.374e-10, 1.235e-10,
02289
             1.108e-10,\ 9.933e-11,\ 8.932e-11,\ 8.022e-11,\ 7.224e-11,\ 6.52e-11,
             5.896e-11, 5.328e-11, 4.813e-11, 4.365e-11, 3.961e-11, 3.594e-11,
02290
             3.266e-11, 2.967e-11, 2.701e-11, 2.464e-11, 2.248e-11, 2.054e-11,
             1.878e-11, 1.721e-11, 1.579e-11, 1.453e-11, 1.341e-11, 1.241e-11,
02292
02293
             1.154e-11, 1.078e-11, 1.014e-11, 9.601e-12, 9.167e-12, 8.838e-12,
02294
             8.614e-12, 8.493e-12, 8.481e-12, 8.581e-12, 8.795e-12, 9.131e-12,
02295
             9.601e-12, 1.021e-11, 1.097e-11, 1.191e-11, 1.303e-11, 1.439e-11,
             1.601e-11, 1.778e-11, 1.984e-11, 2.234e-11, 2.474e-11, 2.766e-11,
02296
             3.085e-11, 3.415e-11, 3.821e-11, 4.261e-11, 4.748e-11, 5.323e-11,
02298
             5.935e-11, 6.619e-11, 7.418e-11, 8.294e-11, 9.26e-11, 1.039e-10,
             1.156e-10, 1.297e-10, 1.46e-10, 1.641e-10, 1.858e-10, 2.1e-10,
02299
02300
             2.383e-10, 2.724e-10, 3.116e-10, 3.538e-10, 4.173e-10, 4.727e-10,
             5.503e-10, 6.337e-10, 7.32e-10, 8.298e-10, 9.328e-10, 1.059e-9, 1.176e-9, 1.328e-9, 1.445e-9, 1.593e-9, 1.77e-9, 1.954e-9, 2.175e-9, 2.405e-9, 2.622e-9, 2.906e-9, 3.294e-9, 3.713e-9,
02301
02302
02303
             3.98e-9, 4.384e-9, 4.987e-9, 5.311e-9, 5.874e-9, 6.337e-9, 7.027e-9, 7.39e-9, 7.769e-9, 8.374e-9, 8.605e-9, 9.165e-9,
02304
02305
02306
             9.415e-9, 9.511e-9, 9.704e-9, 9.588e-9, 9.45e-9, 9.086e-9,
             8.798e-9, 8.469e-9, 7.697e-9, 7.168e-9, 6.255e-9, 5.772e-9, 4.97e-9, 4.271e-9, 3.653e-9, 3.154e-9, 2.742e-9, 2.435e-9,
02307
02308
             2.166e-9, 1.936e-9, 1.731e-9, 1.556e-9, 1.399e-9, 1.272e-9, 1.157e-9, 1.066e-9, 9.844e-10, 9.258e-10, 8.787e-10, 8.421e-10,
02309
             8.083e-10, 8.046e-10, 8.067e-10, 8.181e-10, 8.325e-10, 8.517e-10,
02311
02312
             9.151e-10, 9.351e-10, 9.677e-10, 1.071e-9, 1.126e-9, 1.219e-9,
             1.297e-9, 1.408e-9, 1.476e-9, 1.517e-9, 1.6e-9, 1.649e-9, 1.678e-9, 1.746e-9, 1.742e-9, 1.728e-9, 1.699e-9, 1.655e-9, 1.561e-9, 1.48e-9, 1.451e-9, 1.411e-9, 1.171e-9, 1.106e-9,
02313
02314
02315
```

```
9.714e-10, 8.523e-10, 7.346e-10, 6.241e-10, 5.371e-10, 4.704e-10,
             4.144e-10, 3.683e-10, 3.292e-10, 2.942e-10, 2.62e-10, 2.341e-10,
02317
02318
            2.104e-10, 1.884e-10, 1.7e-10, 1.546e-10, 1.394e-10, 1.265e-10,
02319
            1.14e-10, 1.019e-10, 9.279e-11, 8.283e-11, 7.458e-11, 6.668e-11,
02320
            5.976e-11, 5.33e-11, 4.794e-11, 4.289e-11, 3.841e-11, 3.467e-11, 3.13e-11, 2.832e-11, 2.582e-11, 2.356e-11, 2.152e-11, 1.97e-11,
02321
            1.808e-11, 1.664e-11, 1.539e-11, 1.434e-11, 1.344e-11, 1.269e-11,
            1.209e-11, 1.162e-11, 1.129e-11, 1.108e-11, 1.099e-11, 1.103e-11,
02323
02324
            1.119e-11, 1.148e-11, 1.193e-11, 1.252e-11, 1.329e-11, 1.421e-11,
            1.555e-11, 1.685e-11, 1.839e-11, 2.054e-11, 2.317e-11, 2.571e-11, 2.839e-11, 3.171e-11, 3.49e-11, 3.886e-11, 4.287e-11, 4.645e-11,
02325
02326
            5.047e-11, 5.592e-11, 6.109e-11, 6.628e-11, 7.381e-11, 8.088e-11, 8.966e-11, 1.045e-10, 1.12e-10, 1.287e-10, 1.486e-10, 1.662e-10,
02327
02328
            1.866e-10, 2.133e-10, 2.524e-10, 2.776e-10, 3.204e-10, 3.559e-10,
02329
02330
            4.028e-10, 4.448e-10, 4.882e-10, 5.244e-10, 5.605e-10, 6.018e-10,
02331
            6.328e-10, 6.579e-10, 6.541e-10, 7.024e-10, 7.074e-10, 7.068e-10,
02332
            7.009e-10, 6.698e-10, 6.545e-10, 6.209e-10, 5.834e-10, 5.412e-10,
            5.001e-10, 4.231e-10, 3.727e-10, 3.211e-10, 2.833e-10, 2.447e-10,
02333
            2.097e-10, 1.843e-10, 1.639e-10, 1.449e-10, 1.27e-10, 1.161e-10,
02335
            1.033e-10, 9.282e-11, 8.407e-11, 7.639e-11, 7.023e-11, 6.474e-11,
            6.142e-11, 5.76e-11, 5.568e-11, 5.472e-11, 5.39e-11, 5.455e-11, 5.54e-11, 5.587e-11, 6.23e-11, 6.49e-11, 6.868e-11, 7.382e-11,
02336
02337
            8.022e-11, 8.372e-11, 9.243e-11, 1.004e-10, 1.062e-10, 1.13e-10, 1.176e-10, 1.244e-10, 1.279e-10, 1.298e-10, 1.302e-10, 1.312e-10,
02338
02339
            1.295e-10, 1.244e-10, 1.211e-10, 1.167e-10, 1.098e-10, 9.927e-11,
02340
02341
            8.854e-11, 8.011e-11, 7.182e-11, 5.923e-11, 5.212e-11, 4.453e-11,
            3.832e-11, 3.371e-11, 2.987e-11, 2.651e-11, 2.354e-11, 2.093e-11,
02342
            1.863e-11, 1.662e-11, 1.486e-11, 1.331e-11, 1.193e-11, 1.071e-11, 9.628e-12, 8.66e-12, 7.801e-12, 7.031e-12, 6.347e-12, 5.733e-12,
02343
02344
            5.182e-12, 4.695e-12, 4.26e-12, 3.874e-12, 3.533e-12, 3.235e-12, 2.979e-12, 2.76e-12, 2.579e-12, 2.432e-12, 2.321e-12, 2.246e-12,
02345
02346
02347
            2.205e-12, 2.196e-12, 2.223e-12, 2.288e-12, 2.387e-12, 2.525e-12,
            2.704e-12, 2.925e-12, 3.191e-12, 3.508e-12, 3.876e-12, 4.303e-12,
02348
02349
            4.793e-12, 5.347e-12, 5.978e-12, 6.682e-12, 7.467e-12, 8.34e-12,
02350
            9.293e-12, 1.035e-11, 1.152e-11, 1.285e-11, 1.428e-11, 1.586e-11,
02351
            1.764e-11, 1.972e-11, 2.214e-11, 2.478e-11, 2.776e-11, 3.151e-11,
            3.591e-11, 4.103e-11, 4.66e-11, 5.395e-11, 6.306e-11, 7.172e-11, 8.358e-11, 9.67e-11, 1.11e-10, 1.325e-10, 1.494e-10, 1.736e-10,
02352
02354
            2.007e-10, 2.296e-10, 2.608e-10, 3.004e-10, 3.361e-10, 3.727e-10,
             4.373e-10, 4.838e-10, 5.483e-10, 6.006e-10, 6.535e-10, 6.899e-10,
02355
02356
            7.687e-10, 8.444e-10, 8.798e-10, 9.135e-10, 9.532e-10, 9.757e-10,
            8.51e-10, 1.016e-9, 9.949e-10, 9.789e-10, 9.564e-10, 9.215e-10, 8.51e-10, 8.394e-10, 7.707e-10, 7.152e-10, 6.274e-10, 5.598e-10, 5.028e-10, 4.3e-10, 3.71e-10, 3.245e-10, 2.809e-10, 2.461e-10,
02357
02358
02359
            2.154e-10, 1.91e-10, 1.685e-10, 1.487e-10, 1.313e-10, 1.163e-10,
02360
02361
            1.031e-10, 9.172e-11, 8.221e-11, 7.382e-11, 6.693e-11, 6.079e-11,
02362
            5.581e-11, 5.167e-11, 4.811e-11, 4.506e-11, 4.255e-11, 4.083e-11,
            3.949e-11, 3.881e-11, 3.861e-11, 3.858e-11, 3.951e-11, 4.045e-11, 4.24e-11, 4.487e-11, 4.806e-11, 5.133e-11, 5.518e-11, 5.919e-11, 6.533e-11, 7.031e-11, 7.762e-11, 8.305e-11, 9.252e-11, 9.727e-11,
02363
02364
02365
            1.045e-10, 1.117e-10, 1.2e-10, 1.275e-10, 1.341e-10, 1.362e-10,
02366
            1.438e-10, 1.45e-10, 1.455e-10, 1.455e-10, 1.434e-10, 1.381e-10,
02367
02368
            1.301e-10, 1.276e-10, 1.163e-10, 1.089e-10, 9.911e-11, 8.943e-11,
02369
            7.618e-11, 6.424e-11, 5.717e-11, 4.866e-11, 4.257e-11, 3.773e-11,
            3.331e-11, 2.958e-11, 2.629e-11, 2.316e-11, 2.073e-11, 1.841e-11,
02370
02371
            1.635e-11, 1.464e-11, 1.31e-11, 1.16e-11, 1.047e-11, 9.408e-12, 8.414e-12, 7.521e-12, 6.705e-12, 5.993e-12, 5.371e-12, 4.815e-12,
            4.338e-12, 3.921e-12, 3.567e-12, 3.265e-12, 3.01e-12, 2.795e-12,
02373
            2.613e-12, 2.464e-12, 2.346e-12, 2.256e-12, 2.195e-12, 2.165e-12,
02374
02375
            2.166e-12, 2.198e-12, 2.262e-12, 2.364e-12, 2.502e-12, 2.682e-12,
02376
            2.908e-12, 3.187e-12, 3.533e-12, 3.946e-12, 4.418e-12, 5.013e-12,
            5.708e-12, 6.379e-12, 7.43e-12, 8.39e-12, 9.51e-12, 1.078e-11, 1.259e-11, 1.438e-11, 1.63e-11, 1.814e-11, 2.055e-11, 2.348e-11,
02377
            2.664e-11, 2.956e-11, 3.3e-11, 3.677e-11, 4.032e-11, 4.494e-11,
02379
            4.951e-11, 5.452e-11, 6.014e-11, 6.5e-11, 6.915e-11, 7.45e-11,
02380
            7.971e-11, 8.468e-11, 8.726e-11, 8.995e-11, 9.182e-11, 9.509e-11, 9.338e-11, 9.386e-11, 9.457e-11, 9.21e-11, 9.019e-11, 8.68e-11, 8.298e-11, 7.947e-11, 7.46e-11, 7.082e-11, 6.132e-11, 5.855e-11, 5.073e-11, 4.464e-11, 3.825e-11, 3.375e-11, 2.911e-11, 2.535e-11,
02381
02382
02383
02384
             2.16e-11, 1.907e-11, 1.665e-11, 1.463e-11, 1.291e-11, 1.133e-11,
            9.997e-12, 8.836e-12, 7.839e-12, 6.943e-12, 6.254e-12, 5.6e-12,
02386
02387
            5.029e-12, 4.529e-12, 4.102e-12, 3.737e-12, 3.428e-12, 3.169e-12,
02388
            2.959e-12, 2.798e-12, 2.675e-12, 2.582e-12, 2.644e-12, 2.557e-12,
            2.614e-12, 2.717e-12, 2.874e-12, 3.056e-12, 3.187e-12, 3.631e-12,
02389
            3.979e-12, 4.248e-12, 4.817e-12, 5.266e-12, 5.836e-12, 6.365e-12, 6.807e-12, 7.47e-12, 7.951e-12, 8.636e-12, 8.972e-12, 9.314e-12,
02390
02391
            9.445e-12, 1.003e-11, 1.013e-11, 9.937e-12, 9.729e-12, 9.064e-12,
02392
02393
            9.119e-12, 9.124e-12, 8.704e-12, 8.078e-12, 7.47e-12, 6.329e-12,
02394
            5.674e-12, 4.808e-12, 4.119e-12, 3.554e-12, 3.103e-12, 2.731e-12,
            2.415e-12, 2.15e-12, 1.926e-12, 1.737e-12, 1.578e-12, 1.447e-12, 1.34e-12, 1.255e-12, 1.191e-12, 1.146e-12, 1.121e-12, 1.114e-12,
02395
02396
             1.126e-12, 1.156e-12, 1.207e-12, 1.278e-12, 1.372e-12, 1.49e-12,
            1.633e-12, 1.805e-12, 2.01e-12, 2.249e-12, 2.528e-12, 2.852e-12,
02398
02399
            3.228e-12, 3.658e-12, 4.153e-12, 4.728e-12, 5.394e-12, 6.176e-12,
02400
            7.126e-12, 8.188e-12, 9.328e-12, 1.103e-11, 1.276e-11, 1.417e-11,
            1.615e-11, 1.84e-11, 2.155e-11, 2.429e-11, 2.826e-11, 3.222e-11, 3.664e-11, 4.14e-11, 4.906e-11, 5.536e-11, 6.327e-11, 7.088e-11,
02401
02402
```

```
8.316e-11, 9.242e-11, 1.07e-10, 1.223e-10, 1.341e-10, 1.553e-10,
                   1.703e-10, 1.9e-10, 2.022e-10, 2.233e-10, 2.345e-10, 2.438e-10,
02404
02405
                  2.546e-10, 2.599e-10, 2.661e-10, 2.703e-10, 2.686e-10, 2.662e-10,
02406
                   2.56e-10, 2.552e-10, 2.378e-10, 2.252e-10, 2.146e-10, 1.885e-10,
                  1.668e-10, 1.441e-10, 1.295e-10, 1.119e-10, 9.893e-11, 8.687e-11, 7.678e-11, 6.685e-11, 5.879e-11, 5.127e-11, 4.505e-11, 3.997e-11,
02407
02408
02409
                   3.511e-11
02410
02411
              static double h2ofrn[2001] = { .01095, .01126, .01205, .01322, .0143, .01506, .01548, .01534, .01486, .01373, .01262, .01134, .01001, .008702, .007475, .006481, .00548, .0046, .003833, .00311, .002543, .002049, .00168, .001374, .001046, 8.193e-4, 6.267e-4,
02412
02413
02414
                  .002543, .002049, .00168, .001374, .001046, 8.193e-4, 6.267e-4, 4.968e-4, 3.924e-4, 2.983e-4, 2.477e-4, 1.997e-4, 1.596e-4,
02415
02416
02417
                   1.331e-4, 1.061e-4, 8.942e-5, 7.168e-5, 5.887e-5, 4.848e-5
                  3.817e-5, 3.17e-5, 2.579e-5, 2.162e-5, 1.768e-5, 1.49e-5, 1.231e-5, 1.013e-5, 8.555e-6, 7.328e-6, 6.148e-6, 5.207e-6, 4.387e-6, 3.741e-6, 3.22e-6, 2.753e-6, 2.346e-6, 1.985e-6,
02418
02419
02420
                   1.716e-6, 1.475e-6, 1.286e-6, 1.122e-6, 9.661e-7, 8.284e-7,
                  7.057e-7, 6.119e-7, 5.29e-7, 4.571e-7, 3.948e-7, 3.432e-7, 2.983e-7, 2.589e-7, 2.265e-7, 1.976e-7, 1.704e-7, 1.456e-7, 1.26e-7, 1.101e-7, 9.648e-8, 8.415e-8, 7.34e-8, 6.441e-8,
02422
02423
02424
                  5.643e-8, 4.94e-8, 4.276e-8, 3.703e-8, 3.227e-8, 2.825e-8, 2.478e-8, 2.174e-8, 1.898e-8, 1.664e-8, 1.458e-8, 1.278e-8, 1.126e-8, 9.891e-9, 8.709e-9, 7.652e-9, 6.759e-9, 5.975e-9,
02425
02426
02427
                   5.31e-9, 4.728e-9, 4.214e-9, 3.792e-9, 3.463e-9, 3.226e-9,
02428
02429
                  2.992e-9, 2.813e-9, 2.749e-9, 2.809e-9, 2.913e-9, 3.037e-9,
02430
                  3.413e-9, 3.738e-9, 4.189e-9, 4.808e-9, 5.978e-9, 7.088e-9,
                  8.07le-9, 9.6le-9, 1.2le-8, 1.5e-8, 1.764e-8, 2.22le-8, 2.898e-8, 3.948e-8, 5.068e-8, 6.227e-8, 7.898e-8, 1.033e-7, 1.437e-7,
02431
02432
                  1.889e-7, 2.589e-7, 3.59e-7, 4.971e-7, 7.156e-7, 9.983e-7, 1.381e-6, 1.929e-6, 2.591e-6, 3.453e-6, 4.57e-6, 5.93e-6,
02433
02434
02435
                   7.552e-6, 9.556e-6, 1.183e-5, 1.425e-5, 1.681e-5, 1.978e-5,
02436
                  2.335e-5, 2.668e-5, 3.022e-5, 3.371e-5, 3.715e-5, 3.967e-5,
                  4.06e-5, 4.01e-5, 3.809e-5, 3.491e-5, 3.155e-5, 2.848e-5, 2.678e-5, 2.66e-5, 2.811e-5, 3.071e-5, 3.294e-5, 3.459e-5, 3.569e-5, 3.56e-5, 3.434e-5, 3.186e-5, 2.916e-5, 2.622e-5,
02437
02438
02439
                   2.275e-5, 1.918e-5, 1.62e-5, 1.373e-5, 1.182e-5, 1.006e-5,
02441
                   8.556e-6, 7.26e-6, 6.107e-6, 5.034e-6, 4.211e-6, 3.426e-6,
02442
                   2.865e-6, 2.446e-6, 1.998e-6, 1.628e-6, 1.242e-6, 1.005e-6,
                  7.853e-7, 6.21e-7, 5.071e-7, 4.156e-7, 3.548e-7, 2.825e-7, 2.261e-7, 1.916e-7, 1.51e-7, 1.279e-7, 1.059e-7, 9.14e-8, 7.707e-8, 6.17e-8, 5.311e-8, 4.263e-8, 3.518e-8, 2.961e-8,
02443
02444
02445
                   2.457e-8, 2.119e-8, 1.712e-8, 1.439e-8, 1.201e-8, 1.003e-8,
02446
                   8.564e-9, 7.199e-9, 6.184e-9, 5.206e-9, 4.376e-9, 3.708e-9,
02447
02448
                  3.157e-9, 2.725e-9, 2.361e-9, 2.074e-9, 1.797e-9, 1.562e-9,
02449
                  1.364e-9, 1.196e-9, 1.042e-9, 8.862e-10, 7.648e-10, 6.544e-10,
                  5.609e-10, 4.791e-10, 4.108e-10, 3.531e-10, 3.038e-10, 2.618e-10,
02450
                  2.268e-10, 1.969e-10, 1.715e-10, 1.496e-10, 1.308e-10, 1.147e-10,
02451
                  1.008e-10, 8.894e-11, 7.885e-11, 7.031e-11, 6.355e-11, 5.854e-11, 5.534e-11, 5.466e-11, 5.725e-11, 6.447e-11, 7.943e-11, 1.038e-10,
02452
                   1.437e-10, 2.04e-10, 2.901e-10, 4.051e-10, 5.556e-10, 7.314e-10,
02454
02455
                   9.291e-10, 1.134e-9, 1.321e-9, 1.482e-9, 1.596e-9, 1.669e-9,
                  1.715e-9, 1.762e-9, 1.817e-9, 1.828e-9, 1.848e-9, 1.873e-9, 1.902e-9, 1.894e-9, 1.864e-9, 1.841e-9, 1.797e-9, 1.704e-9,
02456
02457
                   1.559e-9, 1.382e-9, 1.187e-9, 1.001e-9, 8.468e-10, 7.265e-10,
02458
                   6.521e-10, 6.381e-10, 6.66e-10, 7.637e-10, 9.705e-10, 1.368e-9,
                  1.856e-9, 2.656e-9, 3.954e-9, 5.96e-9, 8.72e-9, 1.247e-8,
02460
02461
                   1.781e-8, 2.491e-8, 3.311e-8, 4.272e-8, 5.205e-8, 6.268e-8,
02462
                   7.337e-8, 8.277e-8, 9.185e-8, 1.004e-7, 1.091e-7, 1.159e-7,
                  1.188e-7, 1.175e-7, 1.124e-7, 1.033e-7, 9.381e-8, 8.501e-8,
02463
                   7.956e-8, 7.894e-8, 8.331e-8, 9.102e-8, 9.836e-8, 1.035e-7,
02464
02465
                   1.064e-7, 1.06e-7, 1.032e-7, 9.808e-8, 9.139e-8, 8.442e-8,
                   7.641e-8, 6.881e-8, 6.161e-8, 5.404e-8, 4.804e-8, 4.446e-8
02466
02467
                   4.328e-8, 4.259e-8, 4.421e-8, 4.673e-8, 4.985e-8, 5.335e-8,
02468
                  5.796e-8, 6.542e-8, 7.714e-8, 8.827e-8, 1.04e-7, 1.238e-7,
                  1.499e-7, 1.829e-7, 2.222e-7, 2.689e-7, 3.303e-7, 3.981e-7, 4.84e-7, 5.91e-7, 7.363e-7, 9.087e-7, 1.139e-6, 1.455e-6,
02469
02470
                  1.866e-6, 2.44e-6, 3.115e-6, 3.941e-6, 4.891e-6, 5.992e-6, 7.111e-6, 8.296e-6, 9.21e-6, 9.987e-6, 1.044e-5, 1.073e-5,
02471
02473
                   1.092e-5, 1.106e-5, 1.138e-5, 1.171e-5, 1.186e-5, 1.186e-5,
02474
                  1.179e-5, 1.166e-5, 1.151e-5, 1.16e-5, 1.197e-5, 1.241e-5,
02475
                  1.268e-5, 1.26e-5, 1.184e-5, 1.063e-5, 9.204e-6, 7.584e-6,
                   6.053e-6, 4.482e-6, 3.252e-6, 2.337e-6, 1.662e-6, 1.18e-6,
02476
                  8.15e-7, 5.95e-7, 4.354e-7, 3.302e-7, 2.494e-7, 1.93e-7, 1.545e-7, 1.25e-7, 1.039e-7, 8.602e-8, 7.127e-8, 5.897e-8, 4.838e-8, 4.018e-8, 3.28e-8, 2.72e-8, 2.307e-8, 1.972e-8,
02477
02478
02479
02480
                   1.654e-8, 1.421e-8, 1.174e-8, 1.004e-8, 8.739e-9, 7.358e-9,
                  6.242e-9, 5.303e-9, 4.567e-9, 3.94e-9, 3.375e-9, 2.864e-9, 2.422e-9, 2.057e-9, 1.75e-9, 1.505e-9, 1.294e-9, 1.101e-9,
02481
02482
                  3.759e-10, 3.247e-10, 2.809e-10, 2.438e-10, 2.123e-10, 1.853e-10, 3.759e-10, 3.247e-10, 2.809e-10, 2.438e-10, 2.123e-10, 1.853e-10, 2.438e-10, 2.123e-10, 1.853e-10, 2.438e-10, 
02483
                   1.622e-10, 1.426e-10, 1.26e-10, 1.125e-10, 1.022e-10, 9.582e-11,
02485
02486
                   9.388e-11, 9.801e-11, 1.08e-10, 1.276e-10, 1.551e-10, 1.903e-10,
02487
                  2.291 e^{-10},\ 2.724 e^{-10},\ 3.117 e^{-10},\ 3.4 e^{-10},\ 3.562 e^{-10},\ 3.625 e^{-10},
                  3.619e-10, 3.429e-10, 3.221e-10, 2.943e-10, 2.645e-10, 2.338e-10, 2.062e-10, 1.901e-10, 1.814e-10, 1.827e-10, 1.906e-10, 1.984e-10,
02488
02489
```

```
2.04e-10, 2.068e-10, 2.075e-10, 2.018e-10, 1.959e-10, 1.897e-10,
                  1.852e-10, 1.791e-10, 1.696e-10, 1.634e-10, 1.598e-10, 1.561e-10,
02491
02492
                  1.518e-10, 1.443e-10, 1.377e-10, 1.346e-10, 1.342e-10, 1.375e-10,
02493
                  1.525e-10, 1.767e-10, 2.108e-10, 2.524e-10, 2.981e-10, 3.477e-10,
02494
                  4.262e-10, 5.326e-10, 6.646e-10, 8.321e-10, 1.069e-9, 1.386e-9,
                   1.743e-9, 2.216e-9, 2.808e-9, 3.585e-9, 4.552e-9, 5.907e-9,
02495
                  7.611e-9, 9.774e-9, 1.255e-8, 1.666e-8, 2.279e-8, 3.221e-8,
                  4.531e-8, 6.4e-8, 9.187e-8, 1.295e-7, 1.825e-7, 2.431e-7,
02497
                  3.181e-7, 4.009e-7, 4.941e-7, 5.88e-7, 6.623e-7, 7.155e-7, 7.451e-7, 7.594e-7, 7.541e-7, 7.467e-7, 7.527e-7, 7.935e-7, 8.461e-7, 8.954e-7, 9.364e-7, 9.843e-7, 1.024e-6, 1.05e-6,
02498
02499
02500
                  1.059e-6, 1.074e-6, 1.072e-6, 1.043e-6, 9.789e-7, 8.803e-7, 7.662e-7, 6.378e-7, 5.133e-7, 3.958e-7, 2.914e-7, 2.144e-7,
02501
02502
                  1.57e-7, 1.14e-7, 8.47e-8, 6.2e-8, 4.657e-8, 3.559e-8, 2.813e-8,
02503
02504
                  2.222e-8, 1.769e-8, 1.391e-8, 1.125e-8, 9.186e-9, 7.704e-9,
                  6.447e-9, 5.381e-9, 4.442e-9, 3.669e-9, 3.057e-9, 2.564e-9, 2.153e-9, 1.784e-9, 1.499e-9, 1.281e-9, 1.082e-9, 9.304e-10,
02505
02506
                  8.169e-10, 6.856e-10, 5.866e-10, 5.043e-10, 4.336e-10, 3.731e-10, 3.175e-10, 2.745e-10, 2.374e-10, 2.007e-10, 1.737e-10, 1.508e-10,
02507
                  1.302e-10, 1.13e-10, 9.672e-11, 8.375e-11, 7.265e-11, 6.244e-11,
                  5.343e-11, 4.654e-11, 3.975e-11, 3.488e-11, 3.097e-11, 2.834e-11, 2.649e-11, 2.519e-11, 2.462e-11, 2.443e-11, 2.44e-11, 2.398e-11,
02510
02511
                  2.306e-11, 2.183e-11, 2.021e-11, 1.821e-11, 1.599e-11, 1.403e-11, 1.196e-11, 1.023e-11, 8.728e-12, 7.606e-12, 6.941e-12, 6.545e-12,
02512
02513
                  6.484e-12, 6.6e-12, 6.718e-12, 6.785e-12, 6.746e-12, 6.724e-12,
                  6.764e-12, 6.995e-12, 7.144e-12, 7.32e-12, 7.33e-12, 7.208e-12, 6.789e-12, 6.09e-12, 5.337e-12, 4.62e-12, 4.037e-12, 3.574e-12,
02516
02517
                  3.311e-12, 3.346e-12, 3.566e-12, 3.836e-12, 4.076e-12, 4.351e-12,
                 1. 3.54e 12, 5.54e 12, 5.54e 12, 5.65e 12, 7.56e 12, 7.56e 12, 7.56e 12, 7.55e 12, 7.5
02518
02519
02520
                                                                                                   7.335e-10, 8.98e-10,
                  1.11e-9, 1.363e-9, 1.677e-9, 2.104e-9, 2.681e-9, 3.531e-9,
02522
02523
                  4.621e-9, 6.106e-9, 8.154e-9, 1.046e-8, 1.312e-8, 1.607e-8,
                  1.948e-8, 2.266e-8, 2.495e-8, 2.655e-8, 2.739e-8, 2.739e-8, 2.662e-8, 2.589e-8, 2.59e-8, 2.664e-8, 2.833e-8, 3.023e-8,
02524
02525
                  3.305e-8, 3.558e-8, 3.793e-8, 3.961e-8, 4.056e-8, 4.102e-8,
02526
                  4.025e-8, 3.917e-8, 3.706e-8, 3.493e-8, 3.249e-8, 3.096e-8,
                  3.011e-8, 3.111e-8, 3.395e-8, 3.958e-8, 4.875e-8, 6.066e-8,
                 7.915e-8, 1.011e-7, 1.3e-7, 1.622e-7, 2.003e-7, 2.448e-7, 2.863e-7, 3.317e-7, 3.655e-7, 3.96e-7, 4.098e-7, 4.168e-7, 4.198e-7, 4.207e-7, 4.289e-7, 4.384e-7, 4.471e-7, 4.524e-7,
02529
02530
02531
                  4.574e-7, 4.633e-7, 4.785e-7, 5.028e-7, 5.371e-7, 5.727e-7, 5.955e-7, 5.998e-7, 5.669e-7, 5.082e-7, 4.397e-7, 3.596e-7,
02532
                  2.814e-7, 2.074e-7, 1.486e-7, 1.057e-7, 7.25e-8, 4.946e-8, 3.43e-8, 2.447e-8, 1.793e-8, 1.375e-8, 1.096e-8, 9.091e-9,
02535
                  7.709e-9, 6.631e-9, 5.714e-9, 4.886e-9, 4.205e-9, 3.575e-9, 3.07e-9, 2.631e-9, 2.284e-9, 2.002e-9, 1.745e-9, 1.509e-9,
02536
02537
                  1.284e-9, 1.084e-9, 9.163e-10, 7.663e-10, 6.346e-10, 5.283e-10,
4.354e-10, 3.59e-10, 2.982e-10, 2.455e-10, 2.033e-10, 1.696e-10,
1.432e-10, 1.211e-10, 1.02e-10, 8.702e-11, 7.38e-11, 6.293e-11,
02538
02539
                  5.343e-11, 4.532e-11, 3.907e-11, 3.365e-11, 2.945e-11, 2.558e-11,
02541
02542
                  2.192e-11, 1.895e-11, 1.636e-11, 1.42e-11, 1.228e-11, 1.063e-11,
                  9.348e-12, 8.2e-12, 7.231e-12, 6.43e-12, 5.702e-12, 5.052e-12, 4.469e-12, 4e-12, 3.679e-12, 3.387e-12, 3.197e-12, 3.158e-12,
02543
02544
                  3.327e-12, 3.675e-12, 4.292e-12, 5.437e-12, 7.197e-12, 1.008e-11,
02545
                  1.437e-11, 2.035e-11, 2.905e-11, 4.062e-11, 5.528e-11, 7.177e-11,
                  9.064e-11, 1.109e-10, 1.297e-10, 1.473e-10, 1.652e-10, 1.851e-10,
02547
02548
                  2.079e-10, 2.313e-10, 2.619e-10, 2.958e-10, 3.352e-10, 3.796e-10,
02549
                  4.295e-10, 4.923e-10, 5.49e-10, 5.998e-10, 6.388e-10, 6.645e-10,
                  6.712e-10, 6.549e-10, 6.38e-10, 6.255e-10, 6.253e-10, 6.459e-10, 6.977e-10, 7.59e-10, 8.242e-10, 8.92e-10, 9.403e-10, 9.701e-10, 9.483e-10, 9.135e-10, 8.617e-10, 7.921e-10, 7.168e-10, 6.382e-10,
02550
02551
02553
                  5.677e-10, 5.045e-10, 4.572e-10, 4.312e-10, 4.145e-10, 4.192e-10,
02554
                  4.541e-10, 5.368e-10, 6.771e-10, 8.962e-10, 1.21e-9, 1.659e-9,
02555
                  2.33e-9, 3.249e-9, 4.495e-9, 5.923e-9, 7.642e-9, 9.607e-9,
02556
                  1.178e-8, 1.399e-8, 1.584e-8, 1.73e-8, 1.816e-8, 1.87e-8, 1.868e-8, 1.87e-8, 1.884e-8, 1.99e-8, 2.15e-8, 2.258e-8,
02557
                  2.364e-8, 2.473e-8, 2.602e-8, 2.689e-8, 2.731e-8, 2.816e-8,
02558
                  2.859e-8, 2.839e-8, 2.703e-8, 2.451e-8, 2.149e-8, 1.787e-8,
                  1.449e-8, 1.111e-8, 8.282e-9, 6.121e-9, 4.494e-9, 3.367e-9,
02560
02561
                  2.487e-9, 1.885e-9, 1.503e-9, 1.249e-9, 1.074e-9, 9.427e-10,
                  8.439e-10, 7.563e-10, 6.772e-10, 6.002e-10, 5.254e-10, 4.588e-10, 3.977e-10, 3.449e-10, 3.003e-10, 2.624e-10, 2.335e-10, 2.04e-10,
02562
02563
02564
                  1.771e-10, 1.534e-10, 1.296e-10, 1.097e-10, 9.173e-11, 7.73e-11,
                  6.547e-11, 5.191e-11, 4.198e-11, 3.361e-11, 2.732e-11, 2.244e-11
02565
                  1.791e-11, 1.509e-11, 1.243e-11, 1.035e-11, 8.969e-12, 7.394e-12,
02566
02567
                  6.323e-12, 5.282e-12, 4.543e-12, 3.752e-12, 3.14e-12, 2.6e-12,
                  2.194e-12, 1.825e-12, 1.511e-12, 1.245e-12, 1.024e-12, 8.539e-13, 7.227e-13, 6.102e-13, 5.189e-13, 4.43e-13, 3.774e-13, 3.236e-13, 2.8e-13, 2.444e-13, 2.156e-13, 1.932e-13, 1.775e-13, 1.695e-13,
02568
02569
02570
                  1.672e-13, 1.704e-13, 1.825e-13, 2.087e-13, 2.614e-13, 3.377e-13,
                  4.817e-13, 6.89e-13, 1.062e-12, 1.562e-12, 2.288e-12, 3.295e-12, 4.55e-12, 5.965e-12, 7.546e-12, 9.395e-12, 1.103e-11, 1.228e-11,
02572
02573
02574
                  1.318e-11, 1.38e-11, 1.421e-11, 1.39e-11, 1.358e-11, 1.336e-11,
                  1.342e-11, 1.356e-11, 1.424e-11, 1.552e-11, 1.73e-11, 1.951e-11, 2.128e-11, 2.249e-11, 2.277e-11, 2.226e-11, 2.111e-11, 1.922e-11,
02575
02576
```

```
1.775e-11, 1.661e-11, 1.547e-11, 1.446e-11, 1.323e-11, 1.21e-11,
            1.054e-11, 9.283e-12, 8.671e-12, 8.67e-12, 9.429e-12, 1.062e-11, 1.255e-11, 1.506e-11, 1.818e-11, 2.26e-11, 2.831e-11, 3.723e-11,
02578
02579
02580
            5.092 e-11, \ 6.968 e-11, \ 9.826 e-11, \ 1.349 e-10, \ 1.87 e-10, \ 2.58 e-10,
            3.43e-10, 4.424e-10, 5.521e-10, 6.812e-10, 8.064e-10, 9.109e-10, 9.839e-10, 1.028e-9, 1.044e-9, 1.029e-9, 1.005e-9, 1.002e-9,
02581
02582
            1.038e-9, 1.122e-9, 1.233e-9, 1.372e-9, 1.524e-9, 1.665e-9,
            1.804e-9, 1.908e-9, 2.015e-9, 2.117e-9, 2.219e-9, 2.336e-9,
02584
02585
            2.531e-9, 2.805e-9, 3.189e-9, 3.617e-9, 4.208e-9, 4.911e-9,
            5.619e-9, 6.469e-9, 7.188e-9, 7.957e-9, 8.503e-9, 9.028e-9, 9.571e-9, 9.99e-9, 1.055e-8, 1.102e-8, 1.132e-8, 1.141e-8,
02586
02587
02588
            1.145e-8, 1.145e-8, 1.176e-8, 1.224e-8, 1.304e-8, 1.388e-8,
            1.445e-8, 1.453e-8, 1.368e-8, 1.22e-8, 1.042e-8, 8.404e-9,
            6.403e-9, 4.643e-9, 3.325e-9, 2.335e-9, 1.638e-9, 1.19e-9,
02590
02591
            9.161e-10, 7.412e-10, 6.226e-10, 5.516e-10, 5.068e-10, 4.831e-10,
            4.856e-10, 5.162e-10, 5.785e-10, 6.539e-10, 7.485e-10, 8.565
9.534e-10, 1.052e-9, 1.115e-9, 1.173e-9, 1.203e-9, 1.224e-9,
                                                                                8.565e-10,
02592
02593
            1.243e-9, 1.248e-9, 1.261e-9, 1.265e-9, 1.25e-9, 1.277e-9, 1.176e-9, 1.145e-9, 1.153e-9, 1.199e-9, 1.278e-9, 1.366e-9,
02594
02596
            1.426e-9, 1.444e-9, 1.365e-9, 1.224e-9, 1.051e-9, 8.539e-10,
            6.564e-10, 4.751e-10, 3.404e-10, 2.377e-10, 1.631e-10, 1.114e-10,
02597
02598
            7.87e-11, 5.793e-11, 4.284e-11, 3.3e-11, 2.62e-11, 2.152e-11,
02599
            1.777e-11, 1.496e-11, 1.242e-11, 1.037e-11, 8.725e-12, 7.004e-12,
            5.718e-12, 4.769e-12, 3.952e-12, 3.336e-12, 2.712e-12, 2.213e-12, 1.803e-12, 1.492e-12, 1.236e-12, 1.006e-12, 8.384e-13, 7.063e-13,
02600
02601
            5.879e-13, 4.93e-13, 4.171e-13, 3.569e-13, 3.083e-13, 2.688e-13,
02602
            2.333e-13, 2.035e-13, 1.82e-13, 1.682e-13, 1.635e-13, 1.674e-13,
02603
02604
            1.769e-13, 2.022e-13, 2.485e-13, 3.127e-13, 4.25e-13, 5.928e-13,
02605
            8.514e-13, 1.236e-12, 1.701e-12, 2.392e-12, 3.231e-12, 4.35e-12,
            5.559e-12, 6.915e-12, 8.519e-12, 1.013e-11, 1.146e-11, 1.24e-11,
02606
02607
            1.305e-11, 1.333e-11, 1.318e-11, 1.263e-11, 1.238e-11, 1.244e-11,
02608
            1.305e-11, 1.432e-11, 1.623e-11, 1.846e-11, 2.09e-11, 2.328e-11,
            2.526e-11, 2.637e-11, 2.702e-11, 2.794e-11, 2.889e-11, 2.989e-11, 3.231e-11, 3.68e-11, 4.375e-11, 5.504e-11, 7.159e-11, 9.502e-11,
02609
02610
02611
            1.279e-10, 1.645e-10, 2.098e-10, 2.618e-10, 3.189e-10, 3.79e-10,
            4.303e-10, 4.753e-10, 5.027e-10, 5.221e-10, 5.293e-10, 5.346e-10,
02612
            5.467e-10, 5.796e-10, 6.2e-10, 6.454e-10, 6.705e-10, 6.925e-10, 7.233e-10, 7.35e-10, 7.538e-10, 7.861e-10, 8.077e-10, 8.132e-10,
02613
02615
            7.749e-10, 7.036e-10, 6.143e-10, 5.093e-10, 4.089e-10, 3.092e-10,
            2.299e-10, 1.705e-10, 1.277e-10, 9.723e-11, 7.533e-11, 6.126e-11,
02616
02617
            5.154e-11, 4.428e-11, 3.913e-11, 3.521e-11, 3.297e-11, 3.275e-11,
            3.46e-11, 3.798e-11, 4.251e-11, 4.745e-11, 5.232e-11, 5.606e-11, 5.82e-11, 5.88e-11, 5.79e-11, 5.661e-11, 5.491e-11, 5.366e-11, 5.341e-11, 5.353e-11, 5.336e-11, 5.293e-11, 5.248e-11, 5.235e-11,
02618
02619
02620
            5.208e-11, 5.322e-11, 5.521e-11, 5.725e-11, 5.827e-11, 5.685e-11,
02621
02622
            5.245e-11, 4.612e-11, 3.884e-11, 3.129e-11, 2.404e-11, 1.732e-11,
02623
            1.223e-11, 8.574e-12, 5.888e-12, 3.986e-12, 2.732e-12, 1.948e-12,
02624
            1.414e-12, 1.061e-12, 8.298e-13, 6.612e-13, 5.413e-13, 4.472e-13,
            3.772e-13, 3.181e-13, 2.645e-13, 2.171e-13, 1.778e-13, 1.464e-13,
02625
02626
            1.183e-13, 9.637e-14, 7.991e-14, 6.668e-14, 5.57e-14, 4.663e-14,
            3.848e-14, 3.233e-14, 2.706e-14, 2.284e-14, 1.944e-14, 1.664e-14,
            1.43e-14, 1.233e-14, 1.066e-14, 9.234e-15, 8.023e-15, 6.993e-15,
02628
02629
            6.119e-15, 5.384e-15, 4.774e-15, 4.283e-15, 3.916e-15, 3.695e-15,
02630
            3.682e-15, 4.004e-15, 4.912e-15, 6.853e-15, 1.056e-14, 1.712e-14,
            2.804e-14, 4.516e-14, 7.113e-14, 1.084e-13, 1.426e-13, 1.734e-13,
02631
            1.978e-13, 2.194e-13, 2.388e-13, 2.489e-13, 2.626e-13, 2.865e-13,
02632
            3.105e-13, 3.387e-13, 3.652e-13, 3.984e-13, 4.398e-13, 4.906e-13,
            5.55e-13, 6.517e-13, 7.813e-13, 9.272e-13, 1.164e-12, 1.434e-12,
02634
            1.849e-12, 2.524e-12, 3.328e-12, 4.523e-12, 6.108e-12, 8.207e-12,
02635
02636
            1.122e-11, 1.477e-11, 1.9e-11, 2.412e-11, 2.984e-11, 3.68e-11,
            4.353e-11, 4.963e-11, 5.478e-11, 5.903e-11, 6.233e-11, 6.483e-11, 6.904e-11, 7.569e-11, 8.719e-11, 1.048e-10, 1.278e-10, 1.557e-10, 1.869e-10, 2.218e-10, 2.61e-10, 2.975e-10, 3.371e-10, 3.746e-10,
02637
02638
02639
            4.065e-10, 4.336e-10, 4.503e-10, 4.701e-10, 4.8e-10, 4.917e-10,
02640
02641
            5.038e-10, 5.128e-10, 5.143e-10, 5.071e-10, 5.019e-10, 5.025e-10,
02642
            5.183e-10, 5.496e-10, 5.877e-10, 6.235e-10, 6.42e-10, 6.234e-10,
            5.698e-10, 4.916e-10, 4.022e-10, 3.126e-10, 2.282e-10, 1.639e-10, 1.142e-10, 7.919e-11, 5.69e-11, 4.313e-11, 3.413e-11, 2.807e-11, 2.41e-11, 2.166e-11, 2.024e-11, 1.946e-11, 1.929e-11, 1.963e-11,
02643
02644
02645
            2.035e-11, 2.162e-11, 2.305e-11, 2.493e-11, 2.748e-11, 3.048e-11,
            3.413e-11, 3.754e-11, 4.155e-11, 4.635e-11, 5.11e-11, 5.734e-11, 6.338e-11, 6.99e-11, 7.611e-11, 8.125e-11, 8.654e-11, 8.951e-11,
02647
02648
02649
            9.182e-11, 9.31e-11, 9.273e-11, 9.094e-11, 8.849e-11, 8.662e-11,
02650
            8.67e-11, 8.972e-11, 9.566e-11, 1.025e-10, 1.083e-10, 1.111e-10,
            1.074e-10, 9.771e-11, 8.468e-11, 6.958e-11, 5.47e-11, 4.04e-11,
02651
            2.94e-11, 2.075e-11, 1.442e-11, 1.01e-11, 7.281e-12, 5.409e-12,
02652
            4.138e-12, 3.304e-12, 2.784e-12, 2.473e-12, 2.273e-12, 2.186e-12,
02653
02654
            2.118e-12, 2.066e-12, 1.958e-12, 1.818e-12, 1.675e-12, 1.509e-12,
02655
            1.349e-12, 1.171e-12, 9.838e-13, 8.213e-13, 6.765e-13, 5.378e-13,
            4.161e-13, 3.119e-13, 2.279e-13, 1.637e-13, 1.152e-13, 8.112e-14,
02656
            1.132e-13, 1.132e-13, 1.132e-13, 1.132e-13, 1.132e-13, 1.132e-13, 1.132e-14, 1.948e-14, 1.66e-14, 1.432e-14, 1.251e-14, 1.109e-14, 1.006e-14, 9.45e-15,
02657
            9.384e-15, 1.012e-14, 1.216e-14, 1.636e-14, 2.305e-14, 3.488e-14,
02659
02660
            5.572e-14, 8.479e-14, 1.265e-13, 1.905e-13, 2.73e-13, 3.809e-13,
02661
            4.955e-13, 6.303e-13, 7.861e-13, 9.427e-13, 1.097e-12, 1.212e-12,
            1.328e-12, 1.415e-12, 1.463e-12, 1.495e-12, 1.571e-12, 1.731e-12, 1.981e-12, 2.387e-12, 2.93e-12, 3.642e-12, 4.584e-12, 5.822e-12,
02662
02663
```

```
7.278e-12, 9.193e-12, 1.135e-11, 1.382e-11, 1.662e-11, 1.958e-11,
                2.286e-11, 2.559e-11, 2.805e-11, 2.988e-11, 3.106e-11, 3.182e-11,
02665
02666
                3.2e-11, 3.258e-11, 3.362e-11, 3.558e-11, 3.688e-11, 3.8e-11,
02667
                3.929 e^{-11},\ 4.062 e^{-11},\ 4.186 e^{-11},\ 4.293 e^{-11},\ 4.48 e^{-11},\ 4.643 e^{-11},
02668
                4.704e-11, 4.571e-11, 4.206e-11, 3.715e-11, 3.131e-11, 2.541e-11,
                1.978e-11, 1.508e-11, 1.146e-11, 8.7e-12, 6.603e-12, 5.162e-12,
02669
                4.157e-12, 3.408e-12, 2.829e-12, 2.405e-12, 2.071e-12, 1.826e-12,
                1.648e-12, 1.542e-12, 1.489e-12, 1.485e-12, 1.493e-12, 1.545e-12,
02671
02672
                1.637e-12, 1.814e-12, 2.061e-12, 2.312e-12, 2.651e-12, 3.03e-12,
               3.46e-12, 3.901e-12, 4.306e-12, 4.721e-12, 5.008e-12, 5.281e-12, 5.541e-12, 5.791e-12, 6.115e-12, 6.442e-12, 6.68e-12, 6.791e-12, 6.831e-12, 6.839e-12, 6.946e-12, 7.128e-12, 7.537e-12, 8.036e-12, 8.392e-12, 8.526e-12, 8.11e-12, 7.325e-12, 6.329e-12, 5.183e-12,
02673
02674
02675
02676
                4.081e-12, 2.985e-12, 2.141e-12, 1.492e-12, 1.015e-12, 6.684e-13,
02677
02678
                4.414e-13, 2.987e-13, 2.038e-13, 1.391e-13, 9.86e-14, 7.24e-14,
02679
                5.493e-14, 4.288e-14, 3.427e-14, 2.787e-14, 2.296e-14, 1.909e-14,
               1.598e-14, 1.344e-14, 1.135e-14, 9.616e-15, 8.169e-15, 6.957e-15, 5.938e-15, 5.08e-15, 4.353e-15, 3.738e-15, 3.217e-15, 2.773e-15, 2.397e-15, 2.077e-15, 1.805e-15, 1.575e-15, 1.382e-15, 1.221e-15,
02680
02681
                1.09e-15, 9.855e-16, 9.068e-16, 8.537e-16, 8.27e-16, 8.29e-16,
                8.634e-16, 9.359e-16, 1.055e-15, 1.233e-15, 1.486e-15, 1.839e-15,
02684
02685
                2.326e-15, 2.998e-15, 3.934e-15, 5.256e-15, 7.164e-15, 9.984e-15,
02686
                1.427e-14, 2.099e-14, 3.196e-14, 5.121e-14, 7.908e-14, 1.131e-13,
                1.602e-13, 2.239e-13, 3.075e-13, 4.134e-13, 5.749e-13, 7.886e-13, 1.071e-12, 1.464e-12, 2.032e-12, 2.8e-12, 3.732e-12, 4.996e-12,
02687
02688
                6.483e-12, 8.143e-12, 1.006e-11, 1.238e-11, 1.484e-11, 1.744e-11,
                2.02e-11, 2.274e-11, 2.562e-11, 2.848e-11, 3.191e-11, 3.617e-11,
02690
02691
                4.081e-11, 4.577e-11, 4.937e-11, 5.204e-11, 5.401e-11, 5.462e-11,
               5.507e-11, 5.51e-11, 5.605e-11, 5.686e-11, 5.739e-11, 5.766e-11, 5.74e-11, 5.754e-11, 5.761e-11, 5.777e-11, 5.712e-11, 5.51e-11, 5.088e-11, 4.438e-11, 3.728e-11, 2.994e-11, 2.305e-11, 1.715e-11,
02692
02693
02694
02695
                1.256e-11, 9.208e-12, 6.745e-12, 5.014e-12, 3.785e-12, 2.9e-12,
                2.239e-12, 1.757e-12, 1.414e-12, 1.142e-12, 9.482e-13, 8.01e-13,
02696
02697
                6.961e-13, 6.253e-13, 5.735e-13, 5.433e-13, 5.352e-13, 5.493e-13,
                5.706e-13, 6.068e-13, 6.531e-13, 7.109e-13, 7.767e-13, 8.59e-13, 9.792e-13, 1.142e-12, 1.371e-12, 1.65e-12, 1.957e-12, 2.302e-12,
02698
02699
                2.705e-12, 3.145e-12, 3.608e-12, 4.071e-12, 4.602e-12, 5.133e-12,
02700
                5.572e-12, 5.987e-12, 6.248e-12, 6.533e-12, 6.757e-12, 6.935e-12,
02702
                7.224e-12, 7.422e-12, 7.538e-12, 7.547e-12, 7.495e-12, 7.543e-12,
02703
                7.725e-12, 8.139e-12, 8.627e-12, 9.146e-12, 9.443e-12, 9.318e-12,
02704
                8.649e-12, 7.512e-12, 6.261e-12, 4.915e-12, 3.647e-12, 2.597e-12,
               1.785e-12, 1.242e-12, 8.66e-13, 6.207e-13, 4.61e-13, 3.444e-13, 2.634e-13, 2.1e-13, 1.725e-13, 1.455e-13, 1.237e-13, 1.085e-13,
02705
02706
02707
                9.513e-14, 7.978e-14, 6.603e-14, 5.288e-14, 4.084e-14, 2.952e-14,
                2.157e-14, 1.593e-14, 1.199e-14, 9.267e-15, 7.365e-15, 6.004e-15,
02708
02709
                4.995e-15, 4.218e-15, 3.601e-15, 3.101e-15, 2.692e-15, 2.36e-15,
02710
                2.094e-15, 1.891e-15, 1.755e-15, 1.699e-15, 1.755e-15,
                                                                                                         1.987e-15,
02711
                2.506e-15, 3.506e-15, 5.289e-15, 8.311e-15, 1.325e-14, 2.129e-14,
                3.237e-14, 4.595e-14, 6.441e-14, 8.433e-14, 1.074e-13, 1.383e-13,
02712
02713
                1.762e-13, 2.281e-13, 2.831e-13, 3.523e-13, 4.38e-13, 5.304e-13,
                6.29e-13, 7.142e-13, 8.032e-13, 8.934e-13, 9.888e-13, 1.109e-12,
                1.261e-12, 1.462e-12, 1.74e-12, 2.099e-12, 2.535e-12, 3.008e-12,
02715
02716
                3.462e-12, 3.856e-12, 4.098e-12, 4.239e-12, 4.234e-12, 4.132e-12,
02717
                3.986e-12, 3.866e-12, 3.829e-12, 3.742e-12, 3.705e-12, 3.694e-12,
02718
                3.765e-12, 3.849e-12, 3.929e-12, 4.056e-12, 4.092e-12, 4.047e-12,
                3.792e-12, 3.407e-12, 2.953e-12, 2.429e-12, 1.931e-12, 1.46e-12,
02719
                1.099e-12, 8.199e-13, 6.077e-13, 4.449e-13, 3.359e-13, 2.524e-13,
                1.881e-13, 1.391e-13, 1.02e-13, 7.544e-14, 5.555e-14, 4.22e-14,
02721
                3.321e-14, 2.686e-14, 2.212e-14, 1.78e-14, 1.369e-14, 1.094e-14, 9.13e-15, 8.101e-15, 7.828e-15, 8.393e-15, 1.012e-14, 1.259e-14,
02722
02723
02724
                1.538e-14, 1.961e-14, 2.619e-14, 3.679e-14, 5.049e-14, 6.917e-14,
                1.336e-14, 1.115e-13, 2.013e-14, 3.073e-14, 
02725
02727
                3.162e-13, 3.36e-13, 3.491e-13, 3.541e-13, 3.595e-13, 3.608e-13,
02728
                3.709e-13, 3.869e-13, 4.12e-13, 4.366e-13, 4.504e-13, 4.379e-13,
02729
                3.955e-13, 3.385e-13, 2.741e-13, 2.089e-13, 1.427e-13, 9.294e-14,
                5.775e-14, 3.565e-14, 2.21e-14, 1.398e-14, 9.194e-15, 6.363e-15, 4.644e-15, 3.55e-15, 2.808e-15, 2.274e-15, 1.871e-15, 1.557e-15,
02730
02731
02732
                1.308e-15, 1.108e-15, 9.488e-16, 8.222e-16, 7.238e-16, 6.506e-16,
                6.008e-16, 5.742e-16, 5.724e-16, 5.991e-16, 6.625e-16, 7.775e-16,
                9.734e-16, 1.306e-15, 1.88e-15, 2.879e-15, 4.616e-15, 7.579e-15,
02734
02735
                1.248e-14, 2.03e-14, 3.244e-14, 5.171e-14, 7.394e-14, 9.676e-14,
                1.199e-13, 1.467e-13, 1.737e-13, 2.02e-13, 2.425e-13, 3.016e-13, 3.7e-13, 4.617e-13, 5.949e-13, 7.473e-13, 9.378e-13, 1.191e-12,
02736
02737
                1.481e-12, 1.813e-12, 2.232e-12, 2.722e-12, 3.254e-12, 3.845e-12, 4.458e-12, 5.048e-12, 5.511e-12, 5.898e-12, 6.204e-12, 6.293e-12,
02738
02739
                6.386e-12, 6.467e-12, 6.507e-12, 6.466e-12, 6.443e-12, 6.598e-12,
02740
02741
                6.873e-12, 7.3e-12, 7.816e-12, 8.368e-12, 8.643e-12, 8.466e-12,
02742
                7.871e-12, 6.853e-12, 5.714e-12, 4.482e-12, 3.392e-12, 2.613e-12,
                2.008e-12, 1.562e-12, 1.228e-12, 9.888e-13, 7.646e-13, 5.769e-13, 4.368e-13, 3.324e-13, 2.508e-13, 1.916e-13
02743
02744
02745
02746
02747
             static double xfcrev[15] =
02748
                { 1.003, 1.009, 1.015, 1.023, 1.029, 1.033, 1.037,
                1.039, 1.04, 1.046, 1.036, 1.027, 1.01, 1.002, 1.
02749
02750
```

```
02752
         double a1, a2, a3, dw, ew, dx, xw, xx, vf2, vf6, cw260, cw296,
02753
           sfac, fscal, cwfrn, ctmpth, ctwfrn, ctwslf;
02754
02755
02756
02757
         /* Get H2O continuum absorption... */
02758
         xw = nu / 10 + 1;
02759
         if (xw >= 1 && xw < 2001) {
02760
          iw = (int) xw;
           dw = xw - iw;
ew = 1 - dw;
02761
02762
           cw296 = ew * h2o296[iw - 1] + dw * h2o296[iw];

cw260 = ew * h2o260[iw - 1] + dw * h2o260[iw];

cwfrn = ew * h2ofrn[iw - 1] + dw * h2ofrn[iw];
02763
02764
02765
02766
           if (nu <= 820 || nu >= 960) {
02767
             sfac = 1;
02768
           } else {
            xx = (nu - 820) / 10;
02770
              ix = (int) xx;
02771
              dx = xx - ix;
02772
             sfac = (1 - dx) * xfcrev[ix] + dx * xfcrev[ix + 1];
02773
02774
           ctwslf = sfac * cw296 * pow(cw260 / cw296, (296 - t) / (296 - 260));
02775
           vf2 = POW2 (nu - 370);
02776
           vf6 = POW3(vf2);
02777
           fscal = 36100 / (vf2 + vf6 * 1e-8 + 36100) * -.25 + 1;
02778
           ctwfrn = cwfrn * fscal;
           a1 = nu * u * tanh(.7193876 / t * nu);
a2 = 296 / t;
02779
02780
           a3 = p / P0 * (q * ctwslf + (1 - q) * ctwfrn) * 1e-20;
02781
02782
           ctmpth = a1 * a2 * a3;
02783
02784
           ctmpth = 0;
02785
         return ctmpth;
02786 }
```

5.5.2.8 double ctmn2 (double nu, double p, double t)

Compute nitrogen continuum (absorption coefficient).

Definition at line 2790 of file jurassic.c.

```
02793
02794
02795
           static double ba[98] = { 0., 4.45e-8, 5.22e-8, 6.46e-8, 7.75e-8, 9.03e-8,
              1.06e-7, 1.21e-7, 1.37e-7, 1.57e-7, 1.75e-7, 2.01e-7, 2.3e-7,
02796
              2.59e-7, 2.95e-7, 3.26e-7, 3.66e-7, 4.05e-7, 4.47e-7, 4.92e-7,
              5.34e-7, 5.84e-7, 6.24e-7, 6.67e-7, 7.14e-7, 7.26e-7, 7.54e-7,
02798
02799
              7.84e-7, 8.09e-7, 8.42e-7, 8.62e-7, 8.87e-7, 9.11e-7, 9.36e-7,
02800
              9.76e-7, 1.03e-6, 1.11e-6, 1.23e-6, 1.39e-6, 1.61e-6, 1.76e-6,
             1.94e-6, 1.97e-6, 1.87e-6, 1.75e-6, 1.56e-6, 1.42e-6, 1.35e-6, 1.32e-6, 1.29e-6, 1.29e-6, 1.29e-6, 1.3e-6, 1.32e-6, 1.33e-6,
02801
02802
02803
              1.34e-6, 1.35e-6, 1.33e-6, 1.31e-6, 1.29e-6, 1.24e-6, 1.2e-6,
              1.16e-6, 1.1e-6, 1.04e-6, 9.96e-7, 9.38e-7, 8.63e-7, 7.98e-7,
02804
02805
              7.26e-7, 6.55e-7, 5.94e-7, 5.35e-7, 4.74e-7, 4.24e-7, 3.77e-7
             3.33e-7, 2.96e-7, 2.63e-7, 2.34e-7, 2.08e-7, 1.85e-7, 1.67e-7, 1.47e-7, 1.32e-7, 1.2e-7, 1.09e-7, 9.85e-8, 9.08e-8, 8.18e-8, 7.56e-8, 6.85e-8, 6.14e-8, 5.83e-8, 5.77e-8, 5e-8, 4.32e-8, 0.
02806
02807
02808
02809
02811
           static double betaa[98] = { 802., 802., 761., 722., 679., 646., 609., 562.,
02812
              511., 472., 436., 406., 377., 355., 338., 319., 299., 278., 255.,
              233., 208., 184., 149., 107., 66., 25., -13., -49., -82., -104., -119., -130., -139., -144., -146., -146., -147., -148., -150., -153., -160., -169., -181., -189., -195., -200., -205., -209.,
02813
02814
02815
              -211., -210., -210., -209., -205., -199., -190., -180., -168., -157., -143., -126., -108., -89., -63., -32., 1., 35., 65., 95.,
02816
02817
              121., 141., 152., 161., 164., 164., 161., 155., 148., 143., 137., 133., 131., 133., 139., 150., 165., 187., 213., 248., 284., 321.,
02818
02819
              372., 449., 514., 569., 609., 642., 673., 673.
02820
02821
02822
02823
           static double nua[98] = { 2120., 2125., 2130., 2135., 2140., 2145., 2150.,
02824
              2155., 2160., 2165., 2170., 2175., 2180., 2185., 2190., 2195.,
02825
              2200., 2205., 2210., 2215., 2220., 2225., 2230., 2235., 2240.,
             2245., 2250., 2255., 2260., 2265., 2270., 2275., 2280., 2285., 2300., 2305., 2310., 2315., 2320., 2325., 2330., 2335., 2340., 2345., 2350., 2355., 2360., 2365., 2370., 2375., 2380., 2385., 2390., 2395., 2400., 2405., 2410., 2415., 2420.,
02826
02827
02828
```

```
2425., 2430., 2435., 2440., 2445., 2450., 2455., 2460., 2465.,
           2470., 2475., 2480., 2485., 2490., 2495., 2500., 2505., 2510., 2515., 2520., 2525., 2530., 2535., 2540., 2545., 2550., 2555.,
02831
02832
02833
           2560., 2565., 2570., 2575., 2580., 2585., 2590., 2595., 2600., 2605.
02834
02835
         double b, beta, q_n2 = 0.79, t0 = 273, tr = 296;
02837
02838
         int idx;
02839
02840
         /* Check wavenumber range... */
02841
         if (nu < nua[0] || nu > nua[97])
02842
          return 0;
02843
02844
         /* Interpolate B and beta... */
02845
         idx = locate_reg(nua, 98, nu);
         b = LIN(nua[idx], ba[idx], nua[idx + 1], ba[idx + 1], nu);
02846
02847
         beta = LIN(nua[idx], betaa[idx], nua[idx + 1], betaa[idx + 1], nu);
02848
02849
         /* Compute absorption coefficient... */
         return 0.1 * POW2(p / P0 * t0 / t) * exp(beta * (1 / tr - 1 / t))  
* q_n2 * b * (q_n2 + (1 - q_n2) * (1.294 - 0.4545 * t / tr));
02850
02851
02852 }
```

Here is the call graph for this function:



5.5.2.9 double ctmo2 (double nu, double p, double t)

Compute oxygen continuum (absorption coefficient).

Definition at line 2856 of file jurassic.c.

```
02859
02860
                static double ba[90] = { 0., .061, .074, .084, .096, .12, .162, .208, .246, .285, .314, .38, .444, .5, .571, .673, .768, .853, .966, 1.097,
02861
02862
                     1.214, 1.333, 1.466, 1.591, 1.693, 1.796, 1.922, 2.037, 2.154,
02864
                     2.264, 2.375, 2.508, 2.671, 2.847, 3.066, 3.417, 3.828, 4.204,
02865
                    4.453, 4.599, 4.528, 4.284, 3.955, 3.678, 3.477, 3.346, 3.29,
                   3.251, 3.231, 3.226, 3.212, 3.192, 3.108, 3.033, 2.911, 2.798, 2.646, 2.508, 2.322, 2.13, 1.928, 1.757, 1.588, 1.417, 1.253, 1.109, .99, .888, .791, .678, .587, .524, .464, .403, .357, .32, .29, .267, .242, .215, .182, .16, .146, .128, .103, .087, .081,
02866
02867
02868
02870
                     .071, .064, 0.
02871
02872
                static double betaa[90] = { 467., 467., 400., 315., 379., 368., 475., 521., 531., 512., 442., 444., 430., 381., 335., 324., 296., 248., 215., 193., 158., 127., 101., 71., 31., -6., -26., -47., -63., -79., -88., -88., -87., -90., -98., -99., -109., -134., -160., -167., -164., -158., -153., -151., -156., -166., -168., -173., -170., -161., -145., -126., -108., -84., -59., -29., 4., 41., 73., 97., 123., 159., 188., 220., 242., 256., 281., 311., 334., 319., 313.
02873
02874
02875
02876
02877
02878
                    123., 159., 198., 220., 242., 256., 281., 311., 334., 319., 313., 321., 323., 310., 315., 320., 335., 361., 378., 373., 338., 319., 346., 322., 291., 290., 350., 371., 504., 504.
02879
02880
02881
02882
02883
02884
                static double nua[90] = { 1360., 1365., 1370., 1375., 1380., 1385., 1390.,
                    1395., 1400., 1405., 1410., 1415., 1420., 1425., 1430., 1435., 1440., 1445., 1450., 1455., 1460., 1465., 1470., 1475., 1480., 1485., 1490., 1495., 1500., 1505., 1510., 1515., 1520., 1525.,
02885
02886
02887
                    1530., 1535., 1540., 1545., 1550., 1555., 1560., 1565., 1570.,
```

```
1575., 1580., 1585., 1590., 1595., 1600., 1605., 1610., 1615.,
           1620., 1625., 1630., 1635., 1640., 1645., 1650., 1655., 1660., 1665., 1670., 1675., 1680., 1685., 1690., 1695., 1700., 1705.,
02890
02891
02892
           1710., 1715., 1720., 1725., 1730., 1735., 1740., 1745., 1750.,
02893
           1755., 1760., 1765., 1770., 1775., 1780., 1785., 1790., 1795.,
02894
           1800., 1805.
02895
02896
02897
         double b, beta, q_02 = 0.21, t0 = 273, tr = 296;
02898
02899
         int idx:
02900
02901
         /* Check wavenumber range...
02902
         if (nu < nua[0] || nu > nua[89])
02903
           return 0;
02904
02905
        /* Interpolate B and beta... */
        idx = locate_reg(nua, 90, nu);
b = LIN(nua[idx], ba[idx], nua[idx + 1], ba[idx + 1], nu);
02906
02907
02908
         beta = LIN(nua[idx], betaa[idx], nua[idx + 1], betaa[idx + 1], nu);
02909
02910
         /* Compute absorption coefficient... */
         return 0.1 * POW2(p / P0 * t0 / t) * exp(beta * (1 / tr - 1 / t)) * q_o2 *
02911
02912
           h:
02913 }
```

Here is the call graph for this function:



5.5.2.10 void copy_atm (ctl_t * ctl, atm_t * atm_dest, atm_t * atm_src, int init)

Copy and initialize atmospheric data.

Definition at line 2917 of file jurassic.c.

```
02921
02922
02923
        int ig, ip, iw;
02924
02925
        size_t s;
02926
        /* Data size... */
02927
02928
        s = (size_t) atm_src->np * sizeof(double);
02929
        /* Copy data... */
atm_dest->np = atm_src->np;
02930
02931
        memcpy(atm_dest->time, atm_src->time, s);
02933
        memcpy(atm_dest->z, atm_src->z, s);
02934
        memcpy(atm_dest->lon, atm_src->lon, s);
02935
        memcpy(atm_dest->lat, atm_src->lat, s);
02936
        memcpy(atm_dest->p, atm_src->p, s);
02937
        memcpy(atm_dest->t, atm_src->t, s);
        for (ig = 0; ig < ctl->ng; ig++)
02938
02939
          memcpy(atm_dest->q[ig], atm_src->q[ig], s);
02940
        for (iw = 0; iw < ctl->nw; iw++)
02941
          memcpy(atm_dest->k[iw], atm_src->k[iw], s);
02942
02943
        /* Initialize... */
02944
        if (init)
02945
          for (ip = 0; ip < atm_dest->np; ip++) {
02946
             atm_dest \rightarrow p[ip] = 0;
             atm_dest->t[ip] = 0;
02947
02948
             for (ig = 0; ig < ctl->ng; ig++)
             atm\_dest->q[ig][ip] = 0;
for (iw = 0; iw < ctl->nw; iw++)
02949
02950
02951
               atm_dest->k[iw][ip] = 0;
02952
          }
02953 }
```

```
5.5.2.11 void copy_obs ( ctl_t * ctl, obs_t * obs_dest, obs_t * obs_src, int init )
```

Copy and initialize observation data.

Definition at line 2957 of file jurassic.c.

```
02961
02962
02963
         int id, ir;
02964
02965
         size t s:
02966
02967
         /* Data size... */
02968
         s = (size_t) obs_src->nr * sizeof(double);
02969
         /* Copy data... */
02970
02971
         obs dest->nr = obs_src->nr;
        memcpy(obs_dest->time, obs_src->time, s);
memcpy(obs_dest->obsz, obs_src->obsz, s);
02972
02973
02974
         memcpy(obs_dest->obslon, obs_src->obslon, s);
02975
         memcpy(obs_dest->obslat, obs_src->obslat, s);
02976
         memcpy(obs_dest->vpz, obs_src->vpz, s);
        memcpy(obs_dest->vplon, obs_src->vplon, s);
memcpy(obs_dest->vplat, obs_src->vplat, s);
02977
02978
02979
         memcpy(obs_dest->tpz, obs_src->tpz, s);
02980
         memcpy(obs_dest->tplon, obs_src->tplon, s);
02981
         memcpy(obs_dest->tplat, obs_src->tplat, s);
02982
         for (id = 0; id < ctl->nd; id++)
        memcpy(obs_dest->rad[id], obs_src->rad[id], s);
for (id = 0; id < ctl->nd; id++)
02983
02984
02985
           memcpy(obs_dest->tau[id], obs_src->tau[id], s);
02987
         /* Initialize... */
02988
         if (init)
         for (id = 0; id < ctl->nd; id++)
  for (ir = 0; ir < obs_dest->nr; ir++)
  if (gsl_finite(obs_dest->rad[id][ir])) {
02989
02990
02991
02992
                 obs_dest->rad[id][ir] = 0;
02993
                  obs_dest->tau[id][ir] = 0;
02994
02995 }
```

5.5.2.12 int find_emitter (ctl_t * ctl, const char * emitter)

Find index of an emitter.

Definition at line 2999 of file jurassic.c.

```
03001
03002
03003    int ig;
03004
03005    for (ig = 0; ig < ctl->ng; ig++)
03006         if (strcasecmp(ctl->emitter[ig], emitter) == 0)
03007         return ig;
03008
03009    return -1;
03010 }
```

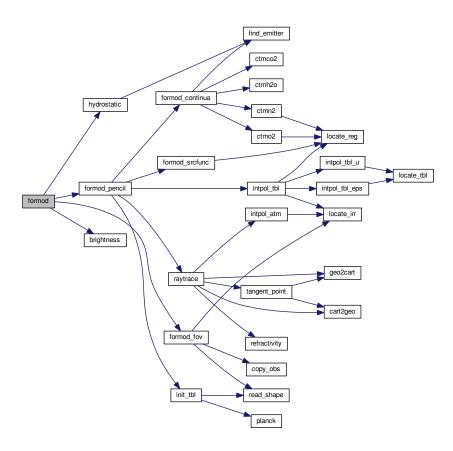
5.5.2.13 void formod ($ctl_t * ctl$, $atm_t * atm$, $obs_t * obs$)

Determine ray paths and compute radiative transfer.

Definition at line 3014 of file jurassic.c.

```
03018
03019
           int id, ir, *mask;
03020
          /* Allocate... */
ALLOC(mask, int,
03021
03022
03023
                  ND * NR);
03024
          /* Save observation mask... */
for (id = 0; id < ctl->nd; id++)
  for (ir = 0; ir < obs->nr; ir++)
    mask[id * NR + ir] = !gsl_finite(obs->rad[id][ir]);
03025
03026
03027
03028
03029
03030
           /* Hydrostatic equilibrium... */
03031
           hydrostatic(ctl, atm);
03032
          /* Calculate pencil beams... */
for (ir = 0; ir < obs->nr; ir++)
  formod_pencil(ctl, atm, obs, ir);
03033
03034
03035
03036
03037
           /* Apply field-of-view convolution... */
03038
           formod_fov(ctl, obs);
03039
03040
           /\star Convert radiance to brightness temperature... \star/
           if (ctl->write_bbt)
03041
03042
            for (id = 0; id < ctl->nd; id++)
03043
                for (ir = 0; ir < obs->nr; ir++)
03044
                   obs->rad[id][ir] = brightness(obs->rad[id][ir], ctl->nu[id]);
03045
          /* Apply observation mask... */
for (id = 0; id < ctl->nd; id++)
  for (ir = 0; ir < obs->nr; ir++)
    if (mask[id * NR + ir])
03046
03047
03048
03049
03050
                   obs->rad[id][ir] = GSL_NAN;
03051
           /* Free... */
03052
03053
          free(mask);
03054 }
```

Here is the call graph for this function:



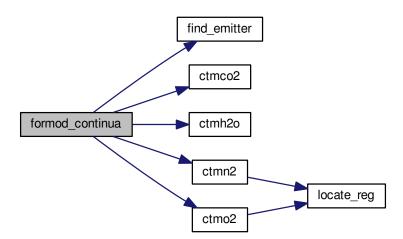
5.5.2.14 void formod_continua ($ctl_t * ctl$, $los_t * los$, int ip, double * beta)

Compute absorption coefficient of continua.

Definition at line 3058 of file jurassic.c.

```
03062
03063
03064
         static int ig_{co2} = -999, ig_{h20} = -999;
03065
03066
        int id;
03067
03068
        /* Extinction... */
for (id = 0; id < ctl->nd; id++)
  beta[id] = los->k[ctl->window[id]][ip];
03069
03070
03071
03072
         /* CO2 continuum... */
03073
        if (ctl->ctm_co2) {
03074
         if (ig_co2 == -999)
             ig_co2 = find_emitter(ct1, "CO2");
03075
03076
           if (ig_co2 >= 0)
03077
             for (id = 0; id < ctl->nd; id++)
03078
               beta[id] += ctmco2(ctl->nu[id], los->p[ip], los->t[ip],
03079
                                      los->u[ig_co2][ip]) / los->ds[ip];
03080
        }
03081
         /* H2O continuum... */
03082
03083
        if (ct1->ctm_h2o) {
         if (ig_h2o == -999)
03084
03085
             ig_h2o = find_emitter(ctl, "H2O");
          if (ig_h2o >= 0)
    for (id = 0; id < ctl->nd; id++)
        beta[id] += ctmh2o(ctl->nu[id], los->p[ip], los->t[ip],
03086
03087
03088
03089
                                      los->q[ig_h2o][ip],
03090
                                      los->u[ig_h2o][ip]) / los->ds[ip];
03091
03092
03093
         /* N2 continuum... */
03094
         if (ctl->ctm_n2)
         for (id = 0; id < ctl->nd; id++)
03095
03096
             beta[id] += ctmn2(ctl->nu[id], los->p[ip], los->t[ip]);
03097
         /* 02 continuum... */
03098
        if (ctl->ctm_o2)
  for (id = 0; id < ctl->nd; id++)
  beta[id] += ctmo2(ctl->nu[id], los->p[ip], los->t[ip]);
03099
03100
03101
03102 }
```

Here is the call graph for this function:



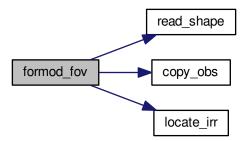
```
5.5.2.15 void formod_fov ( ctl_t * ctl, obs_t * obs )
```

Apply field of view convolution.

Definition at line 3106 of file jurassic.c.

```
03108
03109
03110
        static double dz[NSHAPE], w[NSHAPE];
03111
03112
       static int init = 0, n;
03113
0.3114
        obs t *obs2:
03115
03116
       double rad[ND][NR], tau[ND][NR], wsum, z[NR], zfov;
03117
03118
        int i, id, idx, ir, ir2, nz;
03119
03120
        /* Do not take into account FOV... */
        if (ctl->fov[0] == '-')
03121
03122
          return;
03123
03124
        /* Initialize FOV data... */
03125
        if (!init) {
03126
         init = 1:
03127
          read_shape(ctl->fov, dz, w, &n);
03128
03129
03130
        /* Allocate... */
03131
        ALLOC(obs2, obs_t, 1);
03132
03133
        /* Copy observation data... */
03134
        copy_obs(ctl, obs2, obs, 0);
03135
        /* Loop over ray paths... */
for (ir = 0; ir < obs->nr; ir++) {
03136
03137
03138
03139
          /* Get radiance and transmittance profiles... */
03140
          nz = 0;
          for (ir2 = GSL_MAX(ir - NFOV, 0); ir2 < GSL_MIN(ir + 1 + NFOV, obs->nr);
03141
             ir2++)
if (obs->time[ir2] == obs->time[ir]) {
03142
03143
0.3144
              z[nz] = obs2->vpz[ir2];
               for (id = 0; id < ctl->nd; id++) {
03145
                rad[id][nz] = obs2->rad[id][ir2];
tau[id][nz] = obs2->tau[id][ir2];
03146
03147
03148
              nz++;
03149
03150
          if (nz < 2)
03151
            ERRMSG("Cannot apply FOV convolution!");
03152
03153
03154
          /\star Convolute profiles with FOV... \star/
03155
          for (id = 0; id < ctl->nd; id++) {
03156
03157
            obs \rightarrow rad[id][ir] = 0;
            obs->tau[id][ir] = 0;
03158
03159
03160
          for (i = 0; i < n; i++) {
03161
           zfov = obs->vpz[ir] + dz[i];
             idx = locate_irr(z, nz, zfov);
03162
             for (id = 0; id < ctl->nd; id++) {
03163
              obs->rad[id][ir] += w[i]
03164
              * LIN(z[idx], rad[id][idx], z[idx + 1], rad[id][idx + 1], zfov);
obs->tau[id][ir] += w[i]
03165
03166
03167
                 * LIN(z[idx], tau[id][idx], z[idx + 1], tau[id][idx + 1], zfov);
03168
03169
            wsum += w[i];
03170
          for (id = 0; id < ctl->nd; id++) {
03171
            obs->rad[id][ir] /= wsum;
03172
03173
             obs->tau[id][ir] /= wsum;
03174
03175
        }
03176
03177
        /* Free... */
03178
        free (obs2);
03179 }
```

Here is the call graph for this function:



```
5.5.2.16 void formod_pencil ( ctl_t * ctl, atm_t * atm, obs_t * obs, int ir )
```

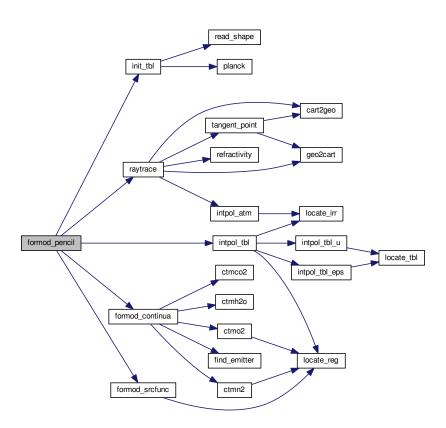
Compute radiative transfer for a pencil beam.

Definition at line 3183 of file jurassic.c.

```
0.3187
03188
        static tbl_t *tbl;
03190
03191
        static int init = 0;
03192
03193
        los t *los:
03194
03195
        double beta_ctm[ND], eps, src_planck[ND], tau_path[NG][ND], tau_gas[ND];
03196
03197
03198
03199
        /* Initialize look-up tables... */
03200
        if (!init) {
03201
         init = 1;
03202
          ALLOC(tbl, tbl_t, 1);
03203
          init_tbl(ctl, tbl);
03204
03205
        /* Allocate... */
03206
03207
        ALLOC(los, los_t, 1);
03208
        /* Initialize... */
for (id = 0; id < ctl->nd; id++) {
03209
03210
03211
         obs->rad[id][ir] = 0;
03212
         obs->tau[id][ir] = 1;
03213
03214
03215
        /* Raytracing... */
03216
        raytrace(ctl, atm, obs, los, ir);
03217
03218
        /* Loop over LOS points... */
03219
        for (ip = 0; ip < los->np; ip++) {
03220
03221
           /* Get trace gas transmittance... */
03222
          intpol_tbl(ctl, tbl, los, ip, tau_path, tau_gas);
03223
03224
          /* Get continuum absorption... */
03225
          formod_continua(ctl, los, ip, beta_ctm);
03226
03227
          /* Compute Planck function... */
03228
          formod_srcfunc(ctl, tbl, los->t[ip], src_planck);
03229
03230
          /* Loop over channels... */
for (id = 0; id < ctl->nd; id++)
03231
03232
            if (tau_gas[id] > 0) {
03233
```

```
/* Get segment emissivity... */
03235
              eps = 1 - tau_gas[id] * exp(-beta_ctm[id] * los->ds[ip]);
03236
03237
              /\star Compute radiance... \star/
03238
              obs->rad[id][ir] += src_planck[id] * eps * obs->tau[id][ir];
03239
03240
              /* Compute path transmittance... */
03241
              obs->tau[id][ir] *= (1 - eps);
03242
03243
03244
       /* Add surface... */
if (los->tsurf > 0) {
03245
03246
        formod_srcfunc(ctl, tbl, los->tsurf, src_planck);
03247
03248
         for (id = 0; id < ctl->nd; id++)
03249
            obs->rad[id][ir] += src_planck[id] * obs->tau[id][ir];
03250
03251
03252
       /* Free... */
03253
       free(los);
03254 }
```

Here is the call graph for this function:



5.5.2.17 void formod_srcfunc ($ctl_t * ctl$, $tbl_t * tbl$, double t, double * src)

Compute Planck source function.

Definition at line 3258 of file jurassic.c.

```
03262 {
03263
03264 int id, it;
```

Here is the call graph for this function:



5.5.2.18 void geo2cart (double z, double lon, double lat, double *x)

Convert geolocation to Cartesian coordinates.

Definition at line 3277 of file jurassic.c.

```
03281 {
03282
03283 double radius;
03284
03285 radius = z + RE;
03286 x[0] = radius * cos(lat / 180 * M_PI) * cos(lon / 180 * M_PI);
03287 x[1] = radius * cos(lat / 180 * M_PI) * sin(lon / 180 * M_PI);
03288 x[2] = radius * sin(lat / 180 * M_PI);
03289 }
```

5.5.2.19 void hydrostatic (ctl_t * ctl, atm_t * atm)

Set hydrostatic equilibrium.

Definition at line 3293 of file jurassic.c.

```
03295
03296
         static int ig_h2o = -999;
03298
03299
         double dzmin = 1e99, e = 0, mean, mmair = 28.96456e-3, mmh2o = 18.0153e-3;
03300
         int i, ip, ipref = 0, ipts = 20;
03301
03302
03303
         /* Check reference height... */
03304
         if (ctl->hydz < 0)
03305
03306
03307
         /* Determine emitter index of H2O... */
         if (ig_h2o == -999)
03308
           ig_h2o = find_emitter(ctl, "H2O");
03309
03310
03311
          /* Find air parcel next to reference height... */
         for (ip = 0; ip < atm->np; ip++)
  if (fabs(atm->z[ip] - ctl->hydz) < dzmin) {
    dzmin = fabs(atm->z[ip] - ctl->hydz);
    ipref = ip;
03312
03313
03314
03315
03316
```

```
03317
        /* Upper part of profile... */
for (ip = ipref + 1; ip < atm->np; ip++) {
03318
03319
          mean = 0;
03320
          for (i = 0; i < ipts; i++) {
  if (ig_h2o >= 0)
03321
03322
              e = LIN(0.0, atm->q[ig_h2o][ip - 1],
03323
03324
                       ipts - 1.0, atm->q[ig_h2o][ip], (double) i);
03325
             mean += (e * mmh2o + (1 - e) * mmair)
              * GO / RI / LIN(0.0, atm->t[ip - 1], ipts - 1.0, atm->t[ip], (double) i) / ipts;
03326
03327
03328
03329
03330
           /* Compute p(z,T)... */
03331
          atm->p[ip] =
03332
            \exp(\log(atm-p[ip-1]) - mean * 1000 * (atm-z[ip] - atm-z[ip - 1]));
03333
03334
03335
        /* Lower part of profile... */
03336
        for (ip = ipref - 1; ip >= 0; ip--) {
03337
          mean = 0;
          for (i = 0; i < ipts; i++) {</pre>
03338
            if (ig_h2o >= 0)
03339
              e = LIN(0.0, atm->q[ig_h2o][ip + 1],
03340
03341
                       ipts - 1.0, atm->q[ig_h2o][ip], (double) i);
03342
            mean += (e * mmh2o + (1 - e) * mmair)
              * G0 / RI
03343
               / LIN(0.0, atm->t[ip + 1], ipts - 1.0, atm->t[ip], (double) i) / ipts;
03344
03345
          }
03346
03347
          /* Compute p(z,T)... */
03348
          atm->p[ip]
03349
            exp(log(atm->p[ip + 1]) - mean * 1000 * (atm->z[ip] - atm->z[ip + 1]));
03350
03351 }
```

Here is the call graph for this function:



5.5.2.20 void idx2name ($ctl_t * ctl$, int idx, char * quantity)

Determine name of state vector quantity for given index.

Definition at line 3355 of file jurassic.c.

```
03358
                         {
03359
03360
        int ig, iw;
03361
03362
        if (idx == IDXP)
          sprintf(quantity, "PRESSURE");
03363
03364
        if (idx == IDXT)
03365
03366
          sprintf(quantity, "TEMPERATURE");
03367
03368
        for (ig = 0; ig < ctl->ng; ig++)
          if (idx == IDXQ(ig))
sprintf(quantity, "%s", ctl->emitter[ig]);
03369
03370
03371
03372
        for (iw = 0; iw < ctl->nw; iw++)
03373
          if (idx == IDXK(iw))
            sprintf(quantity, "EXTINCT_WINDOW%d", iw);
03374
03375 }
```

```
5.5.2.21 void init_tbl ( ctl_t * ctl, tbl_t * tbl )
```

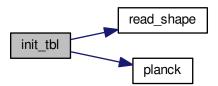
Initialize look-up tables.

Definition at line 3379 of file jurassic.c.

```
03381
                       {
03382
03383
        FILE *in;
03384
        char filename[2 * LEN], line[LEN];
03385
03386
03387
        double eps, eps_old, press, press_old, temp, temp_old, u, u_old,
   f[NSHAPE], fsum, nu[NSHAPE];
03389
03390
        int i, id, ig, ip, it, n;
03391
03392
        /* Loop over trace gases and channels... */
03393
        for (ig = 0; ig < ctl->ng; ig++)
03394 #pragma omp parallel for default (none) shared(ctl,tbl,ig) private(in,filename,line,eps,eps_old,press,
      press_old,temp,temp_old,u,u_old,id,ip,it)
03395
          for (id = 0; id < ctl->nd; id++) {
03396
03397
             /* Initialize... */
            tbl->np[ig][id] = -1;
eps_old = -999;
03398
03399
03400
            press_old = -999;
             temp\_old = -999;
03401
            u_old = -999;
03402
03403
03404
             /* Try to open file... */
            sprintf(filename, "%s_%.4f_%s.tab",
03405
03406
                     ctl->tblbase, ctl->nu[id], ctl->emitter[ig]);
03407
             if (!(in = fopen(filename, "r"))) {
03408
              printf("Missing emissivity table: %s\n", filename);
              continue;
03409
03410
03411
            printf("Read emissivity table: %s\n", filename);
03412
03413
             /* Read data...
03414
             while (fgets(line, LEN, in)) {
03415
              /* Parse line... */ if (sscanf(line, "%lg %lg %lg %lg", &press, &temp, &u, &eps) != 4)
03416
03417
03418
                 continue;
03419
03420
               /* Determine pressure index... */
               if (press != press_old) {
  press_old = press;
03421
03422
                 if ((++tbl->np[ig][id]) >= TBLNP)
03423
                   ERRMSG("Too many pressure levels!");
03424
                tbl->nt[ig][id][tbl->np[ig][id]] = -1;
03425
03426
03427
03428
              /* Determine temperature index... */
if (temp != temp_old) {
03429
03430
                 temp_old = temp;
03431
                 if ((++tbl->nt[ig][id][tbl->np[ig][id]]) >= TBLNT)
                 ERRMSG("Too many temperatures!");
tbl->nu[ig][id][tbl->np[ig][id]]
03432
03433
03434
                   [tbl->nt[ig][id][tbl->np[ig][id]]] = -1;
03435
03436
03437
               /* Determine column density index... */
03438
               03439
                   [tbl->nt[ig][id][tbl->np[ig][id]]] \ < \ 0) \ \ \{
03440
                 eps_old = eps;
03441
                 u_old = u;
03442
                 if ((++tbl->nu[ig][id][tbl->np[ig][id]]
                      [tbl->nt[ig][id][tbl->np[ig][id]]]) >= TBLNU) {
03443
03444
                   tbl->nu[ig][id][tbl->np[ig][id]]
03445
                     [tbl->nt[ig][id][tbl->np[ig][id]]]--;
03446
                   continue;
03447
                }
03448
03449
03450
               /* Store data... */
03451
               tbl->p[ig][id][tbl->np[ig][id]] = press;
03452
               \label{tbl-hp[ig][id][tbl-hp[ig][id]][tbl-ht[ig][id][tbl-hp[ig][id]]]} tbl-ht[ig][id][id][id]-ht[ig][id][id][id]
03453
                 = temp;
03454
               tbl->u[ig][id][tbl->np[ig][id]][tbl->nt[ig][id][tbl->np[ig][id]]]
03455
                [tbl->nu[ig][id][tbl->np[ig][id]]
03456
                  [tbl->nt[ig][id][tbl->np[ig][id]]] = (float) u;
```

```
tbl->eps[ig][id][tbl->np[ig][id]][tbl->nt[ig][id][tbl->np[ig][id]]]
03458
                  [tbl->nu[ig][id][tbl->np[ig][id]]
03459
                    [tbl->nt[ig][id][tbl->np[ig][id]]]] = (float) eps;
03460
03461
              /* Increment counters... */
03462
             tbl->np[ig][id]++;
03463
03464
              for (ip = 0; ip < tbl->np[ig][id]; ip++) {
              tbl->nt[ig][id][ip]++;
for (it = 0; it < tbl->nt[ig][id][ip]; it++)
   tbl->nu[ig][id][ip][it]++;
03465
03466
03467
03468
03469
03470
              /* Close file... */
03471
             fclose(in);
03472
03473
        /* Write info... */ printf("Initialize source function table...\n");
03474
03475
03476
03477
         /* Loop over channels... */
03478 #pragma omp parallel for default(none) shared(ctl,tbl,ig) private(filename,it,i,n,f,fsum,nu) 03479 for (id = 0; id < ctl->nd; id++) {
03480
03481
           /* Read filter function... */
03482
           sprintf(filename, "%s_%.4f.filt", ctl->tblbase, ctl->nu[id]);
03483
           read_shape(filename, nu, f, &n);
03484
           /* Compute source function table... */
for (it = 0; it < TBLNS; it++) {</pre>
03485
03486
03487
03488
              /* Set temperature... */
03489
             tbl->st[it] = LIN(0.0, TMIN, TBLNS - 1.0, TMAX, (double) it);
03490
03491
              /* Integrate Planck function... */
             fsum = 0;
03492
             tbl->sr[id][it] = 0;
03493
03494
              for (i = 0; i < n; i++) {</pre>
03495
                fsum += f[i];
03496
                tbl->sr[id][it] += f[i] * planck(tbl->st[it], nu[i]);
03497
03498
             tbl->sr[id][it] /= fsum;
03499
03500
        }
03501 }
```

Here is the call graph for this function:



5.5.2.22 void intpol_atm (ctl_t * ctl, atm_t * atm, double z, double * p, double * t, double * q, double * k)

Interpolate atmospheric data.

Definition at line 3505 of file jurassic.c.

```
/* Get array index... */
03517
         ip = locate_irr(atm->z, atm->np, z);
03518
03519
         /* Interpolate... */
         *p = EXP(atm->z[ip], atm->p[ip], atm->z[ip + 1], atm->p[ip + 1], z);
*t = LIN(atm->z[ip], atm->t[ip], atm->z[ip + 1], atm->t[ip + 1], z);
03520
03521
         for (ig = 0; ig < ctl->ng; ig++)
03523
           q[ig] =
03524
              \label{eq:linear} LIN(atm->z[ip], atm->q[ig][ip], atm->z[ip+1], atm->q[ig][ip+1], z);
          for (iw = 0; iw < ctl->nw; iw++)
  k[iw] =
03525
03526
03527
               \label{eq:linear} LIN(atm->z[ip], atm->k[iw][ip], atm->z[ip+1], atm->k[iw][ip+1], z);
03528 }
```

Here is the call graph for this function:



5.5.2.23 void intpol tbl (ctl t * ctl, tbl t * tbl, los t * los, int ip, double tau path[NG][ND], double tau seq[ND])

Get transmittance from look-up tables.

Definition at line 3532 of file jurassic.c.

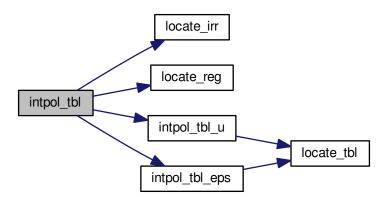
```
03538
03540
       double eps, eps00, eps01, eps10, eps11, u;
03541
03542
       int id, ig, ipr, it0, it1;
03543
03544
        /* Initialize... */
       if (ip <= 0)</pre>
03545
        for (ig = 0; ig < ctl->ng; ig++)
03546
03547
           for (id = 0; id < ctl->nd; id++)
03548
              tau_path[ig][id] = 1;
03549
03550
       /* Loop over channels... */
03551
       for (id = 0; id < ctl->nd; id++) {
03552
03553
          /* Initialize... */
03554
         tau_seg[id] = 1;
03555
03556
         /* Loop over emitters.... */
03557
         for (ig = 0; ig < ctl->ng; ig++) {
03559
            /\star Check size of table (pressure)... \star/
03560
            if (tbl->np[ig][id] < 2)
03561
             eps = 0;
03562
            /* Check transmittance... */
03563
           else if (tau_path[ig][id] < 1e-9)</pre>
03564
03565
             eps = 1;
03566
03567
           /* Interpolate... */
03568
           else {
03569
              /* Determine pressure and temperature indices... */
03571
              ipr = locate_irr(tbl->p[ig][id], tbl->np[ig][id], los->p[ip]);
03572
03573
                locate_irr(tbl->t[ig][id][ipr], tbl->nt[ig][id][ipr], los->
     t[ip]);
03574
              it1 =
03575
               locate_reg(tbl->t[ig][id][ipr + 1], tbl->nt[ig][id][ipr + 1],
03576
                           los->t[ip]);
```

```
03578
                 /\star Check size of table (temperature and column density)... \star/
                03579
03580
                     || tbl->nu[ig][id][ipr][it0 + 1] < 2
03581
                     || tbl=>nu[ig][id][ipr + 1][it1] < 2
|| tbl=>nu[ig][id][ipr + 1][it1 + 1] < 2
03582
03584
                   eps = 0;
03585
03586
                else {
03587
                  /* Get emissivities of extended path... */
u = intpol_tbl_u(tbl, ig, id, ipr, it0, 1 - tau_path[ig][id]);
eps00 = intpol_tbl_eps(tbl, ig, id, ipr, it0, u + los->u[ig][ip]);
03588
03589
03590
03591
03592
                   u = intpol_tbl_u(tbl, ig, id, ipr, it0 + 1, 1 - tau_path[ig][id]);
03593
                   eps01 =
03594
                     intpol_tbl_eps(tbl, ig, id, ipr, it0 + 1, u + los->u[ig][ip]);
03595
03596
                   u = intpol_tbl_u(tbl, ig, id, ipr + 1, it1, 1 - tau_path[ig][id]);
                   eps10 =
03597
03598
                     intpol_tbl_eps(tbl, ig, id, ipr + 1, it1, u + los->u[ig][ip]);
03599
03600
03601
                     intpol_tbl_u(tbl, iq, id, ipr + 1, it1 + 1, 1 - tau_path[iq][id]);
03602
                   eps11 =
03603
                     intpol_tbl_eps(tbl, ig, id, ipr + 1, it1 + 1, u + los->
      u[ig][ip]);
03604
03605
                   /* Interpolate with respect to temperature... */
03606
                   eps00 = LIN(tbl->t[ig][id][ipr][it0], eps00,
                   tbl->t[ig][id][ipr][it0 + 1], eps01, los->t[ip]);

eps11 = LIN(tbl->t[ig][id][ipr + 1][it1], eps10,

tbl->t[ig][id][ipr + 1][it1 + 1], eps11, los->t[ip]);
03607
03608
03609
03610
                  /* Interpolate with respect to pressure... */
03611
                  eps00 = LIN(tbl->p[ig][id][ipr], eps00,
tbl->p[ig][id][ipr + 1], eps11, los->p[ip]);
03612
03613
03614
03615
                   /* Check emssivity range... */
03616
                   eps00 = GSL_MAX(GSL_MIN(eps00, 1), 0);
03617
                   /* Determine segment emissivity... */
eps = 1 - (1 - eps00) / tau_path[ig][id];
03618
03619
03620
03621
03622
03623
              /\!\star Get transmittance of extended path... \star/
              tau_path[ig][id] *= (1 - eps);
03624
03625
03626
              /* Get segment transmittance... */
03627
              tau_seg[id] *= (1 - eps);
03628
03629
        }
03630 }
```

Here is the call graph for this function:



5.5.2.24 double intpol_tbl_eps ($tbl_t * tbl_t$, int ig_t , int

Interpolate emissivity from look-up tables.

Definition at line 3634 of file jurassic.c.

```
03640
                       {
03641
03642
         int idx;
03644
          /* Lower boundary... */
03645
         if (u < tbl->u[ig][id][ip][it][0])
          return LIN(0, 0, tbl->u[ig][id][ip][it][0], tbl->eps[ig][id][ip][it][0],
03646
03647
                          u);
03648
03649
         /* Upper boundary... */
         else if (u > tbl->u[ig][id][ip][it][tbl->nu[ig][id][ip][it] - 1])
   return LIN(tbl->u[ig][id][ip][it][tbl->nu[ig][id][ip][it] - 1],
03650
03651
03652
                          \label{locality} \verb|tbl->eps[ig][id][ip][it][tbl->nu[ig][id][ip][it] - 1|,
03653
                          1e30, 1, u);
03654
03655
         /* Interpolation... */
03656
         else {
03657
03658
            /\star Get index... \star/
03659
            idx = locate_tbl(tbl->u[ig][id][ip][it], tbl->nu[ig][id][ip][it], u);
03660
03661
03662
              LIN(tbl->u[ig][id][ip][it][idx], tbl->eps[ig][id][ip][it][idx], tbl->u[ig][id][ip][it][idx + 1], tbl->eps[ig][id][ip][it][idx + 1],
03663
03664
03665
03666
03667 }
```

Here is the call graph for this function:



5.5.2.25 double intpol_tbl_u ($tbl_t * tbl$, int ig, int id, int ip, int it, double eps)

Interpolate column density from look-up tables.

Definition at line 3671 of file jurassic.c.

```
03677
                 {
03678
      int idx;
03680
03681
       /* Lower boundary... */
      if (eps < tbl->eps[ig][id][ip][it][0])
03682
       return LIN(0, 0, tbl->eps[ig][id][ip][it][0], tbl->u[ig][id][ip][it][0],
03683
03684
                 eps);
03685
03686
      /* Upper boundary... */
03687
      else if (eps > tbl->eps[ig][id][ip][it][tbl->nu[ig][id][ip][it] - 1])
        03688
03689
03690
                 1, 1e30, eps);
03691
```

```
/* Interpolation... */
03693
      else {
03694
03695
        /* \ \mathsf{Get} \ \mathsf{index} \ldots \ */
        idx = locate_tbl(tbl->eps[ig][id][ip][it], tbl->nu[ig][id][ip][it], eps);
03696
03697
03698
        /* Interpolate... */
03699
          03700
03701
03702
             eps);
03703
03704 }
```

Here is the call graph for this function:



```
5.5.2.26 void jsec2time ( double jsec, int * year, int * mon, int * day, int * hour, int * min, int * sec, double * remain )
```

Convert seconds to date.

Definition at line 3708 of file jurassic.c.

```
03716
                        {
03718
       struct tm t0, *t1;
03719
03720
       time_t jsec0;
03721
03722
       t0.tm_year = 100;
03723
        t0.tm_mon = 0;
03724
        t0.tm_mday = 1;
       t0.tm\_hour = 0;
03725
       t0.tm_min = 0;
03726
       t0.tm_sec = 0;
03727
03728
03729
       jsec0 = (time_t) jsec + timegm(&t0);
03730 t1 = gmtime(&jsec0);
03731
03732
       *year = t1->tm_year + 1900;
03733
       *mon = t1->tm_mon + 1;
       *day = t1->tm_mday;
03734
03735
       *hour = t1->tm_hour;
03736
       *min = t1->tm_min;
03737
        *sec = t1->tm_sec;
03738
        *remain = jsec - floor(jsec);
03739 }
```

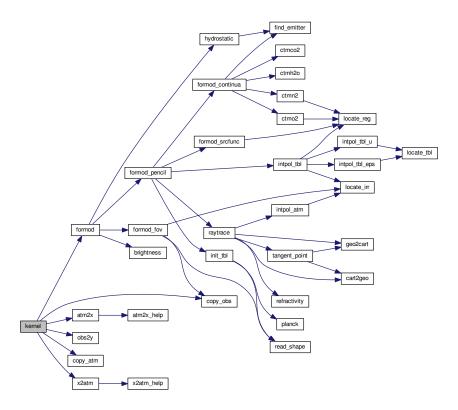
5.5.2.27 void kernel (ctl_t * ctl, atm_t * atm, obs_t * obs, gsl_matrix * k)

Compute Jacobians.

Definition at line 3743 of file jurassic.c.

```
03747
                         {
03748
03749
        atm_t *atm1;
03750
        obs_t *obs1;
03751
03752
        asl vector *x0, *x1, *vv0, *vv1;
03753
03754
        int *iqa, j;
03755
03756
        double h;
03757
03758
        size t i, n, m;
03759
03760
        /* Get sizes... */
03761
        m = k->size1;
        n = k -> size2;
03762
03763
03764
        /* Allocate... */
03765
        x0 = gsl\_vector\_alloc(n);
03766
        yy0 = gsl_vector_alloc(m);
03767
        ALLOC(iqa, int,
03768
              N);
03769
03770
        /\star Compute radiance for undisturbed atmospheric data... \star/
03771
        formod(ctl, atm, obs);
03772
03773
        /* Compose vectors... */
03774
        atm2x(ctl, atm, x0, iqa, NULL);
03775
        obs2y(ctl, obs, yy0, NULL, NULL);
03776
03777
        /* Initialize kernel matrix... */
03778
        gsl matrix set zero(k);
03779
03780
        /\star Loop over state vector elements... \star/
03781 #pragma omp parallel for default(none) shared(ctl,atm,obs,k,x0,yy0,n,m,iqa) private(i, j, h, x1, yy1, atm1,
       obs1)
03782
        for (j = 0; j < (int) n; j++) {
03783
03784
           /* Allocate... */
          x1 = gsl_vector_alloc(n);
yy1 = gsl_vector_alloc(m);
03785
03786
          ALLOC(atm1, atm_t, 1);
03787
03788
          ALLOC(obs1, obs_t, 1);
03789
03790
          /* Set perturbation size... */
03791
          if (iqa[j] == IDXP)
03792
            h = GSL_MAX(fabs(0.01 * gsl_vector_get(x0, (size_t) j)), 1e-7);
03793
          else if (iqa[j] == IDXT)
            h = 1;
03794
03795
          else if (iqa[j] >= IDXQ(0) \&\& iqa[j] < IDXQ(ctl->nq))
03796
            h = GSL_MAX(fabs(0.01 * gsl_vector_get(x0, (size_t) j)), 1e-15);
03797
          else if (iqa[j] >= IDXK(0) && iqa[j] < IDXK(ctl->nw))
03798
            h = 1e-4;
03799
          else
03800
            ERRMSG("Cannot set perturbation size!");
03801
          /* Disturb state vector element... */
03803
          gsl_vector_memcpy(x1, x0);
03804
          gsl_vector_set(x1, (size_t) j, gsl_vector_get(x1, (size_t) j) + h);
03805
          copy_atm(ctl, atm1, atm, 0);
03806
          copy_obs(ctl, obs1, obs, 0);
03807
          x2atm(ctl, x1, atm1);
03808
03809
           /* Compute radiance for disturbed atmospheric data... */
03810
          formod(ctl, atml, obsl);
03811
03812
          /\star Compose measurement vector for disturbed radiance data... \star/
          obs2y(ctl, obs1, yy1, NULL, NULL);
03813
03814
03815
          /* Compute derivatives... */
03816
          for (i = 0; i < m; i++)
03817
            gsl_matrix_set(k, i, (size_t) j,
03818
                            (gsl_vector_get(yy1, i) - gsl_vector_get(yy0, i)) / h);
03819
03820
          /* Free... */
03821
          gsl_vector_free(x1);
03822
          gsl_vector_free(yy1);
03823
           free(atm1);
03824
          free (obs1);
03825
03826
03827
        /* Free... */
03828
        gsl_vector_free(x0);
03829
        gsl_vector_free(yy0);
03830
       free(iqa);
03831 }
```

Here is the call graph for this function:



5.5.2.28 int locate_irr (double *xx, int n, double x)

Find array index for irregular grid.

Definition at line 3835 of file jurassic.c.

```
03838
03839
03840
         int i, ilo, ihi;
03841
         ilo = 0;
ihi = n - 1;
i = (ihi + ilo) >> 1;
03842
03843
03844
03845
         if (xx[i] < xx[i + 1])
  while (ihi > ilo + 1) {
   i = (ihi + ilo) >> 1;
03846
03847
03848
               <u>if</u> (xx[i] > x)
03849
03850
                 ihi = i;
               else
03851
03852
                 ilo = i;
03853
         } else
            while (ihi > ilo + 1) {
03854
             i = (ihi + ilo) >> 1;
if (xx[i] <= x)
03856
03857
                 ihi = i;
03858
               else
03859
                 ilo = i;
03860
03861
03862
         return ilo;
03863 }
```

```
5.5.2.29 int locate_reg ( double *xx, int n, double x )
```

Find array index for regular grid.

Definition at line 3867 of file jurassic.c.

```
03870
03871
03872
          int i;
03873
         /* Calculate index... */
i = (int) ((x - xx[0]) / (xx[1] - xx[0]));
03874
03875
03877
          /* Check range... */
03878
          <u>if</u> (i < 0)
         i = 0;
else if (i >= n - 2)
i = n - 2;
03879
03880
03881
03883
         return i;
03884 }
```

5.5.2.30 int locate_tbl (float *xx, int n, double x)

Find array index in float array.

Definition at line 3888 of file jurassic.c.

```
03891
                   {
03892
03893
        int i, ilo, ihi;
03894
       ilo = 0;
ihi = n - 1;
03895
03896
        i = (ihi + ilo) >> 1;
03897
03898
        while (ihi > ilo + 1) {
        i = (ihi + ilo) >> 1;
03900
         if (xx[i] > x)
03901
03902
            ihi = i;
         else
03903
03904
            ilo = i;
03905
        }
03906
03907
        return ilo;
03908 }
```

5.5.2.31 size_t obs2y (ctl_t * ctl, obs_t * obs, gsl_vector * y, int * ida, int * ira)

Compose measurement vector.

Definition at line 3912 of file jurassic.c.

```
03917
                    {
03918
03919
        int id, ir;
03920
03921
        size_t m = 0;
03922
03923
        /* Determine measurement vector... */
03924
        for (ir = 0; ir < obs->nr; ir++)
03925
         for (id = 0; id < ctl->nd; id++)
03926
             if (gsl_finite(obs->rad[id][ir])) {
              if (y != NULL)
   gsl_vector_set(y, m, obs->rad[id][ir]);
if (ida != NULL)
   ida[m] = id;
03927
03928
03929
03930
               if (ira != NULL)
03932
                 ira[m] = ir;
03933
               m++;
             }
03934
03935
03936
        return m:
03937 }
```

5.5.2.32 double planck (double t, double nu)

Compute Planck function.

Definition at line 3941 of file jurassic.c.

```
03943 {
03944
03945 return C1 * POW3(nu) / gsl_expm1(C2 * nu / t);
03946 }
```

5.5.2.33 void raytrace ($ctl_t * ctl$, $atm_t * atm$, $obs_t * obs$, $los_t * los$, int ir)

Do ray-tracing to determine LOS.

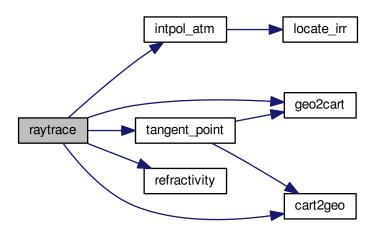
Definition at line 3950 of file jurassic.c.

```
03955
03956
03957
        double cosa, d, dmax, dmin = 0, ds, ex0[3], ex1[3], frac, h = 0.02, k[NW],
03958
          lat, lon, n, naux, ng[3], norm, p, q[NG], t, x[3], xh[3],
03959
          xobs[3], xvp[3], z = 1e99, zmax, zmin, zrefrac = 60;
03960
03961
        int i, ig, ip, iw, stop = 0;
03962
        /* Initialize... */
03964
        los->np = 0;
03965
        los \rightarrow tsurf = -999;
03966
        obs->tpz[ir] = obs->vpz[ir];
        obs->tplon[ir] = obs->vplon[ir];
03967
        obs->tplat[ir] = obs->vplat[ir];
03968
03969
03970
        /* Get altitude range of atmospheric data... */
03971
        gsl_stats_minmax(&zmin, &zmax, atm->z, 1, (size_t) atm->np);
03972
03973
        /* Check observer altitude... */
03974
        if (obs->obsz[ir] < zmin)</pre>
03975
          ERRMSG("Observer below surface!");
03976
03977
        /\star Check view point altitude... \star/
03978
        if (obs->vpz[ir] > zmax)
03979
          return;
03980
03981
        /* Determine Cartesian coordinates for observer and view point... */
03982
        geo2cart(obs->obsz[ir], obs->obslon[ir], obs->obslat[ir], xobs);
03983
        geo2cart(obs->vpz[ir], obs->vplon[ir], obs->vplat[ir], xvp);
03984
03985
        /\star Determine initial tangent vector... \star/
        for (i = 0; i < 3; i++)
  ex0[i] = xvp[i] - xobs[i];</pre>
03986
03987
03988
        norm = NORM(ex0);
03989
        for (i = 0; i < 3; i++)</pre>
03990
          ex0[i] /= norm;
03991
03992
        /* Observer within atmosphere... */
        for (i = 0; i < 3; i++)
03993
          x[i] = xobs[i];
03995
03996
        /\star Observer above atmosphere (search entry point)... \star/
03997
        if (obs->obsz[ir] > zmax) {
03998
          dmax = norm;
03999
          while (fabs(dmin - dmax) > 0.001) {
04000
            d = (dmax + dmin) / 2;
04001
            for (i = 0; i < 3; i++)</pre>
04002
              x[i] = xobs[i] + d * ex0[i];
04003
            cart2geo(x, &z, &lon, &lat);
            if (z <= zmax && z > zmax - 0.001)
04004
04005
              break;
            if (z < zmax - 0.0005)
04006
04007
              dmax = d;
04008
            else
04009
              dmin = d;
04010
04011
        }
04012
04013
        /* Ray-tracing... */
```

```
04014
        while (1) {
04015
04016
           /* Set step length... */
04017
           ds = ctl->rayds;
           if (ctl->raydz > 0) {
04018
            norm = NORM(x);
04019
             for (i = 0; i < 3; i++)
04020
04021
               xh[i] = x[i] / norm;
04022
             cosa = fabs(DOTP(ex0, xh));
04023
             if (cosa != 0)
               ds = GSL_MIN(ctl->rayds, ctl->raydz / cosa);
04024
04025
04026
04027
           /* Determine geolocation... */
04028
           cart2geo(x, &z, &lon, &lat);
04029
           /\star Check if LOS hits the ground or has left atmosphere... \star/
04030
04031
           if (z < zmin || z > zmax)
            stop = (z < zmin ? 2 : 1);
04032
04033
             frac =
               ((z <
04034
04035
                 zmin ? zmin : zmax) - los->z[los->np - 1]) / (z - los->z[los->np - 1])
04036
                                                                                 11);
             04037
04038
04039
             for (i = 0; i < 3; i++)
04040
              x[i] = xh[i] + frac * (x[i] - xh[i]);
04041
             cart2geo(x, &z, &lon, &lat);
04042
             los->ds[los->np - 1] = ds * frac;
04043
             ds = 0;
04044
04045
04046
           /* Interpolate atmospheric data... */
04047
           intpol_atm(ctl, atm, z, &p, &t, q, k);
04048
04049
           /* Save data... */
           los->lon[los->np] = lon;
los->lat[los->np] = lat;
04050
04051
04052
           los \rightarrow z[los \rightarrow np] = z;
04053
           los \rightarrow p[los \rightarrow np] = p;
04054
           los \rightarrow t[los \rightarrow np] = t;
           for (ig = 0; ig < ctl->ng; ig++)
04055
          los->q[ig][los->np] = q[ig];
for (iw = 0; iw < ctl->nw; iw++)
los->k[iw][los->np] = k[iw];
04056
04057
04058
04059
           los->ds[los->np] = ds;
04060
04061
           /\star Increment and check number of LOS points... \star/
           if ((++los->np) > NLOS)
04062
            ERRMSG("Too many LOS points!");
04063
04064
04065
           /* Check stop flag... */
04066
           if (stop) {
04067
             los->tsurf = (stop == 2 ? t : -999);
04068
             break;
04069
           }
04070
04071
           /* Determine refractivity... */
04072
           if (ctl->refrac && z <= zrefrac)</pre>
04073
            n = 1 + refractivity(p, t);
04074
           else
04075
            n = 1;
04076
04077
           /* Construct new tangent vector (first term)... */
04078
           for (i = 0; i < 3; i++)
04079
             ex1[i] = ex0[i] * n;
04080
           /* Compute gradient of refractivity... */
04081
04082
           if (ctl->refrac && z <= zrefrac) {
             for (i = 0; i < 3; i++)
04084
               xh[i] = x[i] + 0.5 * ds * ex0[i];
             cart2geo(xh, &z, &lon, &lat);
04085
04086
             intpol_atm(ctl, atm, z, &p, &t, q, k);
             n = refractivity(p, t);
for (i = 0; i < 3; i++) {
   xh[i] += h;</pre>
04087
04088
04089
04090
               cart2geo(xh, &z, &lon, &lat);
04091
               intpol_atm(ctl, atm, z, &p, &t, q, k);
               naux = refractivity(p, t);
04092
               naux - rerractivity(p,
ng[i] = (naux - n) / h;
xh[i] -= h;
04093
04094
04095
04096
           } else
             for (i = 0; i < 3; i++)
04097
04098
               ng[i] = 0;
04099
04100
           /* Construct new tangent vector (second term) ... */
```

```
04101
             for (i = 0; i < 3; i++)</pre>
04102
               ex1[i] += ds * ng[i];
04103
04104
             /\star Normalize new tangent vector... \star/
             norm = NORM(ex1);
for (i = 0; i < 3; i++)
  ex1[i] /= norm;</pre>
04105
04106
04107
04108
04109
             /\star Determine next point of LOS... \star/
             for (i = 0; i < 3; i++)
  x[i] += 0.5 * ds * (ex0[i] + ex1[i]);</pre>
04110
04111
04112
            /* Copy tangent vector... */
for (i = 0; i < 3; i++)</pre>
04113
04114
04115
               ex0[i] = ex1[i];
04116
04117
04118
          /\star Get tangent point (to be done before changing segment lengths!)... \star/
          tangent_point(los, &obs->tpz[ir], &obs->tplon[ir], &obs->
04119
       tplat[ir]);
04120
04121
           /\star Change segment lengths according to trapezoid rule... \star/
          for (ip = los->np - 1; ip >= 1; ip--)
los->ds[ip] = 0.5 * (los->ds[ip - 1] + los->ds[ip]);
los->ds[0] *= 0.5;
04122
04123
04124
04125
04126
          /\star Compute column density... \star/
04127
          for (ip = 0; ip < los->np; ip++)
            for (ig = 0; ig < ctl->ng; ig++)
  los->u[ig][ip] = 10 * los->q[ig][ip] * los->p[ip]
  / (KB * los->t[ip]) * los->ds[ip];
04128
04129
04130
04131 }
```

Here is the call graph for this function:



5.5.2.34 void read_atm (const char * dirname, const char * filename, ctl_t * ctl, atm_t * atm)

Read atmospheric data.

Definition at line 4135 of file jurassic.c.

```
04139 {
04140
04141 FILE *in;
04142
04143 char file[LEN], line[LEN], *tok;
```

```
04144
04145
           int ig, iw;
04146
04147
            /* Init... */
04148
           atm->np = 0;
04149
04150
            /* Set filename... */
04151
            if (dirname != NULL)
04152
              sprintf(file, "%s/%s", dirname, filename);
04153
            else
              sprintf(file, "%s", filename);
04154
04155
           /* Write info... */
04156
04157
           printf("Read atmospheric data: %s\n", file);
04158
04159
            /* Open file... */
           if (!(in = fopen(file, "r")))
04160
              ERRMSG("Cannot open file!");
04161
04162
04163
           /* Read line... */
04164
           while (fgets(line, LEN, in)) {
04165
              /* Read data... */

TOK(line, tok, "%lg", atm->time[atm->np]);

TOK(NULL, tok, "%lg", atm->z[atm->np]);

TOK(NULL, tok, "%lg", atm->lon[atm->np]);

TOK(NULL, tok, "%lg", atm->lat[atm->np]);

TOK(NULL, tok, "%lg", atm->p[atm->np]);

TOK(NULL, tok, "%lg", atm->t[atm->np]);

TOK(NULL, tok, "%lg", atm->p[atm->np]);

for (ig = 0; ig < ctl->ng; ig++)

TOK(NULL, tok, "%lg", atm->q[ig][atm->np]);

for (iw = 0; iw < ctl->nw; iw++)

TOK(NULL, tok, "%lg", atm->k[iw][atm->np]);
04166
04167
04168
04169
04170
04171
04172
04173
04174
04175
04176
04177
              /* Increment data point counter... */
if ((++atm->np) > NP)
04178
04179
                  ERRMSG("Too many data points!");
04180
04181
04182
04183
           /* Close file... */
04184
           fclose(in);
04185
04186
           /* Check number of points... */
04187
            if (atm->np < 1)
               ERRMSG("Could not read any data!");
04188
04189 }
```

5.5.2.35 void read_ctl (int argc, char * argv[], ctl_t * ctl)

Read forward model control parameters.

Definition at line 4193 of file jurassic.c.

```
04196
04197
04198
       int id, ig, iw;
04199
       /* Write info... */
04200
       04201
04202
               argv[0], __DATE__, __TIME__);
04204
04205
       /* Emitters... */
       ctl->ng = (int) scan_ctl(argc, argv, "NG", -1, "0", NULL);
if (ctl->ng < 0 || ctl->ng > NG)
04206
04207
         ERRMSG("Set 0 <= NG <= MAX!");
04208
        for (ig = 0; ig < ctl->ng; ig++)
04209
         scan_ctl(argc, argv, "EMITTER", ig, "", ctl->emitter[ig]);
04210
04211
       /* Radiance channels... */
04212
       ctl->nd = (int) scan_ctl(argc, argv, "ND", -1, "0", NULL);
04213
       if (ctl->nd < 0 || ctl->nd > ND)
04214
         ERRMSG("Set 0 <= ND <= MAX!");</pre>
04215
04216
       for (id = 0; id < ctl->nd; id++)
04217
         ctl->nu[id] = scan_ctl(argc, argv, "NU", id, "", NULL);
04218
        /* Spectral windows... */
04219
       ctl->nw = (int) scan_ctl(argc, argv, "NW", -1, "1", NULL);
if (ctl->nw < 0 || ctl->nw > NW)
04220
04221
         ERRMSG("Set 0 <= NW <= MAX!");</pre>
```

```
for (id = 0; id < ctl->nd; id++)
04224
              ctl->window[id] = (int) scan_ctl(argc, argv, "WINDOW", id, "0", NULL);
04225
           /* Emissivity look-up tables... */
scan_ctl(argc, argv, "TBLBASE", -1, "-", ctl->tblbase);
04226
04227
04228
04229
            /* Hydrostatic equilibrium... */
04230
            ctl->hydz = scan_ctl(argc, argv, "HYDZ", -1, "-999", NULL);
04231
04232
            /* Continua... */
           ctl->ctm_co2 = (int) scan_ctl(argc, argv, "CTM_CO2", -1, "1", NULL); ctl->ctm_h2o = (int) scan_ctl(argc, argv, "CTM_H2O", -1, "1", NULL); ctl->ctm_n2 = (int) scan_ctl(argc, argv, "CTM_N2", -1, "1", NULL); ctl->ctm_o2 = (int) scan_ctl(argc, argv, "CTM_O2", -1, "1", NULL);
04233
04234
04235
04236
04237
04238
           ctl->refrac = (int) scan_ctl(argc, argv, "REFRAC", -1, "1", NULL);
ctl->rayds = scan_ctl(argc, argv, "RAYDS", -1, "10", NULL);
ctl->raydz = scan_ctl(argc, argv, "RAYDZ", -1, "0.5", NULL);
04239
04240
04241
04242
            /* Field of view... */
scan_ctl(argc, argv, "FOV", -1, "-", ctl->fov);
04243
04244
04245
            /* Retrieval interface... */
04246
           /* Retrieval interface... */
ctl->retp_zmin = scan_ctl(argc, argv, "RETP_ZMIN", -1, "-999", NULL);
ctl->retp_zmax = scan_ctl(argc, argv, "RETP_ZMAX", -1, "-999", NULL);
ctl->rett_zmin = scan_ctl(argc, argv, "RETT_ZMIN", -1, "-999", NULL);
ctl->rett_zmax = scan_ctl(argc, argv, "RETT_ZMAX", -1, "-999", NULL);
04247
04248
04249
04250
04251
            for (ig = 0; ig < ctl->ng; ig++) {
             ctl->retq_zmin[ig] = scan_ctl(argc, argv, "RETO_ZMIN", ig, "-999", NULL);
ctl->retq_zmax[ig] = scan_ctl(argc, argv, "RETO_ZMAX", ig, "-999", NULL);
04252
04253
04254
04255
            for (iw = 0; iw < ctl->nw; iw++) {
04256
             ctl->retk_zmin[iw] = scan_ctl(argc, argv, "RETK_ZMIN", iw, "-999", NULL);
              ctl->retk_zmax[iw] = scan_ctl(argc, argv, "RETK_ZMAX", iw, "-999", NULL);
04257
04258
04259
04260
            /* Output flags... */
04261
            ctl->write_bbt = (int) scan_ctl(argc, argv, "WRITE_BBT", -1, "0", NULL);
04262
            ctl->write_matrix =
                (int) scan_ctl(argc, argv, "WRITE_MATRIX", -1, "0", NULL);
04263
04264 }
```

Here is the call graph for this function:



5.5.2.36 void read_matrix (const char * dirname, const char * filename, gsl_matrix * matrix)

Read matrix.

Definition at line 4268 of file jurassic.c.

```
04271
                              {
04272
04273
       FILE *in;
04275
       char dum[LEN], file[LEN], line[LEN];
04276
04277
       double value;
04278
04279
       int i, i;
04280
04281
       /* Set filename... */
```

```
04282
        if (dirname != NULL)
04283
          sprintf(file, "%s/%s", dirname, filename);
04284
        else
04285
          sprintf(file, "%s", filename);
04286
        /* Write info... */
04287
        printf("Read matrix: %s\n", file);
04288
04289
04290
         /* Open file... */
        if (!(in = fopen(file, "r")))
04291
          ERRMSG("Cannot open file!");
04292
04293
04294
        /* Read data... */
04295
        gsl_matrix_set_zero(matrix);
04296
        while (fgets(line, LEN, in))
04297
         if (sscanf(line, "%d %s %s %s %s %d %s %s %s %s %s %lg",
04298
                       &i, dum, dum, dum, dum, dum,
            &j, dum, dum, dum, dum, dum, &value) == 13)
gsl_matrix_set(matrix, (size_t) i, (size_t) j, value);
04299
04301
04302
         /* Close file... */
04303
        fclose(in);
04304 }
```

5.5.2.37 void read_obs (const char * dirname, const char * filename, ctl_t * ctl, obs_t * obs)

Read observation data.

Definition at line 4308 of file jurassic.c.

```
04312
04313
04314
             FILE *in:
04315
04316
            char file[LEN], line[LEN], *tok;
04317
04318
04319
04320
             /* Init... */
04321
             obs->nr = 0;
04322
04323
             /* Set filename... */
04324
             if (dirname != NULL)
04325
                sprintf(file, "%s/%s", dirname, filename);
04326
                sprintf(file, "%s", filename);
04327
04328
04329
             /* Write info... */
04330
             printf("Read observation data: %s\n", file);
04331
04332
             /* Open file... */
             if (!(in = fopen(file, "r")))
04333
                ERRMSG("Cannot open file!");
04334
04335
04336
             /* Read line... */
04337
             while (fgets(line, LEN, in)) {
04338
                /* Read data... */
TOK(line, tok, "%lg", obs->time[obs->nr]);
TOK(NULL, tok, "%lg", obs->obsz[obs->nr]);
04339
04340
04341
                TOK (NULL, tok, "%lg", obs->obsz[obs->nr]);
TOK (NULL, tok, "%lg", obs->obslon[obs->nr]);
TOK (NULL, tok, "%lg", obs->obslat[obs->nr]);
TOK (NULL, tok, "%lg", obs->vpz[obs->nr]);
TOK (NULL, tok, "%lg", obs->vplon[obs->nr]);
TOK (NULL, tok, "%lg", obs->vplon[obs->nr]);
TOK (NULL, tok, "%lg", obs->tpz[obs->nr]);
TOK (NULL, tok, "%lg", obs->tplon[obs->nr]);
TOK (NULL, tok, "%lg", obs->tplat[obs->nr]);
TOK (NULL, tok, "%lg", obs->tplat[obs->nr]);
for (id = 0; id < ctl->nd; id+)
TOK (NULL, tok, "%lg", obs->rad[id][obs->nr]);
for (id = 0; id < ctl->nd; id+)
TOK (NULL, tok, "%lg", obs->tpulid][obs->nr]);
04342
04343
04344
04345
04346
04347
04348
04349
04350
04351
04352
04353
04354
04355
                 /* Increment counter... */
04356
                 if ((++obs->nr) > NR)
                    ERRMSG("Too many rays!");
04357
04358
04359
04360
             /* Close file... */
04361
             fclose(in);
```

```
04362

04363  /* Check number of points... */

04364  if (obs->nr < 1)

04365  ERRMSG("Could not read any data!");

04366 }
```

5.5.2.38 void read_shape (const char * filename, double * x, double * y, int * n)

Read shape function.

Definition at line 4370 of file jurassic.c.

```
04374
04375
04376
        FILE *in;
04377
04378
        char line[LEN];
04379
04380
        /* Write info... */
04381
        printf("Read shape function: %s\n", filename);
04382
04383
        /* Open file... */
        if (!(in = fopen(filename, "r")))
04384
          ERRMSG("Cannot open file!");
04386
04387
        /* Read data... */
04388
        *n = 0;
        while (fgets(line, LEN, in))
  if (sscanf(line, "%lg %lg", &x[*n], &y[*n]) == 2)
  if ((++(*n)) > NSHAPE)
04389
04390
04391
04392
              ERRMSG("Too many data points!");
04393
04394
        /* Check number of points... */
        if (*n < 1)
04395
          ERRMSG("Could not read any data!");
04396
04397
04398
        /* Close file... */
04399 fclose(in);
04400 }
```

5.5.2.39 double refractivity (double p, double t)

Compute refractivity (return value is n - 1).

Definition at line 4404 of file jurassic.c.

```
04406

04407

04408  /* Refractivity of air at 4 to 15 micron... */

04409  return 7.753e-05 * p / t;

04410 }
```

5.5.2.40 double scan_ctl (int argc, char * argv[], const char * varname, int arridx, const char * defvalue, char * value)

Search control parameter file for variable entry.

Definition at line 4414 of file jurassic.c.

```
04420
04421
04422
        FILE *in = NULL;
04423
04424
         char dummy[LEN], fullname1[LEN], fullname2[LEN], line[LEN],
04425
          msg[2 * LEN], rvarname[LEN], rval[LEN];
04426
04427
04428
        /* Open file... */
if (argv[1][0] != '-')
04429
04430
         if (!(in = fopen(argv[1], "r")))
04431
             ERRMSG("Cannot open file!");
04432
04433
04434
         /* Set full variable name... */
04435
        if (arridx >= 0) {
         sprintf(fullname1, "%s[%d]", varname, arridx);
sprintf(fullname2, "%s[*]", varname);
04436
04437
04438
        } else {
          sprintf(fullname1, "%s", varname);
sprintf(fullname2, "%s", varname);
04439
04440
04441
04442
04443
         /* Read data... */
04444
        if (in != NULL)
         while (fgets(line, LEN, in))
04446
             if (sscanf(line, "%s %s %s", rvarname, dummy, rval) == 3)
              if (strcasecmp(rvarname, fullname1) == 0 ||
04447
04448
                   strcasecmp(rvarname, fullname2) == 0) {
04449
                  contain = 1;
04450
                 break:
04451
               }
04452
        for (i = 1; i < argc - 1; i++)</pre>
04453
         if (strcasecmp(argv[i], fullname1) == 0 ||
             strcasecmp(argv[i], fullname2) == 0) {
sprintf(rval, "%s", argv[i + 1]);
04454
04455
04456
             contain = 1;
04457
             break;
04458
04459
04460
        /* Close file... */
        if (in != NULL)
04461
04462
          fclose(in);
04463
04464
        /* Check for missing variables... */
04465
         if (!contain) {
         if (strlen(defvalue) > 0)
   sprintf(rval, "%s", defvalue);
04466
04467
           else {
04468
            sprintf(msg, "Missing variable %s!\n", fullname1);
04469
             ERRMSG (msg);
04471
04472
04473
04474
        /* Write info... */
        printf("%s = %s\n", fullname1, rval);
04475
04477
        /* Return values... */
04478
        if (value != NULL)
04479
          sprintf(value, "%s", rval);
04480
        return atof(rval);
04481 }
```

5.5.2.41 void tangent_point (los t * los, double * tpz, double * tplon, double * tplat)

Find tangent point of a given LOS.

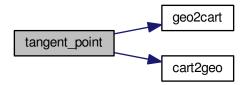
Definition at line 4485 of file jurassic.c.

```
04489
04490
04491
        double a, b, c, dummy, v[3], v0[3], v2[3], x, x1, x2, yy0, yy1, yy2;
04492
04493
       size_t i, ip;
04494
04495
        /\star Find minimum altitude... \star/
04496
       ip = gsl_stats_min_index(los->z, 1, (size_t) los->np);
04497
04498
       /* Nadir or zenith... */
04499
       if (ip <= 0 || ip >= (size_t) los->np - 1) {
```

```
*tpz = los -> z[los -> np - 1];
          *tplon = los->lon[los->np - 1];

*tplat = los->lat[los->np - 1];
04501
04502
04503
04504
04505
        /* Limb... */
04506
        else {
04507
04508
           /* Determine interpolating polynomial y=a*x^2+b*x+c...*/
04509
           yy0 = los \rightarrow z[ip - 1];
          yy1 = los \rightarrow z[ip];
04510
           yy2 = los -> z[ip + 1];
04511
04512
           x1 = sqrt(POW2(los->ds[ip]) - POW2(yy1 - yy0));
04513
           x2 = x1 + sqrt(POW2(los->ds[ip + 1]) - POW2(yy2 - yy1));
04514
           a = 1 / (x1 - x2) * (-(yy0 - yy1) / x1 + (yy0 - yy2) / x2);
          b = -(yy0 - yy1) / x1 - a * x1;
04515
          c = yy0;
04516
04517
04518
           /* Get tangent point location... */
04519
           x = -b / (2 * a);
04520
           *tpz = a * x * x + b * x + c;
04521
           geo2cart(los->z[ip - 1], los->lon[ip - 1], los->lat[ip - 1], v0);
           geo2cart(los->z[ip + 1], los->lon[ip + 1], los->lat[ip + 1], v2);
04522
           for (i = 0; i < 3; i++)
v[i] = LIN(0.0, v0[i], x2, v2[i], x);</pre>
04523
04524
04525
           cart2geo(v, &dummy, tplon, tplat);
04526
04527 }
```

Here is the call graph for this function:



5.5.2.42 void time2jsec (int year, int mon, int day, int hour, int min, int sec, double remain, double * jsec)

Convert date to seconds.

Definition at line 4531 of file jurassic.c.

```
04539
                       {
04540
04541
       struct tm t0, t1;
04542
04543
        t0.tm_year = 100;
04544
        t0.tm\_mon = 0;
        t0.tm_mday = 1;
04545
        t0.tm_hour = 0;
04546
        t0.tm_min = 0;
04547
04548
        t0.tm\_sec = 0;
04549
04550
        t1.tm_year = year - 1900;
04551
        t1.tm_mon = mon - 1;
04552
        t1.tm_mday = day;
        t1.tm_hour = hour;
04553
04554
        t1.tm_min = min;
04555
        t1.tm_sec = sec;
04556
04557
        *jsec = (double) timegm(&t1) - (double) timegm(&t0) + remain;
04558 }
```

5.5.2.43 void timer (const char * name, const char * file, const char * func, int line, int mode)

Measure wall-clock time.

Definition at line 4562 of file jurassic.c.

```
04567
                   {
04568
04569
       static double w0[10];
04571
       static int 10[10], nt;
04572
04573
        /* Start new timer... */
04574
       if (mode == 1) {
        w0[nt] = omp_get_wtime();
10[nt] = line;
04575
             ((++nt) >= 10)
04577
         if
04578
            ERRMSG("Too many timers!");
04579
04580
04581
        /* Write elapsed time... */
04582
        else {
04583
04584
          /\star Check timer index... \star/
04585
         if (nt - 1 < 0)
           ERRMSG("Coding error!");
04586
04587
         /* Write elapsed time... */
04588
        printf("Timer '%s' (%s, %s, 1%d-%d): %.3f sec\n",
04590
                name, file, func, 10[nt - 1], line, omp_get_wtime() - w0[nt - 1]);
04591
04592
04593
       /* Stop timer... */
04594
        if (mode == 3)
04595
          nt--;
04596 }
```

5.5.2.44 void write_atm (const char * dirname, const char * filename, ctl_t * ctl, atm_t * atm)

Write atmospheric data.

Definition at line 4600 of file jurassic.c.

```
04604
04605
04606
        FILE *out;
04607
04608
        char file[LEN];
04609
04610
        int iq, ip, iw, n = 6;
04611
         /* Set filename...
04612
04613
        if (dirname != NULL)
          sprintf(file, "%s/%s", dirname, filename);
04614
04615
        else
04616
          sprintf(file, "%s", filename);
04617
04618
        /* Write info... */
04619
        printf("Write atmospheric data: %s\n", file);
04620
04621
        /* Create file... */
04622
        if (!(out = fopen(file, "w")))
          ERRMSG("Cannot create file!");
04623
04624
04625
        /* Write header... */
04626
        fprintf(out,
                 "# $1 = time (seconds since 2000-01-01T00:00Z) \n"
04627
                 "# $2 = altitude [km] \n"
04628
                 "# $3 = longitude [deg]\n
04629
04630
                 "# $4 = latitude [deg] \n"
04631
                 "# $5 = pressure [hPa] \n" "# $6 = temperature [K] \n");
        for (ig = 0; ig < ctl->ng; ig++)
  fprintf(out, "# $%d = %s volume mixing ratio\n", ++n, ctl->emitter[ig]);
for (iw = 0; iw < ctl->nw; iw++)
04632
04633
04634
04635
          fprintf(out, "# \$%d = window %d: extinction [1/km]\n", ++n, iw);
04636
```

```
04637
          /* Write data... */
04638
          for (ip = 0; ip < atm->np; ip++) {
             if (ip == 0 || atm->lat[ip] != atm->lat[ip - 1]
04639
             || atm->lon[ip] != atm->lon[ip - 1])
fprintf(out, "\n");
fprintf(out, "%.2f %g %g %g %g", atm->time[ip], atm->z[ip],
04640
04641
04642
                       atm->lon[ip], atm->lat[ip], atm->p[ip], atm->t[ip]);
04643
             for (ig = 0; ig < ctl->ng; ig++)
  fprintf(out, " %g", atm->q[ig][ip]);
04644
04645
             for (iw = 0; iw < ctl->nw; iw++)
  fprintf(out, " %g", atm->k[iw][ip]);
fprintf(out, "\n");
04646
04647
04648
04649
04650
04651
           /\star Close file... \star/
04652
          fclose(out);
04653 }
```

5.5.2.45 void write_matrix (const char * dirname, const char * filename, ctl_t * ctl, gsl_matrix * matrix, atm_t * atm, obs_t * obs, const char * rowspace, const char * colspace, const char * sort)

Write matrix.

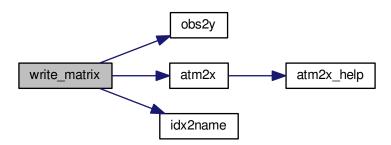
Definition at line 4657 of file jurassic.c.

```
04666
04667
04668
        FILE *out;
04669
        char file[LEN], quantity[LEN];
04671
04672
        int *cida, *ciqa, *cipa, *cira, *rida, *riqa, *ripa, *rira;
04673
04674
        size t i, j, nc, nr;
04675
04676
        /* Check output flag... */
04677
        if (!ctl->write_matrix)
04678
          return;
04679
        /* Allocate... */
04680
        ALLOC(cida, int, M);
04681
04682
        ALLOC(ciqa, int,
04683
              N);
04684
        ALLOC(cipa, int,
04685
              N);
        ALLOC(cira, int,
04686
04687
              M);
04688
        ALLOC(rida, int,
04689
              M);
04690
        ALLOC(riqa, int,
04691
              N);
        ALLOC(ripa, int,
04692
04693
              N);
04694
        ALLOC(rira, int,
04695
              M);
04696
04697
        /* Set filename... */
        if (dirname != NULL)
04698
         sprintf(file, "%s/%s", dirname, filename);
04699
04700
        else
04701
          sprintf(file, "%s", filename);
04702
04703
        /* Write info... */
04704
        printf("Write matrix: %s\n", file);
04705
04706
        /* Create file... */
04707
        if (!(out = fopen(file, "w")))
04708
          ERRMSG("Cannot create file!");
04709
04710
        /* Write header (row space)... */
04711
        if (rowspace[0] == 'y') {
04712
04713
          fprintf(out,
04714
                   "# $1 = Row: index (measurement space) \n"
04715
                   "# $2 = Row: channel wavenumber [cm^-1]\n"
04716
                   "# \$3 = \text{Row: time (seconds since 2000-01-01T00:00Z)} \n"
                   "# $4 = Row: view point altitude [km]\n"
"# $5 = Row: view point longitude [deg]\n"
04717
04718
04719
                   "# $6 = Row: view point latitude [deg]\n");
04720
```

```
04721
           /* Get number of rows...
04722
          nr = obs2y(ctl, obs, NULL, rida, rira);
04723
04724
        } else {
04725
04726
          fprintf(out,
04727
                    "# $1 = Row: index (state space)\n"
04728
                    "# $2 = Row: name of quantity n"
04729
                    "# \$3 = Row: time (seconds since 2000-01-01T00:00Z)\n"
04730
                    "# $4 = Row: altitude [km]\n"
                    "# $5 = \text{Row: longitude [deg]} \n" "# $6 = \text{Row: latitude [deg]} \n");
04731
04732
04733
           /* Get number of rows... */
04734
          nr = atm2x(ctl, atm, NULL, riqa, ripa);
04735
04736
04737
         /\star Write header (column space)... \star/
04738
        if (colspace[0] == 'y') {
04740
          fprintf(out,
04741
                    "# \$7 = \text{Col: index (measurement space)} \n"
                    "# $8 = Col: channel wavenumber [cm^-1]\n"
04742
                    "# $9 = Col: time (seconds since 2000-01-01T00:00Z)\n"
04743
                    "# $10 = Col: view point altitude [km]\n" "# $11 = Col: view point longitude [deg]\n"
04744
04745
04746
                    "# $12 = Col: view point latitude [deg]\n");
04747
           /\star Get number of columns... \star/
04748
04749
          nc = obs2y(ctl, obs, NULL, cida, cira);
04750
04751
        } else {
04752
04753
           fprintf(out,
04754
                    "# $7 = Col: index (state space) \n"
                    "# $8 = Col: name of quantity n"
04755
                    "# $9 = Col: time (seconds since 2000-01-01T00:00Z)\n"
04756
04757
                    "# $10 = Col: altitude [km] \n"
04758
                    "# $11 = Col: longitude [deg]\n" "# $12 = Col: latitude [deg]\n");
04759
04760
           /* Get number of columns... */
04761
          nc = atm2x(ctl, atm, NULL, ciqa, cipa);
04762
04763
        /* Write header entry... */
fprintf(out, "# $13 = Matrix element\n\n");
04764
04765
04766
04767
         /* Write matrix data... */
04768
        i = j = 0;
        while (i < nr && j < nc) {
04769
04770
04771
           /* Write info about the row... */
          if (rowspace[0] == 'y')
  fprintf(out, "%d %g %.2f %g %g %g",
04772
04773
04774
                      (int) i, ctl->nu[rida[i]],
04775
                      obs->time[rira[i]], obs->vpz[rira[i]],
04776
                      obs->vplon[rira[i]], obs->vplat[rira[i]]);
04777
             idx2name(ctl, riqa[i], quantity);
fprintf(out, "%d %s %.2f %g %g %g", (int) i, quantity,
04778
04779
04780
                     atm->time[ripa[i]], atm->z[ripa[i]],
04781
                      atm->lon[ripa[i]], atm->lat[ripa[i]]);
04782
           }
04783
04784
           /\star Write info about the column... \star/
           if (colspace[0] == 'y')
  fprintf(out, " %d %g %.2f %g %g %g",
04785
04786
04787
                      (int) j, ctl->nu[cida[j]],
04788
                      obs->time[cira[j]], obs->vpz[cira[j]],
04789
                      obs->vplon[cira[j]], obs->vplat[cira[j]]);
04790
           else {
             idx2name(ctl, ciqa[j], quantity);
fprintf(out, " %d %s %.2f %g %g %g", (int) j, quantity,
04791
04792
                      atm->time[cipa[j]], atm->z[cipa[j]],
04793
04794
                      atm->lon[cipa[j]], atm->lat[cipa[j]]);
04795
           }
04796
04797
           /* Write matrix entry... */
04798
           fprintf(out, " %g\n", gsl_matrix_get(matrix, i, j));
04799
           /* Set matrix indices... */
04800
           if (sort[0] == 'r') {
04801
04802
             j++;
04803
             if (j >= nc) {
04804
              j = 0;
04805
               i++:
               fprintf(out, "\n");
04806
04807
```

```
04808
          } else {
04809
            i++;
            if (i >= nr) {
  i = 0;
04810
04811
04812
               j++;
04813
              fprintf(out, "\n");
04814
04815
04816
04817
        /* Close file... */
04818
04819
        fclose(out);
04820
04821
        /* Free... */
04822
        free(cida);
04823
        free(ciqa);
04824
        free(cipa);
04825
        free(cira);
04826
        free(rida);
04827
        free(riqa);
04828
        free(ripa);
04829
        free(rira);
04830 }
```

Here is the call graph for this function:



5.5.2.46 void write_obs (const char * dirname, const char * filename, ctl_t * ctl, obs_t * obs)

Write observation data.

Definition at line 4834 of file jurassic.c.

```
04838
04839
04840
        FILE *out;
04841
04842
        char file[LEN];
04843
04844
        int id, ir, n = 10;
04845
04846
         /∗ Set filename...
        if (dirname != NULL)
   sprintf(file, "%s/%s", dirname, filename);
04847
04848
04849
        else
04850
          sprintf(file, "%s", filename);
04851
04852
         /* Write info... */
04853
        printf("Write observation data: %s\n", file);
04854
        /* Create file... */
if (!(out = fopen(file, "w")))
04855
04856
04857
          ERRMSG("Cannot create file!");
04858
```

```
04859
          /* Write header... */
04860
          fprintf(out,
04861
                    "# $1 = time (seconds since 2000-01-01T00:00Z) \n"
                    "# $2 = observer altitude [km] \n"
04862
                    "# $3 = observer longitude [deg] \n"
04863
                    "# $4 = observer latitude [deg]\n"
04864
                    "# $5 = view point altitude [km]\n"
04865
04866
                    "# $6 = view point longitude [deg]\n"
04867
                    "# $7 = view point latitude [deg] n"
                    "# $8 = tangent point altitude [km]\n"
04868
                    "# $9 = tangent point longitude [deg]\n"
04869
                    "# $10 = tangent point latitude [deg]\n");
04870
         for (id = 0; id < ctl->nd; id++)
04871
04872
          fprintf(out, "# \$%d = channel %g: radiance [W/(m^2 sr cm^-1)]\n",
04873
                      ++n, ctl->nu[id]);
         for (id = 0; id < ctl->nd; id++)
  fprintf(out, "# $%d = channel %g: transmittance\n", ++n, ctl->nu[id]);
04874
04875
04876
          /* Write data... */
04877
04878
          for (ir = 0; ir < obs->nr; ir++) {
            if (ir == 0 || obs->time[ir] != obs->time[ir - 1])
   fprintf(out, "\n");
fprintf(out, "%.2f %g %g %g %g %g %g %g %g %g", obs->time[ir],
04879
04880
04881
                      obs->obsz[ir], obs->obslon[ir], obs->obslat[ir],
obs->vpz[ir], obs->vplon[ir], obs->vplat[ir],
obs->tpz[ir], obs->tplon[ir], obs->tplat[ir]);
04882
04883
04884
            for (id = 0; id < ctl->nd; id++)
  fprintf(out, " %g", obs->rad[id][ir]);
04885
04886
            for (id = 0; id < ctl->nd; id+)
    fprintf(out, " %g", obs->tau[id][ir]);
fprintf(out, "\n");
04887
04888
04889
04890
04891
04892
          /* Close file... */
04893
         fclose(out);
04894 }
```

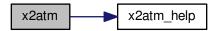
5.5.2.47 void x2atm (ctl_t * ctl, gsl_vector * x, atm_t * atm)

Decompose parameter vector or state vector.

Definition at line 4898 of file jurassic.c.

```
04902
04903
        int ig, iw;
04904
04905
       size_t n = 0;
04906
04907
        /* Set pressure... */
       x2atm_help(atm, ctl->retp_zmin, ctl->retp_zmax, atm->
04908
     p, x, &n);
04909
04910
        /* Set temperature... */
04911
       x2atm_help(atm, ctl->rett_zmin, ctl->rett_zmax, atm->
      t, x, &n);
04912
04913
        /\star Set volume mixing ratio... \star/
04914
        for (ig = 0; ig < ctl->ng; ig++)
04915
         x2atm_help(atm, ctl->retq_zmin[ig], ctl->retq_zmax[ig],
04916
                     atm->q[ig], x, &n);
04917
04918
       /* Set extinction... */
04919
        for (iw = 0; iw < ctl->nw; iw++)
04920
          x2atm_help(atm, ctl->retk_zmin[iw], ctl->retk_zmax[iw],
04921
                     atm->k[iw], x, &n);
04922 }
```

Here is the call graph for this function:



5.5.2.48 void x2atm_help (atm_t * atm, double zmin, double zmax, double * value, gsl_vector * x, size_t * n)

Extract elements from state vector.

Definition at line 4926 of file jurassic.c.

```
04932
04933
04934
         int ip;
04935
04936
         /* Extract state vector elements... */
         for (ip = 0; ip < atm->np; ip++)
  if (atm->z[ip] >= zmin && atm->z[ip] <= zmax) {</pre>
04937
04939
             value[ip] = gsl_vector_get(x, *n);
04940
               (*n)++;
            }
04941
04942 }
```

5.5.2.49 void y2obs ($ctl_t * ctl$, $gsl_vector * y$, $obs_t * obs$)

Decompose measurement vector.

Definition at line 4946 of file jurassic.c.

```
04949
04950
04951
          int id, ir;
04952
04953
          size_t m = 0;
04955
           /\star Decompose measurement vector... \star/
          for (ir = 0; ir < obs->nr; ir++)
  for (id = 0; id < ctl->nd; id++)
   if (gsl_finite(obs->rad[id][ir])) {
04956
04957
04958
04959
                  obs->rad[id][ir] = gsl_vector_get(y, m);
04960
                  m++;
04961
04962 }
```

5.6 jurassic.h

```
00001 /*
00002
        This file is part of JURASSIC.
00003
00004
        JURASSIC is free software: you can redistribute it and/or modify
00005
        it under the terms of the GNU General Public License as published by
00006
        the Free Software Foundation, either version 3 of the License, or
00007
        (at your option) any later version.
80000
00009
        {\tt JURASSIC} is distributed in the hope that it will be useful,
        but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
00010
00011
00012
        GNU General Public License for more details.
00013
00014
        You should have received a copy of the GNU General Public License
00015
        along with JURASSIC. If not, see <a href="http://www.gnu.org/licenses/">http://www.gnu.org/licenses/</a>.
00016
00017
        Copright (C) 2003-2015 Forschungszentrum Juelich GmbH
00018 */
00019
00034 #include <gsl/gsl_math.h>
00035 #include <gsl/gsl_blas.h>
00036 #include <gsl/gsl_linalg.h>
00037 #include <gsl/gsl_statistics.h>
00038 #include <math.h>
00039 #include <omp.h>
00040 #include <stdio.h>
00041 #include <stdlib.h>
00042 #include <string.h>
00043 #include <time.h>
00044
00045 /* -----
```

5.6 jurassic.h 221

```
Macros...
00047
00048
00050 #define ALLOC(ptr, type, n)
00051    if((ptr=malloc((size_t)(n)*sizeof(type)))==NULL)
         ERRMSG("Out of memory!");
00052
00055 #define DIST(a, b) sqrt(DIST2(a, b))
00056
00058 #define DIST2(a, b)
        ((a[0]-b[0])*(a[0]-b[0])+(a[1]-b[1])*(a[1]-b[1])+(a[2]-b[2])*(a[2]-b[2]))
00059
00060
00062 #define DOTP(a, b) (a[0]*b[0]+a[1]*b[1]+a[2]*b[2])
00063
00065 #define ERRMSG(msg)
      printf("\nError (%s, %s, l%d): %s\n\n",
00066
          __FILE__, __func__, __LINE__, msg);
exit(EXIT_FAILURE);
00067
00068
00069
00070
00072 #define EXP(x0, y0, x1, y1, x)
00073 (((y0)>0 && (y1)>0)
        ? ((y0)*exp(log((y1)/(y0))/((x1)-(x0))*((x)-(x0))))
: LIN(x0, y0, x1, y1, x))
00074
00075
00076
00078 #define LIN(x0, y0, x1, y1, x)
00079
        ((y0)+((y1)-(y0))/((x1)-(x0))*((x)-(x0))
00080
00082 #define NORM(a) sqrt(DOTP(a, a))
00083
00085 #define POW2(x) ((x)*(x))
00086
00088 #define POW3(x) ((x)*(x)*(x))
00089
00091 #define PRINT(format, var)  
00092    printf("Print (%s, %s, 1%d): %s= "format"\n",
00093
              __FILE__, __func__, __LINE__, #var, var);
00096 #define TIMER(name, mode)
00097
       {timer(name, __FILE__, __func__, __LINE__, mode);}
00098
00100 #define TOK(line, tok, format, var) {
00101         if(((tok)=strtok((line), " \t"))) {
00102         if(sscanf(tok, format, &(var))!=1) continue;
00103
          } else ERRMSG("Error while reading!");
00104 }
00105
00106 /* -----
        Constants...
00107
00108
00109
00111 #define TMIN 100.
00112
00114 #define TMAX 400.
00115
00117 #define C1 1.19104259e-8
00120 #define C2 1.43877506
00121
00123 #define G0 9.80665
00124
00126 #define KB 1.3806504e-23
00127
00129 #define NA 6.02214199e23
00130
00132 #define P0 1013.25
00133
00135 #define T0 273.15
00136
00138 #define RE 6367.421
00139
00141 #define RI 8.3144598
00142
00144 #define ME 5.976e24
00145
00146 /*
00147
00148
00149
00151 #define ND 50
00152
00154 #define NG 20
00155
00157 #define NP 1000
00158
00160 #define NR 1000
00161
```

```
00163 #define NW 5
00164
00166 #define LEN 5000
00167
00169 #define M (NR*ND)
00170
00172 #define N (NQ*NP)
00173
00175 #define NQ (2+NG+NW)
00176
00178 #define NLOS 1000
00179
00181 #define NSHAPE 10000
00182
00184 #define NFOV 5
00185
00187 #define TBLNP 41
00188
00190 #define TBLNT 30
00191
00193 #define TBLNU 320
00194
00196 #define TBLNS 1200
00197
00198 /* -
00199
       Quantity indices...
00200
00201
00203 #define IDXP 0
00204
00206 #define IDXT 1
00207
00209 #define IDXQ(ig) (2+ig)
00210
00212 #define IDXK(iw) (2+ctl->ng+iw)
00213
00214 /* -
00215
        Structs...
00216
00217
00219 typedef struct {
00220
00222
       int np;
00223
00225
       double time[NP];
00226
00228
       double z[NP];
00229
00231
       double lon[NP];
00232
00234
        double lat[NP];
00235
00237
        double p[NP];
00238
00240
       double t[NP];
00241
       double q[NG][NP];
00244
00246
        double k[NW][NP];
00247
00248 } atm_t;
00249
00251 typedef struct {
00252
00254
        int ng;
00255
00257
       char emitter[NG][LEN];
00258
00260
       int nd:
00261
00263
        int nw;
00264
00266
        double nu[ND];
00267
00269
        int window[ND];
00270
00272
        char tblbase[LEN];
00273
00275
        double hydz;
00276
00278
        int ctm_co2;
00279
00281
        int ctm_h2o;
00282
00284
       int ctm_n2;
00285
00287
        int ctm o2:
```

5.6 jurassic.h 223

```
00288
00290
        int refrac;
00291
00293
        double rayds;
00294
00296
        double raydz;
00297
00299
        char fov[LEN];
00300
00302
        double retp_zmin;
00303
00305
        double retp_zmax;
00306
00308
        double rett_zmin;
00309
00311
        double rett_zmax;
00312
00314
        double retq_zmin[NG];
00315
00317
        double retq_zmax[NG];
00318
00320
        double retk_zmin[NW];
00321
00323
        double retk_zmax[NW];
00324
00326
        int write_bbt;
00327
00329
        int write_matrix;
00330
00331 } ctl_t;
00332
00334 typedef struct {
00335
00337
        int np;
00338
        double z[NLOS];
00340
00341
00343
        double lon[NLOS];
00344
00346
        double lat[NLOS];
00347
00349
        double p[NLOS];
00350
00352
        double t[NLOS];
00353
00355
        double q[NG][NLOS];
00356
00358
        double k[NW][NLOS];
00359
00361
        double tsurf:
00362
00364
        double ds[NLOS];
00365
00367
00368
        double u[NG][NLOS];
00369 } los_t;
00370
00372 typedef struct {
00373
00375
        int nr;
00376
00378
        double time[NR];
00379
00381
        double obsz[NR];
00382
00384
        double obslon[NR];
00385
00387
        double obslat[NR];
00388
        double vpz[NR];
00391
00393
        double vplon[NR];
00394
00396
        double vplat[NR];
00397
        double tpz[NR];
00400
00402
        double tplon[NR];
00403
00405
        double tplat[NR];
00406
00408
        double tau[ND][NR];
00409
00411
        double rad[ND][NR];
00412
00413 }
        obs_t;
00414
```

```
00416 typedef struct {
00417
00419
        int np[NG][ND];
00420
        int nt[NG][ND][TBLNP];
00422
00423
        int nu[NG][ND][TBLNP][TBLNT];
00426
00428
        double p[NG][ND][TBLNP];
00429
        double t[NG][ND][TBLNP][TBLNT];
00431
00432
00434
        float u[NG][ND][TBLNP][TBLNT][TBLNU];
00435
00437
        float eps[NG][ND][TBLNP][TBLNT][TBLNU];
00438
        double st[TBLNS]:
00440
00441
00443
        double sr[ND][TBLNS];
00444
00445 } tbl_t;
00446
00447 /* -----
         Functions...
00448
00449
00450
00452 size_t atm2x(
00453 ctl_t * ctl,
00454 atm_t * atm,
        gsl_vector * x,
00455
00456
        int *iqa,
00457
        int *ipa);
00458
00460 void atm2x\_help(
        atm_t * atm,
double zmin,
00461
00462
00463
        double zmax,
00464
        double *value,
00465
        int val_iqa,
00466
        gsl_vector * x,
        int *iqa,
int *ipa,
00467
00468
00469
        size_t * n);
00470
00472 double brightness(
00473
        double rad,
00474
        double nu);
00475
00477 void cart2geo(
00478
       double *x,
00479
        double *z,
00480
        double *lon,
00481
        double *lat);
00482
00484 void climatology(
       ctl_t * ctl,
atm_t * atm_mean);
00485
00486
00487
00489 double ctmco2(
00490
        double nu,
00491
        double p,
00492
        double t,
00493
        double u);
00494
00496 double ctmh2o(
00497
        double nu,
00498
        double p,
00499
        double t.
00500
        double q,
00501
        double u);
00502
00504 double ctmn2(
00505
        double nu,
00506
        double p,
00507
        double t);
00508
00510 double ctmo2(
00511
        double nu,
00512
        double p,
00513
        double t):
00514
00516 void copy_atm(
       ctl_t * ctl,
atm_t * atm_dest,
00517
00518
        atm_t * atm_src,
00519
00520
        int init);
00521
```

5.6 jurassic.h 225

```
00523 void copy_obs(
      ctl_t * ctl,
obs_t * obs_dest,
obs_t * obs_src,
00524
00525
00526
00527
        int init);
00528
00530 int find_emitter(
00531
        ctl_t * ctl,
00532
        const char *emitter);
00533
00535 void formod(
00536 ctl_t * ctl,
00537 atm_t * atm,
00538
        obs_t * obs);
00539
00541 void formod_continua(
        ctl_t * ctl,
los_t * los,
00542
00543
00544
        int ip,
00545
        double *beta);
00546
00548 void formod_fov(
        ctl_t * ctl,
obs_t * obs);
00549
00550
00551
00553 void formod_pencil(
00554
        ctl_t * ctl,
        atm_t * atm,
obs_t * obs,
00555
00556
00557
        int ir);
00558
00560 void formod_srcfunc(
        ctl_t * ctl,
tbl_t * tbl,
00561
00562
00563
        double t,
00564
        double *src);
00565
00567 void geo2cart(
00568
        double z,
00569
         double lon,
00570
        double lat,
00571
        double *x);
00572
00574 void hydrostatic(
00575 ctl_t * ctl,
00576
        atm_t * atm);
00577
00579 void idx2name(
        ctl_t * ctl,
int idx,
00580
00581
00582
        char *quantity);
00583
00585 void init_tbl(
00586 ctl_t * ctl,
00587 tbl_t * tbl);
00588
00590 void intpol_atm(
        ctl_t * ctl,
atm_t * atm,
00591
00592
00593
         double z,
00594
        double *p,
        double *t,
00595
00596
        double *q,
00597
         double *k);
00598
00600 void intpol_tbl(
        ctl_t * ctl,
tbl_t * tbl,
00601
00602
         los_t * los,
00603
00604
         int ip,
00605
         double tau_path[NG][ND],
00606
        double tau_seg[ND]);
00607
00609 double intpol_tbl_eps(
00610
         tbl_t * tbl,
00611
         int ig,
00612
         int id,
00613
         int ip,
00614
        int it,
00615
        double u);
00616
00618 double intpol_tbl_u(
00619
        tbl_t * tbl,
00620
        int ig,
00621
        int id,
00622
        int ip,
00623
        int it,
```

```
double eps);
00625
00627 void jsec2time(
00628
        double jsec,
00629
        int *year,
00630
        int *mon.
00631
        int *day,
00632
        int *hour,
00633
        int *min,
00634
        int *sec,
        double *remain);
00635
00636
00638 void kernel(
       ctl_t * ctl,
atm_t * atm,
00639
00640
        obs_t * obs,
00641
00642
        qsl_matrix * k);
00643
00645 int locate_irr(
00646
        double *xx,
00647
        int n,
00648
        double x);
00649
00651 int locate_reg(
00652
        double *xx,
00653
        int n,
00654
        double x);
00655
00657 int locate_tbl(
        float *xx,
00658
00659
        int n.
00660
        double x);
00661
00663 size_t obs2y(
        ctl_t * ctl,
obs_t * obs,
00664
00665
00666
        gsl_vector * y,
00667
        int *ida,
00668
        int *ira);
00669
00671 double planck(
00672
        double t,
00673
        double nu);
00674
00676 void raytrace(
        ctl_t * ctl,
atm_t * atm,
00677
00678
        obs_t * obs,
los_t * los,
00679
00680
00681
        int ir);
00682
00684 void read_atm(
00685
        const char *dirname,
        const char *filename,
00686
00687
        ctl_t * ctl,
atm_t * atm);
00688
00689
00691 void read_ctl(
00692 int argc,
        char *argv[],
ctl_t * ctl);
00693
00694
00695
00697 void read_matrix(
00698 const char *dirname,
00699 const char *filename,
00700
       gsl_matrix * matrix);
00701
00703 void read_obs(
00704 const char *dirname,
00705
        const char *filename,
        ctl_t * ctl,
obs_t * obs);
00706
00707
00708
00710 void read_shape(
00711
        const char *filename,
00712
        double *x,
00713
        double *y,
00714
        int *n);
00715
00717 double refractivity(
00718
       double p,
00719
        double t);
00720
00722 double scan_ctl(
00723
       int argc,
00724
       char *argv[],
const char *varname,
00725
```

```
int arridx,
00727
        const char *defvalue,
00728
        char *value);
00729
00731 void tangent_point(
00732
       los_t * los,
double *tpz,
00734
        double *tplon,
00735
        double *tplat);
00736
00738 void time2jsec(
00739
        int year,
00740
        int mon,
00741
        int day,
00742
00743
        int min,
00744
        int sec,
00745
        double remain,
double *jsec);
00746
00747
00749 void timer(
00750
        const char *name,
00751
        const char *file,
00752
        const char *func,
00753
        int line,
00754
        int mode);
00755
00757 void write_atm(
00758
        const char *dirname,
00759
        const char *filename,
00760
        ctl_t * ctl,
00761
        atm_t * atm);
00762
00764 void write_matrix(
       const char *dirname,
const char *filename,
00765
00766
00767
        ctl t * ctl,
00768
        gsl_matrix * matrix,
00769
        atm_t * atm,
00770
        obs_t * obs,
00771
        const char *rowspace,
00772
        const char *colspace,
00773
        const char *sort);
00774
00776 void write_obs(
00777
        const char *dirname,
00778
        const char *filename,
00779
        ctl_t * ctl,
00780
       obs_t * obs);
00781
00783 void x2atm(
00784
       ctl_t * ctl,
00785
        gsl_vector * x,
00786
        atm_t * atm);
00787
00789 void x2atm_help(
00790
       atm_t * atm,
00791
        double zmin,
00792
       double zmax,
        double *value,
00793
00794
        gsl\_vector * x,
00795
        size_t * n);
00796
00798 void y2obs(
00799
00800
       gsl_vector * y,
00801
        obs_t * obs);
```

5.7 libgps.c File Reference

Functions

 void add_var (int ncid, const char *varname, const char *unit, const char *longname, int type, int dimid[], int *varid, int ndims)

Add variable to netCDF file.

void detrend met (gps t *gps, char *metbase, double dt met)

Detrending by means of meteo data.

void gauss (gps_t *gps, double dx, double dy)

Calculate horizontal Gaussian mean to extract perturbations.

void grid_gps (gps_t *gps, double zmin, double zmax, int nz)

Interpolate GPS data to regular altitude grid.

void get_met (char *metbase, double dt_met, double t, met_t *met0, met_t *met1)

Get meteorological data for given timestep.

void get met help (double t, int direct, char *metbase, double dt met, char *filename)

Get meteorological data for timestep.

• void intpol_met_3d (float array[EX][EY][EP], int ip, int ix, int iy, double wp, double wx, double wy, double *var)

Linear interpolation of 3-D meteorological data.

void intpol_met_space (met_t *met, double p, double lon, double lat, double *t)

Spatial interpolation of meteorological data.

void intpol_met_time (met_t *met0, met_t *met1, double ts, double p, double lon, double lat, double *t)

Temporal interpolation of meteorological data.

void hamming_low_pass (gps_t *gps, double dz)

Apply vertical Hamming filter to extract perturbations.

void hamming_high_pass (gps_t *gps, double dz)

Apply vertical Hamming filter to reduce noise.

• void poly (gps_t *gps, int dim, double zmin, double zmax)

Remove polynomial fit from perturbation profile.

void poly help (double *xx, double *yy, int n, int dim, double xmin, double xmax)

Auxiliary function for polynomial interpolation.

void read_gps_prof (char *filename, gps_t *gps)

Read GPS-RO profile.

void read gps (char *filename, gps t *gps)

Read GPS-RO data file.

void read_met (char *filename, met_t *met)

Read meteorological data file.

void read_met_extrapolate (met_t *met)

Extrapolate meteorological data at lower boundary.

• void read_met_help (int ncid, char *varname, char *varname2, met_t *met, float dest[EX][EY][EP], float scl)

Read and convert variable from meteorological data file.

void read_met_periodic (met_t *met)

Create meteorological data with periodic boundary conditions.

void tropopause (gps t *gps)

Find tropopause height.

void write_gps (char *filename, gps_t *gps)

Write GPS-RO data file.

5.7.1 Function Documentation

5.7.1.1 void add_var (int *ncid*, const char * *varname*, const char * *unit*, const char * *longname*, int *type*, int *dimid[]*, int * *varid*, int *ndims*)

Add variable to netCDF file.

Definition at line 5 of file libgps.c.

```
00013
                   {
00014
00015
        double dp = GSL_NAN;
00016
00017
        /* Define variable... */
00018
        NC(nc_def_var(ncid, varname, type, ndims, dimid, varid));
00019
00020
        /* Set long name... */
00021
       NC(nc_put_att_text(ncid, *varid, "long_name", strlen(longname), longname));
00022
00023
        /* Set units... */
       NC(nc_put_att_text(ncid, *varid, "units", strlen(unit), unit));
00024
00025
00026
        /* Set fill value... */
00027
        NC(nc_put_att_double(ncid, *varid, "_FillValue", type, 1, &dp));
00028 }
```

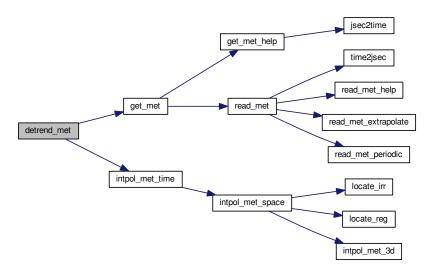
5.7.1.2 void detrend_met (gps_t * gps, char * metbase, double dt_met)

Detrending by means of meteo data.

Definition at line 32 of file libgps.c.

```
00035
                     {
00036
00037
       met_t *met0, *met1;
00038
00039
       double t;
00040
00041
       int ids, iz;
00042
00043
       /* Allocate... */
00044
       ALLOC(met0, met_t, 1);
00045
       ALLOC(met1, met_t, 1);
00046
00047
       /* Loop over profiles... */
       for (ids = 0; ids < gps->nds; ids++) {
00048
00049
00050
         /* Loop over altitudes... */
00051
         for (iz = 0; iz < gps->nz[ids]; iz++) {
00052
00053
           /\star Get meteorological data... \star/
           get_met(metbase, dt_met, gps->time[ids], met0, met1);
00054
00055
00056
           /* Interpolate meteorological data... */
           00057
00058
00059
00060
           /* Set perturbation... */
           gps->pt[ids][iz] = gps->t[ids][iz] - t;
00061
00062
00063
00064
       /* Free... */
00065
00066
       free (met0);
00067
       free (met1);
00068 }
```

Here is the call graph for this function:



5.7.1.3 void gauss (gps_t * gps, double dx, double dy)

Calculate horizontal Gaussian mean to extract perturbations.

Definition at line 72 of file libgps.c.

```
00075
00076
00077
           double dlat, dlon, w, wsum;
00078
00079
           int ids, ids2, iz;
08000
00081
           /* Loop over profiles... */
00082
           for (ids = 0; ids < gps->nds; ids++) {
00083
00084
              /* Initialize... */
             wsum = 0;
for (iz = 0; iz < gps->nz[ids]; iz++)
00085
00086
00087
                gps->pt[ids][iz] = 0;
00088
00089
              /\star Calculate lon-lat standard deviations... \star/
             /* Catculate 101-1at Standard deviations... */
dlat = dx * 180. / (M_PI * RE) / 2.3548;
dlon = dy * 180. / 2.3548
    / (M_PI * RE * cos(gps->lat[ids][gps->nz[ids] / 2] * M_PI / 180.));
00090
00091
00092
00093
00094
              /\star Calculate mean temperature... \star/
             for (ids2 = 0; ids2 < gps->nds; ids2++) {
   w = exp(-0.5 * gsl_pow_2((gps->lon[ids][gps->nz[ids] / 2]
00095
00096
                            - gps->lon[ids2][gps->nz[ids2] / 2]) / dlon)
- 0.5 * gsl_pow_2((gps->lat[ids][gps->nz[ids] / 2]
00097
00098
00099
00100
                                                      gps->lat[ids2][gps->nz[ids2] / 2]) / dlat));
                wsum += w;
for (iz = 0; iz < gps->nz[ids]; iz++)
  gps->pt[ids][iz] += w * gps->t[ids2][iz];
00101
00102
00103
00104
00105
00106
              /* Normalize... */
              if (wsum > 0)
  for (iz = 0; iz < gps->nz[ids]; iz++)
    gps->pt[ids][iz] = gps->t[ids][iz] - gps->pt[ids][iz] / wsum;
00107
00108
00109
00110
00111 }
```

5.7.1.4 void grid_gps (gps_t * gps, double zmin, double zmax, int nz)

Interpolate GPS data to regular altitude grid.

Definition at line 115 of file libgps.c.

```
00119
00120
       double lat[NZ], lon[NZ], p[NZ], pt[NZ], t[NZ], wv[NZ], z[NZ];
00122
00123
       int ids, iz, iz2;
00124
00125
       /\star Check number of altitudes... \star/
00126
       if (nz > NZ)
00127
         ERRMSG("Too many altitudes!");
00128
00129
        /* Loop over profiles... */
00130
       for (ids = 0; ids < gps->nds; ids++) {
00131
00132
          /* Loop over altitudes... */
00133
         for (iz = 0; iz < nz; iz++) {
00134
00135
           /\star Set altitude... \star/
00136
           z[iz] = LIN(0.0, zmin, nz - 1.0, zmax, (double) iz);
00137
00138
           /* Get index... */
00139
           iz2 = locate_irr(gps->z[ids], gps->nz[ids], z[iz]);
00141
           /* Interpolate... */
           00142
00143
00144
00145
00146
00147
00148
00149
           00150
00151
00152
00153
00154
00155
00156
          /* Copy data... */
00157
         gps->nz[ids] = nz;
for (iz = 0; iz < nz; iz++) {</pre>
00158
           gps \rightarrow z[ids][iz] = z[iz];
00160
           gps->lon[ids][iz] = lon[iz];
           gps->lat[ids][iz] = lat[iz];
00161
           gps->p[ids][iz] = p[iz];
gps->t[ids][iz] = t[iz];
00162
00163
00164
           gps->wv[ids][iz] = wv[iz];
           gps->pt[ids][iz] = pt[iz];
00165
00166
00167
       }
00168 }
```

Here is the call graph for this function:



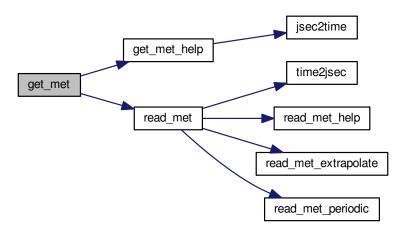
5.7.1.5 void get_met (char * metbase, double dt_met, double t, met_t * met0, met_t * met1)

Get meteorological data for given timestep.

Definition at line 172 of file libgps.c.

```
00177
00178
00179
        char filename[LEN];
00180
00181
        static int init;
00182
        /* Init... */
if (!init) {
00183
00184
00185
          init = 1;
00186
00187
           get_met_help(t, -1, metbase, dt_met, filename);
          read_met(filename, met0);
00188
00189
00190
          get_met_help(t + 1.0, 1, metbase, dt_met, filename);
00191
          read_met(filename, met1);
00192 }
00193
        /∗ Read new data... ∗/
00194
        if (t > met1->time) {
00195
         memcpy(met0, met1, sizeof(met_t));
get_met_help(t, 1, metbase, dt_met, filename);
00196
00197
00198
          read_met(filename, met1);
00199 }
00200 }
```

Here is the call graph for this function:



5.7.1.6 void get_met_help (double t, int direct, char * metbase, double dt_met, char * filename)

Get meteorological data for timestep.

Definition at line 204 of file libgps.c.

```
00210
00211
        double t6, r;
00212
00213
        int year, mon, day, hour, min, sec;
00214
00215
         /\star Round time to fixed intervals... \star/
00216
        if (direct == -1)
00217
          t6 = floor(t / dt_met) * dt_met;
00218
        else
00219
          t6 = ceil(t / dt_met) * dt_met;
00220
00221
        /* Decode time... */
00222
        jsec2time(t6, &year, &mon, &day, &hour, &min, &sec, &r);
00223
        /* Set filename... */
sprintf(filename, "%s_%d_%02d_%02d_%02d.nc", metbase, year, mon, day, hour);
00224
00225
00226 }
```

Here is the call graph for this function:



5.7.1.7 void intpol_met_3d (float array[EX][EY][EP], int ip, int ip, int iy, double wp, double wx, double wy, double * var)

Linear interpolation of 3-D meteorological data.

Definition at line 230 of file libgps.c.

```
00238
00239
00240
        double aux00, aux01, aux10, aux11;
00241
00242
         /* Interpolate vertically... */
00243
        aux00 = wp * (array[ix][iy][ip] - array[ix][iy][ip + 1])
00244
          + array[ix][iy][ip + 1];
        aux01 = wp * (array[ix][iy + 1][ip] - array[ix][iy + 1][ip + 1]) + array[ix][iy + 1][ip + 1];
00245
00246
        aux10 = wp * (array[ix + 1][iy][ip] - array[ix + 1][iy][ip + 1])
00247
00248
           + array[ix + 1][iy][ip + 1];
00249
        aux11 = wp * (array[ix + 1][iy + 1][ip] - array[ix + 1][iy + 1][ip + 1])
00250
           + array[ix + 1][iy + 1][ip + 1];
00251
00252
        /* Interpolate horizontally... */
        aux00 = wy * (aux00 - aux01) + aux01;
aux11 = wy * (aux10 - aux11) + aux11;
00253
00254
00255
         *var = wx * (aux00 - aux11) + aux11;
00256 }
```

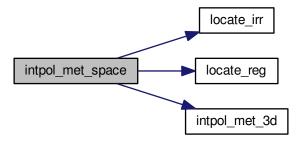
5.7.1.8 void intpol_met_space (met_t * met, double p, double lon, double lat, double * t)

Spatial interpolation of meteorological data.

Definition at line 260 of file libgps.c.

```
00265
                              {
00266
00267
            double wp, wx, wy;
00268
00269
            int ip, ix, iy;
00270
            /* Check longitude... */    if (met->lon[met->nx - 1] > 180 && lon < 0)
00271
00272
00273
              lon += 360;
00274
00275
            /* Get indices... */
           ip = locate_irr(met->p, met->np, p);
ix = locate_reg(met->lon, met->nx, lon);
iy = locate_reg(met->lat, met->ny, lat);
00276
00277
00278
00279
            /* Get weights... */
wp = (met->p[ip + 1] - p) / (met->p[ip + 1] - met->p[ip]);
wx = (met->lon[ix + 1] - lon) / (met->lon[ix + 1] - met->lon[ix]);
wy = (met->lat[iy + 1] - lat) / (met->lat[iy + 1] - met->lat[iy]);
00280
00281
00282
00283
00284
00285
             /* Interpolate... */
            intpol_met_3d(met->t, ip, ix, iy, wp, wx, wy, t);
00286
00287 }
```

Here is the call graph for this function:



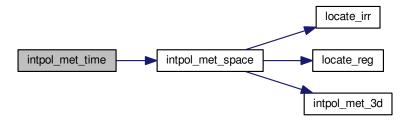
5.7.1.9 void intpol_met_time (met_t * met0, met_t * met1, double ts, double p, double lon, double lat, double * t)

Temporal interpolation of meteorological data.

Definition at line 291 of file libgps.c.

```
{
00299
00300
           double t0, t1, wt;
00301
          /* Spatial interpolation... */
intpol_met_space(met0, p, lon, lat, &t0);
intpol_met_space(met1, p, lon, lat, &t1);
00302
00303
00304
00305
          /* Get weighting factor... */ wt = (met1->time - ts) / (met1->time - met0->time);
00306
00307
00308
00309
          /* Interpolate... */
00310
          *t = wt * (t0 - t1) + t1;
00311 }
```

Here is the call graph for this function:



5.7.1.10 void hamming_low_pass ($gps_t * gps_t$ double dz)

Apply vertical Hamming filter to extract perturbations.

Definition at line 315 of file libgps.c.

```
00317
00318
00319
         double ham[NZ], wsum;
00320
00321
         int ids, iham, iz, nham;
00322
00323
         /* Loop over profiles... */
00324
         for (ids = 0; ids < gps->nds; ids++) {
00325
00326
           /* Calculate Hamming window coefficients... */
00327
           \label{eq:nham} nham = (int) (dz / fabs((gps->z[ids][0] - gps->z[ids][gps->nz[ids] - 1])
                                      / (gps->nz[ids] - 1.0)) + 0.5);
00328
           nham = GSL_MAX(GSL_MIN(nham, NZ), 2);
for (iham = 0; iham < nham; iham++)</pre>
00329
00330
00331
             ham[iham] = 0.54 + 0.46 * cos(M_PI * iham / (nham - 1.0));
00332
00333
           /* Loop over altitudes... */
00334
           for (iz = 0; iz < gps->nz[ids]; iz++) {
00335
             /* Initialize... */
gps->pt[ids][iz] = ham[0] * gps->t[ids][iz];
00336
00337
00338
             wsum = ham[0];
00339
00340
             /\star Loop over filter window... \star/
00341
             for (iham = 1; iham < nham; iham++) {</pre>
00342
00343
                /* Check array range... */
00344
               if (iz - iham < 0 || iz + iham >= gps->nz[ids])
00345
                 continue;
00346
               /* Check temperature value... */
if (!gsl_finite(gps->t[ids][iz - iham]) ||
00347
00348
00349
                    !gsl_finite(gps->t[ids][iz + iham]))
00350
                 continue;
00351
00352
                /\star Check for tropopause... \star/
00353
                if (gsl\_finite(gps->th[ids]) \&\& gps->th[ids] > 0)
                  if ((gps->z[ids][iz] >= gps->th[ids]

&& gps->z[ids][iz - iham] < gps->th[ids])
00354
00355
                       || (gps->z[ids][iz] <= gps->th[ids]
00356
00357
                           && gps->z[ids][iz + iham] > gps->th[ids]))
00358
                    continue;
00359
00360
                /\star Apply Hamming filter... \star/
               gps->pt[ids][iz]
00361
                 += ham[iham] * (gps->t[ids][iz - iham] + gps->t[ids][iz + iham]);
00362
00363
                wsum += 2 * ham[iham];
00364
00365
00366
             /* Calculate perturbation... */
             gps->pt[ids][iz] = gps->t[ids][iz] - gps->pt[ids][iz] / wsum;
00367
00368
00369
        }
00370 }
```

5.7.1.11 void hamming_high_pass (gps_t * gps, double dz)

Apply vertical Hamming filter to reduce noise.

Definition at line 374 of file libgps.c.

```
00376
                   {
00377
00378
       double ham[NZ], pt[NZ], wsum;
00380
       int ids, iham, iz, nham;
00381
00382
        /* Loop over profiles... */
00383
        for (ids = 0; ids < gps->nds; ids++) {
00384
00385
          /* Calculate Hamming window coefficients... */
         00386
00387
00388
          nham = GSL\_MAX(GSL\_MIN(nham, NZ), 2);
         for (iham = 0; iham < nham; iham++)
ham[iham] = 0.54 + 0.46 * cos(M_PI * iham / (nham - 1.0));</pre>
00389
00390
00391
00392
          /* Loop over altitudes... */
00393
          for (iz = 0; iz < gps->nz[ids]; iz++) {
00394
00395
            /* Initialize... */
00396
            pt[iz] = ham[0] * gps->pt[ids][iz];
            wsum = ham[0];
00397
00398
00399
            /* Loop over filter window... */
00400
            for (iham = 1; iham < nham; iham++) {</pre>
00401
              /* Check array range... */
00402
             if (iz - iham < 0 || iz + iham >= gps->nz[ids])
00403
               continue;
00404
00405
00406
              /* Check temperature value... */
              if (!gsl_finite(gps->t[ids][iz - iham]) ||
00407
00408
                  !gsl_finite(gps->t[ids][iz + iham]))
00409
               continue:
00410
00411
              /* Apply Hamming filter... */
             pt[iz]
00412
00413
                += ham[iham] * (gps->pt[ids][iz - iham] + gps->pt[ids][iz + iham]);
00414
              wsum += 2 * ham[iham];
00415
00416
00417
            /* Normalize... */
00418
           pt[iz] /= wsum;
00419
00420
00421
         /\star Set perturbation... \star/
         for (iz = 0; iz < gps->nz[ids]; iz++)
   gps->pt[ids][iz] = pt[iz];
00422
00424
00425 }
```

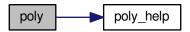
5.7.1.12 void poly (gps_t * gps, int dim, double zmin, double zmax)

Remove polynomial fit from perturbation profile.

Definition at line 429 of file libgps.c.

```
00433
00434
00435
         double bg[NZ];
00436
00437
         int ids, iz;
00438
00439
          /* Loop over profiles... ∗/
00440
         for (ids = 0; ids < gps->nds; ids++) {
00441
00442
           /* Set profile... */
for (iz = 0; iz < gps->nz[ids]; iz++)
bg[iz] = gps->pt[ids][iz];
00443
00444
00445
```

Here is the call graph for this function:



5.7.1.13 void poly_help (double *xx, double *yy, int n, int dim, double xmin, double xmax)

Auxiliary function for polynomial interpolation.

Definition at line 457 of file libgps.c.

```
00463
00464
00465
         gsl_multifit_linear_workspace *work;
00466
         gsl_matrix *cov, *X;
00467
         gsl_vector *c, *x, *y;
00468
00469
         double chisq, xx2[NZ], yy2[NZ];
00470
00471
         size_t i, i2, n2 = 0;
00472
00473
         /* Check for nan... */
00474
         for (i = 0; i < (size_t) n; i++)</pre>
          if (xx[i] \ge xmin & xx[i] \le xmax & gsl_finite(yy[i])) {
00475
00476
             xx2[n2] = xx[i];
00477
              yy2[n2] = yy[i];
00478
             n2++;
00479
00480
         if ((int) n2 < dim) {</pre>
         for (i = 0; i < (size_t) n; i++)
    yy[i] = GSL_NAN;</pre>
00481
00482
00483
           return;
00484
00485
00486
         /* Allocate...
         work = gsl_multifit_linear_alloc((size_t) n2, (size_t) dim);
00487
00488
         cov = gsl_matrix_alloc((size_t) dim, (size_t) dim);
         X = gs1_matrix_alloc((size_t) n2, (size_t) dim);
c = gs1_vector_alloc((size_t) dim);
00489
00490
00491
         x = gsl_vector_alloc((size_t) n2);
00492
         y = gsl_vector_alloc((size_t) n2);
00493
         /* Compute polynomial fit... */
for (i = 0; i < (size_t) n2; i++)</pre>
00494
00495
          gsl_vector_set(x, i, xx2[i]);
gsl_vector_set(y, i, yy2[i]);
for (i2 = 0; i2 < (size_t) dim; i2++)
00496
00497
00498
00499
              gsl_matrix_set(X, i, i2, pow(gsl_vector_get(x, i), (double) i2));
00500
         gsl_multifit_linear(X, y, c, cov, &chisq, work);
for (i = 0; i < (size_t) n; i++)</pre>
00501
00502
           yy[i] = gsl_poly_eval(c->data, (int) dim, xx[i]);
00503
00504
00505
00506
         gsl_multifit_linear_free(work);
00507
         gsl_matrix_free(cov);
00508
         gsl_matrix_free(X);
gsl_vector_free(c);
00509
         gsl_vector_free(x);
00511
         gsl_vector_free(y);
00512 }
```

```
5.7.1.14 void read_gps_prof ( char * filename, gps_t * gps )
```

Read GPS-RO profile.

Definition at line 516 of file libgps.c.

```
00518
00519
00520
          char bad[10];
00521
         double t0, t1, zmin = 1e100, zmax = -1e100;
00522
00523
00524
         int ncid, dimid, varid;
00526
          size_t iz, nz;
00527
00528
          /* Open netCDF file... */
          printf("Read GPS-RO profile: %s\n", filename);
00529
00530
          NC (nc_open(filename, NC_NOWRITE, &ncid));
          /* Get dimensions... */
NC(nc_ing_dimid(ncid, "MSL_alt", &dimid));
NC(nc_ing_dimlen(ncid, dimid, &nz));
00532
00533
00534
00535
          gps->nz[gps->nds] = (int) nz;
if (nz > NZ)
00536
00537
            ERRMSG("Too many altitudes!");
00538
00539
          /* Check data quality flag... */
          NC(nc_get_att_text(ncid, NC_GLOBAL, "bad", bad));
if (bad[0] != '0') {
00540
00541
00542
            NC(nc_close(ncid));
00543
            return;
00544
00545
          /* Get time... */
00546
          \label{eq:ncd_ncd} $$NC(nc\_get\_att\_double(ncid, NC\_GLOBAL, "start\_time", &t0)); $$NC(nc\_get\_att\_double(ncid, NC\_GLOBAL, "stop\_time", &t1)); $$gps->time[gps->nds] = 0.5 * (t0 + t1) - 630720000.0;
00547
00548
00549
00550
00551
00552
          NC(nc_inq_varid(ncid, "MSL_alt", &varid));
          NC(nc_get_var_double(ncid, varid, gps->z[gps->nds]));
NC(nc_inq_varid(ncid, "Lon", &varid));
00553
00554
          NC(nc_get_var_double(ncid, varid, gps->lon[gps->nds]));
NC(nc_inq_varid(ncid, "Lat", &varid));
NC(nc_get_var_double(ncid, varid, gps->lat[gps->nds]));
00555
00557
00558
          NC(nc_inq_varid(ncid, "Pres", &varid));
          NC(nc_get_var_double(ncid, varid, gps->p[gps->nds]));
NC(nc_inq_varid(ncid, "Temp", &varid));
00559
00560
          NC(nc_get_var_double(ncid, varid, gps->t[gps->nds]));
if (nc_ing_varid(ncid, "Vp", &varid) == NC_NOERR)
00561
00562
00563
             NC(nc_get_var_double(ncid, varid, gps->wv[gps->nds]));
00564
00565
          /* Check altitude range... */
          for (iz = 0; iz < nz; iz++)
  if (gps->p[gps->nds][iz] != -999 && gps->t[gps->nds][iz] != -999) {
00566
00567
               zmin = GSL_MIN(zmin, gps->z[gps->nds][iz]);
zmax = GSL_MAX(zmax, gps->z[gps->nds][iz]);
00568
00569
00570
00571
          if (zmin > 5 || zmax < 35) {
00572
            NC(nc_close(ncid));
00573
            return:
00574
00575
00576
          /* Check data... */
          for (iz = 0; iz < nz; iz++)
00577
00578
             if (gps \rightarrow lon[gps \rightarrow nds][iz] == -999 \mid \mid
                  gps->lat[gps->nds][iz] == -999 ||
00579
                  gps > Idt(gps > nds)[iz] == -999 ||
gps->t[gps->nds][iz] == -999 || gps->wv[gps->nds][iz] == -999) {
00580
00581
00582
                gps->lon[gps->nds][iz] = GSL_NAN;
00583
                gps->lat[gps->nds][iz] = GSL_NAN;
00584
                gps->p[gps->nds][iz] = GSL_NAN;
00585
                gps->t[gps->nds][iz] = GSL_NAN;
00586
               gps->wv[gps->nds][iz] = GSL_NAN;
00587
00588
00589
          /* Convert temperature... */
00590
          for (iz = 0; iz < nz; iz++)
00591
            gps->t[gps->nds][iz] += 273.15;
00592
00593
          /* Convert water vapor... */
00594
         for (iz = 0; iz < nz; iz++)
```

5.7.1.15 void read_gps (char * filename, gps_t * gps_)

Read GPS-RO data file.

Definition at line 607 of file libgps.c.

```
00609
00610
00611
        int ids, ncid, dimid, varid;
00612
00613
        size_t start[2], count[2], nds, nz;
00614
        /* Read netCDF file... */
printf("Read GPS-RO file: %s\n", filename);
00615
00616
00617
        NC(nc open(filename, NC NOWRITE, &ncid));
00618
00619
        /* Get dimensions... */
NC(nc_inq_dimid(ncid, "NDS", &dimid));
00620
00621
        NC(nc_inq_dimlen(ncid, dimid, &nds));
        gps->nds = (int) nds;
if (nds > NDS)
00622
00623
00624
          ERRMSG("Too many profiles!");
00625
00626
        NC(nc_inq_dimid(ncid, "NZ", &dimid));
00627
        NC(nc_inq_dimlen(ncid, dimid, &nz));
00628
        if (nz > NZ)
          ERRMSG("Too many profiles!");
00629
00630
00631
        /* Loop over profiles... ∗/
00632
        for (ids = 0; ids < gps->nds; ids++) {
00633
00634
          /* Set profile index... */
          start[0] = (size_t) ids;
count[0] = 1;
00635
00636
          start[1] = 0;
00637
00638
          count[1] = nz;
00639
00640
           /\star Set number of altitudes... \star/
00641
          gps->nz[ids] = (int) nz;
00642
00643
           /* Read data... */
00644
           NC(nc_inq_varid(ncid, "time", &varid));
00645
          NC(nc_get_vara_double(ncid, varid, start, count, &gps->time[ids]));
00646
           NC(nc_inq_varid(ncid, "z", &varid));
00647
00648
          \label{local_noise} \mbox{NC(nc\_get\_vara\_double(ncid, varid, start, count, gps->z[ids]));}
00649
           NC(nc_inq_varid(ncid, "lon", &varid));
00650
00651
          NC(nc_get_vara_double(ncid, varid, start, count, gps->lon[ids]));
00652
          NC(nc_inq_varid(ncid, "lat", &varid));
NC(nc_get_vara_double(ncid, varid, start, count, gps->lat[ids]));
00653
00654
00655
00656
           NC(nc_inq_varid(ncid, "p", &varid));
00657
           NC(nc_get_vara_double(ncid, varid, start, count, gps->p[ids]));
00658
00659
           NC(nc_inq_varid(ncid, "t", &varid));
00660
          NC(nc_get_vara_double(ncid, varid, start, count, gps->t[ids]));
00661
00662
           NC(nc_inq_varid(ncid, "wv", &varid));
00663
           NC(nc_get_vara_double(ncid, varid, start, count, gps->wv[ids]));
00664
          NC(nc_inq_varid(ncid, "pt", &varid));
00665
00666
          NC(nc_get_vara_double(ncid, varid, start, count, gps->pt[ids]));
00667
00668
           NC(nc_inq_varid(ncid, "th", &varid));
00669
          NC(nc_get_vara_double(ncid, varid, start, count, &gps->th[ids]));
00670
00671
00672
         /* Close file... */
00673
        NC(nc_close(ncid));
00674 }
```

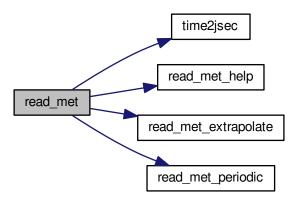
```
5.7.1.16 void read_met ( char * filename, met_t * met )
```

Read meteorological data file.

Definition at line 678 of file libgps.c.

```
00680
00681
        char tstr[10]:
00682
00683
00684
        int ip, dimid, ncid, varid, year, mon, day, hour;
00686
        size_t np, nx, ny;
00687
00688
        /* Write info... */
        printf("Read meteorological data: %s\n", filename);
00689
00690
00691
        /* Get time from filename... */
00692
        sprintf(tstr, "%.4s", &filename[strlen(filename) - 16]);
00693
        year = atoi(tstr);
         sprintf(tstr, "%.2s", &filename[strlen(filename) - 11]);
00694
        mon = atoi(tstr);
00695
00696
        sprintf(tstr, "%.2s", &filename[strlen(filename) - 8]);
00697
        day = atoi(tstr);
00698
        sprintf(tstr, "%.2s", &filename[strlen(filename) - 5]);
00699
        hour = atoi(tstr);
00700
        time2jsec(year, mon, day, hour, 0, 0, 0, &met->time);
00701
00702
         /* Open netCDF file... */
00703
        NC(nc_open(filename, NC_NOWRITE, &ncid));
00704
        /* Get dimensions... */
NC(nc_inq_dimid(ncid, "lon", &dimid));
00705
00706
00707
        NC(nc_inq_dimlen(ncid, dimid, &nx));
00708
            (nx > EX)
00709
           ERRMSG("Too many longitudes!");
00710
00711
         NC(nc_inq_dimid(ncid, "lat", &dimid));
00712
        NC(nc_inq_dimlen(ncid, dimid, &ny));
00713
        if (ny > EY)
00714
           ERRMSG("Too many latitudes!");
00715
00716
        NC(nc_inq_dimid(ncid, "lev", &dimid));
00717
        NC(nc_inq_dimlen(ncid, dimid, &np));
        if (np > EP)
00718
00719
          ERRMSG("Too many levels!");
00720
00721
        /* Store dimensions... */
        met->np = (int) np;
met->nx = (int) nx;
00722
00723
00724
        met->ny = (int) ny;
00725
        /* Get horizontal grid... */
NC(nc_inq_varid(ncid, "lon", &varid));
NC(nc_get_var_double(ncid, varid, met->lon));
NC(nc_inq_varid(ncid, "lat", &varid));
00726
00727
00728
00729
00730
        NC(nc_get_var_double(ncid, varid, met->lat));
00731
00732
        /* Read meteorological data... */
read_met_help(ncid, "t", "T", met, met->t, 1.0);
00733
00734
00735
         /* Read pressure levels from file...
00736
        NC(nc_inq_varid(ncid, "lev", &varid));
00737
        \label{local_nc_delta} \mbox{NC (nc\_get\_var\_double (ncid, varid, met->p));}
        for (ip = 0; ip < met->np; ip++)
  met->p[ip] /= 100.;
00738
00739
00740
00741
        /* Extrapolate data for lower boundary... */
00742
        read_met_extrapolate(met);
00743
00744
         /\star Check ordering of pressure levels... \star/
00745
        for (ip = 1; ip < met->np; ip++)
  if (met->p[ip - 1] < met->p[ip])
00746
00747
             ERRMSG("Pressure levels must be descending!");
00748
         /* Create periodic boundary conditions... */
00749
00750
        read_met_periodic(met);
00751
00752
         /* Close file... */
00753
        NC(nc_close(ncid));
00754 }
```

Here is the call graph for this function:



5.7.1.17 void read_met_extrapolate (met_t * met)

Extrapolate meteorological data at lower boundary.

Definition at line 758 of file libgps.c.

```
00759
00760
         int ip, ip0, ix, iy;
00763
         /* Loop over columns... */
00764
         for (ix = 0; ix < met->nx; ix++)
00765
          for (iy = 0; iy < met->ny; iy++) {
00766
             /* Find lowest valid data point... */
for (ip0 = met->np - 1; ip0 >= 0; ip0--)
00767
00768
00769
               if (!gsl_finite(met->t[ix][iy][ip0]))
00770
                 break;
00771
00772
             /* Extrapolate... */
for (ip = ip0; ip >= 0; ip--)
00773
00774
               met->t[ix][iy][ip] = met->t[ix][iy][ip + 1];
00775
00776 }
```

5.7.1.18 void read_met_help (int ncid, char * varname, char * varname2, met_t * met, float dest[EX][EY][EP], float scl)

Read and convert variable from meteorological data file.

Definition at line 780 of file libgps.c.

```
00786
00787
00788
        static float help[EX * EY * EP];
00790
       int ip, ix, iy, n = 0, varid;
00791
00792
        /\star Check if variable exists... \star/
        if (nc_inq_varid(ncid, varname, &varid) != NC_NOERR)
00793
00794
         if (nc_inq_varid(ncid, varname2, &varid) != NC_NOERR)
00795
           return;
00796
```

```
/* Read data...
00798
         NC(nc_get_var_float(ncid, varid, help));
00799
00800
          /* Copy and check data... */
          for (ip = 0; ip < met->np; ip++)
  for (iy = 0; iy < met->ny; iy++)
   for (ix = 0; ix < met->nx; ix++) {
00801
00802
00804
                 dest[ix][iy][ip] = scl * help[n++];
00805
                 if (fabs(dest[ix][iy][ip] / scl) > 1e14)
00806
                    dest[ix][iy][ip] = GSL_NAN;
               }
00807
00808 }
```

5.7.1.19 void read_met_periodic (met_t * met)

Create meteorological data with periodic boundary conditions.

Definition at line 812 of file libgps.c.

```
00813
00814
00815
        int ip, iy;
00816
        /* Check longitudes... */
00817
        if (!(fabs(met->lon[met->nx - 1] - met->lon[0]
                     + met->lon[1] - met->lon[0] - 360) < 0.01))
00819
00820
00821
        /\star Increase longitude counter... \star/
00822
00823
        if ((++met->nx) > EX)
00824
          ERRMSG("Cannot create periodic boundary conditions!");
00825
        /* Set longitude... */
00826
00827
        met \rightarrow lon[met \rightarrow nx - 1] = met \rightarrow lon[met \rightarrow nx - 2] + met \rightarrow lon[1] - met \rightarrow
      lon[0];
00828
00829
         /* Loop over latitudes and pressure levels... */
        for (iy = 0; iy < met->ny; iy++)
00831
          for (ip = 0; ip < met->np; ip++)
00832
             met->t[met->nx - 1][iy][ip] = met->t[0][iy][ip];
00833 }
```

5.7.1.20 void tropopause (gps_t * gps)

Find tropopause height.

Definition at line 837 of file libgps.c.

```
00838
00839
00840
         double zmin;
00841
        int ids, iz, iz2, okay;
00842
00843
        /* Loop over profiles... */
00844
00845
         for (ids = 0; ids < gps->nds; ids++) {
00846
00847
           /* Set default value... */
00848
           gps->th[ids] = GSL_NAN;
00849
00850
           /* Set minimum altitude... */
00851
           zmin =
00852
            8 - 4 * fabs(cos((90 - gps->lat[ids][gps->nz[ids] / 2]) * M_PI / 180));
00853
00854
           /\star Search tropopause (WMO definition)... \star/
00855
           for (iz = 0; iz < gps->nz[ids]; iz++)
  if (gps->z[ids][iz] >= zmin && gps->z[ids][iz] <= 20.0) {</pre>
00856
               okay = 1;
00858
                for (iz2 = iz + 1; iz2 < gps->nz[ids]; iz2++)
00859
                  if (gps->z[ids][iz2] - gps->z[ids][iz] <= 2.0)</pre>
                    if (!gsl_finite(gps->t[ids][iz]) ||
00860
                         !gsl_finite(gps->t[ids][iz2]) ||
00861
                        (gps->t[ids][iz2] - gps->t[ids][iz])
/ (gps->z[ids][iz2] - gps->z[ids][iz]) < -2.0)</pre>
00862
00863
00864
                      okay = 0;
               if (okay) {
00865
00866
                  gps->th[ids] = gps->z[ids][iz];
00867
                 break:
00868
00869
00870 }
00871 }
```

```
5.7.1.21 void write_gps ( char * filename, gps_t * gps )
```

Write GPS-RO data file.

Definition at line 875 of file libgps.c.

```
00877
00878
00879
          static double help[NDS * NZ];
00880
          int ids, iz, ncid, dimid[2], time_id, z_id, lon_id, lat_id, p_id, t_id,
00881
            pt_id, wv_id, th_id, nzmax = 0;
00883
00884
          /* Create netCDF file...
          printf("Write GPS-RO file: %s\n", filename);
00885
          NC(nc_create(filename, NC_CLOBBER, &ncid));
00886
00887
00888
          /* Set dimensions... */
NC(nc_def_dim(ncid, "NDS", (size_t) gps->nds, &dimid[0]));
00889
00890
          for (ids = 0; ids < gps->nds; ids++)
          nzmax = GSL_MAX(nzmax, gps->nz[ids]);
NC(nc_def_dim(ncid, "NZ", (size_t) nzmax, &dimid[1]));
00891
00892
00893
          /* Add variables... */
add_var(ncid, "time", "s", "time (seconds since 2000-01-01T00:00Z)",
00894
00895
         add_var(ncid, "time", "s", "time (seconds since 2000-01-01T00:00Z)",

NC_DOUBLE, dimid, &time_id, 1);

add_var(ncid, "z", "km", "altitude", NC_FLOAT, dimid, &z_id, 2);

add_var(ncid, "lon", "deg", "longitude", NC_FLOAT, dimid, &lon_id, 2);

add_var(ncid, "lat", "deg", "latitude", NC_FLOAT, dimid, &lat_id, 2);

add_var(ncid, "p", "hPa", "pressure", NC_FLOAT, dimid, &p_id, 2);

add_var(ncid, "t", "K", "temperature", NC_FLOAT, dimid, &t_id, 2);

add_var(ncid, "wv", "ppm", "water vapor volume mixing ratio",
00896
00897
00898
00899
00900
00901
00902
         00903
00904
00905
00906
00907
00908
           /* Leave define mode... */
00909
          NC(nc_enddef(ncid));
00910
00911
          /* Write data... */
00912
          NC(nc_put_var_double(ncid, time_id, gps->time));
          NC(nc_put_var_double(ncid, th_id, gps->th));
00913
00914
          for (ids = 0; ids < gps->nds; ids++)
            for (iz = 0; iz < gps->nz[ids]; iz++)
00915
00916
               help[ids * nzmax + iz] = gps->z[ids][iz];
          NC(nc_put_var_double(ncid, z_id, help));
for (ids = 0; ids < gps->nds; ids++)
00917
00918
            for (iz = 0; iz < gps->nz[ids]; iz++)
help[ids * nzmax + iz] = gps->lon[ids][iz];
00919
00920
00921
          NC(nc_put_var_double(ncid, lon_id, help));
00922
          for (ids = 0; ids < gps->nds; ids++)
            for (iz = 0; iz < gps->nz[ids]; iz++)
00923
               help[ids * nzmax + iz] = gps->lat[ids][iz];
00924
          NC(nc_put_var_double(ncid, lat_id, help));
00925
          for (ids = 0; ids < gps->nds; ids++)
            for (iz = 0; iz < gps->nz[ids]; iz++)
00927
00928
               help[ids * nzmax + iz] = gps->p[ids][iz];
          NC(nc_put_var_double(ncid, p_id, help));
00929
          for (ids = 0; ids < gps->nds; ids++)
  for (iz = 0; iz < gps->nz[ids]; iz++)
00930
00931
               help[ids * nzmax + iz] = gps->t[ids][iz];
00932
00933
          NC(nc_put_var_double(ncid, t_id, help));
          for (ids = 0; ids < gps->nds; ids++)
  for (iz = 0; iz < gps->nz[ids]; iz++)
00934
00935
               help[ids * nzmax + iz] = gps->wv[ids][iz];
00936
          NC(nc_put_var_double(ncid, wv_id, help));
for (ids = 0; ids < gps->nds; ids++)
00937
00938
00939
           for (iz = 0; iz < gps->nz[ids]; iz++)
00940
               help[ids * nzmax + iz] = gps->pt[ids][iz];
00941
          NC(nc_put_var_double(ncid, pt_id, help));
00942
00943
          /* Close file... */
00944
          NC(nc_close(ncid));
00945 }
```

Here is the call graph for this function:



5.8 libgps.c

```
00001 #include "libgps.h"
00002
00004
00005 void add_var(
00006
       int ncid,
00007
       const char *varname,
80000
       const char *unit,
00009
       const char *longname,
00010
       int type,
00011
       int dimid[],
00012
       int *varid,
00013
       int ndims) {
00014
00015
       double dp = GSL_NAN;
00016
       /* Define variable... */
00017
00018
       NC(nc_def_var(ncid, varname, type, ndims, dimid, varid));
00019
00020
        /* Set long name... */
00021
       NC(nc_put_att_text(ncid, *varid, "long_name", strlen(longname), longname));
00022
00023
       /* Set units... */
00024
       NC(nc_put_att_text(ncid, *varid, "units", strlen(unit), unit));
00025
00026
        /* Set fill value... */
00027
       NC(nc_put_att_double(ncid, *varid, "_FillValue", type, 1, &dp));
00028 }
00029
00031
00032 void detrend_met(
       gps_t * gps,
char *metbase,
00033
00034
00035
       double dt_met)
00036
00037
       met_t *met0, *met1;
00038
00039
       double t;
00040
00041
       int ids, iz;
00042
00043
       /* Allocate... */
00044
       ALLOC(met0, met_t, 1);
00045
       ALLOC(met1, met_t, 1);
00046
       /* Loop over profiles... */
for (ids = 0; ids < gps->nds; ids++) {
00047
00048
00049
00050
         /* Loop over altitudes... */
00051
         for (iz = 0; iz < gps->nz[ids]; iz++) {
00052
00053
           /\star Get meteorological data... \star/
           get_met(metbase, dt_met, gps->time[ids], met0, met1);
00054
00055
00056
           /* Interpolate meteorological data... */
00057
           intpol_met_time(met0, met1, gps->time[ids], gps->p[ids][iz],
00058
                          gps->lon[ids][iz], gps->lat[ids][iz], &t);
00059
00060
           /* Set perturbation... */
           gps->pt[ids][iz] = gps->t[ids][iz] - t;
00061
00062
00063
```

5.8 libgps.c 245

```
00064
00065
        /* Free... */
00066
        free (met0);
00067
       free (met1);
00068 }
00069
00071
00072 void gauss(
       gps_t * gps,
double dx,
00073
00074
00075
       double dy)
00076
00077
       double dlat, dlon, w, wsum;
00078
00079
       int ids, ids2, iz;
08000
        /* Loop over profiles... */
for (ids = 0; ids < gps->nds; ids++) {
00081
00082
00083
00084
          /* Initialize... */
          wsum = 0;
for (iz = 0; iz < gps->nz[ids]; iz++)
00085
00086
            gps \rightarrow pt[ids][iz] = 0;
00087
00088
00089
          /* Calculate lon-lat standard deviations... */
          dlat = dx * 180. / (M_PI * RE) / 2.3548;
dlon = dy * 180. / 2.3548
00090
00091
            / (M_PI * RE * cos(gps->lat[ids][gps->nz[ids] / 2] * M_PI / 180.));
00092
00093
00094
          /* Calculate mean temperature... */
for (ids2 = 0; ids2 < gps->nds; ids2++)
00095
00096
            w = \exp(-0.5 * gsl_pow_2((gps->lon[ids][gps->nz[ids] / 2])
                                        gps->lon[ids2][gps->nz[ids2] / 2]) / dlon)
00097
00098
                    - 0.5 * gsl_pow_2((gps->lat[ids][gps->nz[ids] / 2]
00099
00100
                                        gps->lat[ids2][gps->nz[ids2] / 2]) / dlat));
00101
            wsum += w;
00102
            for (iz = 0; iz < gps->nz[ids]; iz++)
00103
             gps->pt[ids][iz] += w * gps->t[ids2][iz];
00104
00105
00106
          /* Normalize... */
00107
          if (wsum > 0)
            for (iz = 0; iz < gps->nz[ids]; iz++)
00108
00109
              gps->pt[ids][iz] = gps->t[ids][iz] - gps->pt[ids][iz] / wsum;
00110
00111 }
00112
00113 /
00114
00115 void grid_gps(
00116
        gps_t * gps,
00117
        double zmin,
00118
        double zmax,
00119
       int nz) {
00121
       double lat[NZ], lon[NZ], p[NZ], pt[NZ], t[NZ], wv[NZ], z[NZ];
00122
00123
        int ids, iz, iz2;
00124
00125
        /* Check number of altitudes... */
00126
        if (nz > NZ)
         ERRMSG("Too many altitudes!");
00127
00128
00129
        /* Loop over profiles... */
00130
        for (ids = 0; ids < gps->nds; ids++) {
00131
00132
          /* Loop over altitudes... */
          for (iz = 0; iz < nz; iz++) {
00133
00134
00135
            /* Set altitude... */
00136
            z[iz] = LIN(0.0, zmin, nz - 1.0, zmax, (double) iz);
00137
            /* Get index... */
iz2 = locate_irr(gps->z[ids], gps->nz[ids], z[iz]);
00138
00139
00140
00141
            /* Interpolate... */
            00142
00143
00144
00145
            00146
00147
             \begin{split} t[iz] &= LIN(gps - z[ids][iz2], \ gps - t[ids][iz2], \\ &= gps - z[ids][iz2 + 1], \ gps - t[ids][iz2 + 1], \ z[iz]); \end{split} 
00148
00149
            wv[iz] = LIN(gps->z[ids][iz2], gps->wv[ids][iz2],
00150
```

```
gps->z[ids][iz2 + 1], gps->wv[ids][iz2 + 1], z[iz]);
          pt[iz] = LIN(gps->z[ids][iz2], gps->pt[ids][iz2], gps->z[ids][iz2 + 1], z[iz]);
00152
00153
00154
         }
00155
         /* Copy data... */
gps->nz[ids] = nz;
00156
00157
00158
         for (iz = 0; iz < nz; iz++) {
00159
           gps->z[ids][iz] = z[iz];
00160
           gps->lon[ids][iz] = lon[iz];
           gps->lat[ids][iz] = lat[iz];
00161
00162
           gps->p[ids][iz] = p[iz];
gps->t[ids][iz] = t[iz];
00163
00164
           gps->wv[ids][iz] = wv[iz];
00165
           gps->pt[ids][iz] = pt[iz];
00166
       }
00167
00168 }
00169
00170 /
       ******************************
00171
00172 void get_met(
00173
       char *metbase,
       double dt_met,
00174
00175
       double t,
00176
       met_t * met0,
00177
       met_t * met1) {
00178
00179
       char filename[LEN];
00180
00181
       static int init:
00182
00183
       /* Init... */
00184
       if (!init) {
00185
         init = 1;
00186
00187
         get_met_help(t, -1, metbase, dt_met, filename);
00188
         read_met(filename, met0);
00189
00190
         get_met_help(t + 1.0, 1, metbase, dt_met, filename);
00191
         read_met(filename, met1);
       1
00192
00193
00194
       /* Read new data... */
00195
       if (t > met1->time) {
00196
         memcpy(met0, met1, sizeof(met_t));
00197
         get_met_help(t, 1, metbase, dt_met, filename);
00198
         read_met(filename, met1);
00199
00200 }
00201
00203
00204 void get_met_help(
00205
      double t.
00206
       int direct,
00207
       char *metbase,
00208
       double dt_met,
00209
       char *filename) {
00210
00211
       double t6, r;
00212
00213
       int year, mon, day, hour, min, sec;
00214
00215
       /* Round time to fixed intervals... */
00216
       if (direct == -1)
        t6 = floor(t / dt_met) * dt_met;
00217
00218
       else
00219
        t6 = ceil(t / dt_met) * dt_met;
00220
       /* Decode time... */
00221
00222
       jsec2time(t6, &year, &mon, &day, &hour, &min, &sec, &r);
00223
       /* Set filename... */
sprintf(filename, "%s_%d_%02d_%02d_nc", metbase, year, mon, day, hour);
00224
00225
00226 }
00227
00229
00230 void intpol met 3d(
00231
       float array[EX][EY][EP],
00232
       int ip,
00233
       int ix,
00234
       int iy,
00235
       double wp,
00236
       double wx,
00237
       double wy,
```

5.8 libgps.c 247

```
double *var) {
00239
00240
       double aux00, aux01, aux10, aux11;
00241
       /* Interpolate vertically... */
aux00 = wp * (array[ix][iy][ip] - array[ix][iy][ip + 1])
+ array[ix][iy][ip + 1];
00242
00243
00245
       aux01 = wp * (array[ix][iy + 1][ip] - array[ix][iy + 1][ip + 1])
       00246
00247
00248
00249
00250
         + array[ix + 1][iy + 1][ip + 1];
00251
00252
       /* Interpolate horizontally... */
       aux00 = wy * (aux00 - aux01) + aux01;

aux11 = wy * (aux10 - aux11) + aux11;
00253
00254
00255
       *var = wx * (aux00 - aux11) + aux11;
00256 }
00257
00259
00260 void intpol_met_space(
00261
       met t * met,
       double p,
00262
       double lon,
00263
00264
       double lat,
00265
       double *t) {
00266
00267
       double wp, wx, wy;
00268
00269
       int ip, ix, iv;
00270
00271
       /\star Check longitude... \star/
       if (met->lon[met->nx - 1] > 180 && lon < 0)
00272
00273
         lon += 360;
00274
       /* Get indices... */
00276
       ip = locate_irr(met->p, met->np, p);
00277
       ix = locate_reg(met->lon, met->nx, lon);
00278
       iy = locate_reg(met->lat, met->ny, lat);
00279
00280
       /* Get weights... */
       wp = (met->p[ip + 1] - p) / (met->p[ip + 1] - met->p[ip]);
wx = (met->lon[ix + 1] - lon) / (met->lon[ix + 1] - met->lon[ix]);
wy = (met->lat[iy + 1] - lat) / (met->lat[iy + 1] - met->lat[iy]);
00281
00283
00284
       /* Interpolate... */
intpol_met_3d(met->t, ip, ix, iy, wp, wx, wy, t);
00285
00286
00287 }
00288
00290
00291 void intpol_met_time(
00292
       met_t * met0,
met_t * met1,
00293
00294
       double ts,
00295
       double p,
00296
       double lon,
       double lat,
00297
00298
       double *t) {
00299
00300
       double t0, t1, wt;
00301
00302
       /* Spatial interpolation... */
00303
       intpol_met_space(met0, p, lon, lat, &t0);
00304
       intpol_met_space(met1, p, lon, lat, &t1);
00305
00306
       /* Get weighting factor... */
       wt = (met1->time - ts) / (met1->time - met0->time);
00307
00308
00309
       /* Interpolate... */
00310
       *t = wt * (t0 - t1) + t1;
00311 }
00312
00314
00315 void hamming_low_pass(
00316
       gps_t * gps,
       double dz) {
00317
00318
00319
       double ham[NZ], wsum;
00320
00321
       int ids, iham, iz, nham;
00322
00323
       /* Loop over profiles... */
00324
       for (ids = 0; ids < gps->nds; ids++) {
```

```
00325
00326
           /* Calculate Hamming window coefficients... */
          00327
00328
          nham = GSL_MAX(GSL_MIN(nham, NZ), 2);
for (iham = 0; iham < nham; iham++)</pre>
00329
00330
            ham[iham] = 0.54 + 0.46 * cos(M_PI * iham / (nham - 1.0));
00331
00332
00333
           /* Loop over altitudes... */
00334
          for (iz = 0; iz < gps->nz[ids]; iz++) {
00335
00336
            /* Initialize... */
gps->pt[ids][iz] = ham[0] * gps->t[ids][iz];
00337
00338
             wsum = ham[0];
00339
            /* Loop over filter window... */
for (iham = 1; iham < nham; iham++) {</pre>
00340
00341
00342
               /* Check array range... */
00343
00344
               if (iz - iham < 0 \mid \mid iz + iham >= gps->nz[ids])
00345
                continue;
00346
              /* Check temperature value... */
if (!gsl_finite(gps->t[ids][iz - iham]) ||
    !gsl_finite(gps->t[ids][iz + iham]))
00347
00348
00349
00350
                 continue;
00351
00352
               /* Check for tropopause... */
               00353
00354
00355
00356
00357
                          && gps->z[ids][iz + iham] > gps->th[ids]))
00358
                   continue;
00359
               /* Apply Hamming filter... */
00360
00361
              gps->pt[ids][iz]
                += ham[iham] * (gps->t[ids][iz - iham] + gps->t[ids][iz + iham]);
00362
00363
               wsum += 2 * ham[iham];
00364
00365
00366
             /\star Calculate perturbation... \star/
            gps->pt[ids][iz] = gps->t[ids][iz] - gps->pt[ids][iz] / wsum;
00367
00368
00369
00370 }
00371
00372
        ******************************
00373
00374 void hamming_high_pass(
00375
        gps_t * gps,
00376
        double dz) {
00377
00378
        double ham[NZ], pt[NZ], wsum;
00379
00380
        int ids, iham, iz, nham;
00382
        /* Loop over profiles... */
00383
        for (ids = 0; ids < gps->nds; ids++) {
00384
          /* Calculate Hamming window coefficients... */
00385
          \label{eq:nham} nham = (int) (dz / fabs((gps->z[ids][0] - gps->z[ids][gps->nz[ids] - 1])
00386
00387
                                    / (gps->nz[ids] - 1.0)) + 0.5);
00388
          nham = GSL_MAX(GSL_MIN(nham, NZ), 2);
00389
           for (iham = 0; iham < nham; iham++)</pre>
            ham[iham] = 0.54 + 0.46 * cos(M_PI * iham / (nham - 1.0));
00390
00391
          /* Loop over altitudes... */
for (iz = 0; iz < gps->nz[ids]; iz++) {
00392
00393
00394
            /* Initialize... */
pt[iz] = ham[0] * gps->pt[ids][iz];
00395
00396
             wsum = ham[0];
00397
00398
             /* Loop over filter window... ∗/
00399
00400
             for (iham = 1; iham < nham; iham++) {</pre>
00401
               /* Check array range... */
if (iz - iham < 0 || iz + iham >= gps->nz[ids])
00402
00403
00404
                continue:
00405
00406
               /* Check temperature value... */
00407
               if (!gsl_finite(gps->t[ids][iz - iham]) ||
                   !gsl_finite(gps->t[ids][iz + iham]))
00408
00409
                 continue;
00410
00411
              /* Apply Hamming filter... */
```

5.8 libgps.c 249

```
00412
             pt[iz]
00413
                += ham[iham] * (gps->pt[ids][iz - iham] + gps->pt[ids][iz + iham]);
00414
              wsum += 2 * ham[iham];
00415
           }
00416
            /* Normalize... */
00417
           pt[iz] /= wsum;
00418
00419
00420
00421
          /* \ {\tt Set \ perturbation...} \ */
         for (iz = 0; iz < gps->nz[ids]; iz++)
00422
00423
           gps->pt[ids][iz] = pt[iz];
00424
00425 }
00426
00428
00429 void poly(
00430 gps_t * gps,
        int dim,
00431
00432
        double zmin,
00433
       double zmax)
00434
00435
       double bg[NZ];
00436
00437
       int ids, iz;
00438
00439
        /* Loop over profiles... */
00440
        for (ids = 0; ids < gps->nds; ids++) {
00441
00442
         /* Set profile... */
for (iz = 0; iz < gps->nz[ids]; iz++)
bg[iz] = gps->pt[ids][iz];
00443
00444
00445
00446
          /\star Polynomial interpolation... \star/
         poly_help(gps->z[ids], bg, gps->nz[ids], dim, zmin, zmax);
00447
00448
          /* Remove background... */
00450
          for (iz = 0; iz < gps->nz[ids]; iz++)
00451
           gps->pt[ids][iz] -= bg[iz];
00452
00453 }
00454
00457 void poly_help(
00458
       double *xx,
00459
       double *yy,
00460
        int n.
00461
       int dim.
00462
       double xmin,
00463
       double xmax) {
00464
00465
       gsl_multifit_linear_workspace *work;
       gsl_matrix *cov, *X;
gsl_vector *c, *x, *y;
00466
00467
00468
00469
       double chisq, xx2[NZ], yy2[NZ];
00470
00471
       size_t i, i2, n2 = 0;
00472
00473
        /* Check for nan... */
00474
        for (i = 0; i < (size_t) n; i++)</pre>
00475
        if (xx[i] >= xmin && xx[i] <= xmax && gsl_finite(yy[i])) {
00476
            xx2[n2] = xx[i];
00477
            yy2[n2] = yy[i];
00478
           n2++;
00479
00480
        if ((int) n2 < dim) {</pre>
        for (i = 0; i < (size_t) n; i++)
00481
00482
           yy[i] = GSL_NAN;
00483
         return;
00484
       }
00485
00486
       /* Allocate... */
        work = gsl_multifit_linear_alloc((size_t) n2, (size_t) dim);
00488
        cov = gsl_matrix_alloc((size_t) dim, (size_t) dim);
00489
        X = gsl_matrix_alloc((size_t) n2, (size_t) dim);
00490
       c = gsl_vector_alloc((size_t) dim);
       x = gsl_vector_alloc((size_t) n2);
00491
00492
       y = gsl_vector_alloc((size_t) n2);
00493
        /* Compute polynomial fit... */
for (i = 0; i < (size_t) n2; i++)
00494
00495
00496
         gsl_vector_set(x, i, xx2[i]);
          gsl_vector_set(y, i, yy2[i]);
for (i2 = 0; i2 < (size_t) dim; i2++)
00497
00498
```

```
gsl_matrix_set(X, i, i2, pow(gsl_vector_get(x, i), (double) i2));
00500
00501
         gsl_multifit_linear(X, y, c, cov, &chisq, work);
         for (i = 0; i < (size_t) n; i++)
  yy[i] = gsl_poly_eval(c->data, (int) dim, xx[i]);
00502
00503
00504
00506
         gsl_multifit_linear_free(work);
00507
         gsl_matrix_free(cov);
00508
          gsl_matrix_free(X);
00509
         gsl_vector_free(c);
00510
         gsl vector free(x);
00511
         qsl vector free(v);
00512 }
00513
00515
00516 void read gps prof(
        char *filename,
00518
         gps_t * gps) {
00519
00520
         char bad[10];
00521
         double t0, t1, zmin = 1e100, zmax = -1e100;
00522
00523
00524
         int ncid, dimid, varid;
00525
00526
         size_t iz, nz;
00527
00528
         /* Open netCDF file... */
00529
         printf("Read GPS-RO profile: %s\n", filename);
00530
         NC(nc_open(filename, NC_NOWRITE, &ncid));
00531
          /* Get dimensions... */
00532
         NC(nc_inq_dimid(ncid, "MSL_alt", &dimid));
NC(nc_inq_dimlen(ncid, dimid, &nz));
00533
00534
         gps->nz[gps->nds] = (int) nz;
00535
         if (nz > NZ)
00537
            ERRMSG("Too many altitudes!");
00538
00539
          /* Check data quality flag... */
         NC(nc_get_att_text(ncid, NC_GLOBAL, "bad", bad));
if (bad[0] != '0') {
00540
00541
00542
           NC(nc_close(ncid));
00543
           return;
00544
00545
00546
          /* Get time... */
         NC(nc_get_att_double(ncid, NC_GLOBAL, "start_time", &t0));
NC(nc_get_att_double(ncid, NC_GLOBAL, "stop_time", &t1));
00547
00548
         qps->time[qps->nds] = 0.5 * (t0 + t1) - 630720000.0;
00549
00550
00551
          /* Get data... */
00552
         NC(nc_inq_varid(ncid, "MSL_alt", &varid));
         NC(nc_get_var_double(ncid, varid, gps->z[gps->nds]));
NC(nc_inq_varid(ncid, "Lon", &varid));
00553
00554
         NC(nc_inq_varid(ncid, "son", &varid));
NC(nc_get_var_double(ncid, varid, gps->lon[gps->nds]));
NC(nc_inq_varid(ncid, "Lat", &varid));
NC(nc_get_var_double(ncid, varid, gps->lat[gps->nds]));
NC(nc_inq_varid(ncid, "Pres", &varid));
00556
00557
00558
         NC(nc_get_var_double(ncid, varid, gps->p[gps->nds]));
00559
         NC(nc_inq_varid(ncid, "Temp", &varid));
00560
         NC(nc_get_var_double(ncid, varid, gps->t[gps->nds]));
if (nc_inq_varid(ncid, "Vp", &varid) == NC_NOERR)
    NC(nc_get_var_double(ncid, varid, gps->wv[gps->nds]));
00561
00562
00563
00564
00565
          /* Check altitude range...
         for (iz = 0; iz < nz; iz++)
  if (gps->p[gps->nds][iz] != -999 && gps->t[gps->nds][iz] != -999) {
00566
00567
00568
              zmin = GSL_MIN(zmin, gps->z[gps->nds][iz]);
00569
              zmax = GSL_MAX(zmax, gps->z[gps->nds][iz]);
00570
00571
         if (zmin > 5 || zmax < 35) {</pre>
00572
           NC(nc_close(ncid));
00573
           return;
00574
00575
         /* Check data... */
for (iz = 0; iz < nz; iz++)
00576
00577
00578
            if (gps->lon[gps->nds][iz] == -999 ||
00579
                 gps->lat[gps->nds][iz] == -999 ||
                 ggb > Idt(ggps > Idd) [iz] == -999 ||
gps->t[gps->nds][iz] == -999 || gps->wv[gps->nds][iz] == -999) {
00580
00581
              gps->lon[gps->nds][iz] = GSL_NAN;
00582
00583
              gps->lat[gps->nds][iz] = GSL_NAN;
              gps->p[gps->nds][iz] = GSL_NAN;
gps->t[gps->nds][iz] = GSL_NAN;
00584
00585
```

5.8 libgps.c 251

```
gps->wv[gps->nds][iz] = GSL_NAN;
00587
00588
00589
        /\star Convert temperature... \star/
00590
        for (iz = 0; iz < nz; iz++)
00591
          qps->t[qps->nds][iz] += 273.15;
00592
00593
        /* Convert water vapor...
00594
        for (iz = 0; iz < nz; iz++)
          gps->wv[gps->nds][iz] *= 1e6 / gps->p[gps->nds][iz];
00595
00596
00597
        /* Close file... */
00598
        NC(nc_close(ncid));
00599
00600
        /* Count profiles... */
        if ((++gps->nds) >= NDS)
    ERRMSG("Too many profiles!");
00601
00602
00603 }
00604
00606
00607 void read_gps(
00608
        char *filename,
00609
        gps_t * gps) {
00610
00611
        int ids, ncid, dimid, varid;
00612
00613
        size_t start[2], count[2], nds, nz;
00614
00615
        /* Read netCDF file... */
        printf("Read GPS-RO file: %s\n", filename);
00616
00617
        NC(nc_open(filename, NC_NOWRITE, &ncid));
00618
        /* Get dimensions... */
NC(nc_inq_dimid(ncid, "NDS", &dimid));
NC(nc_inq_dimlen(ncid, dimid, &nds));
00619
00620
00621
        gps->nds = (int) nds;
if (nds > NDS)
00622
00623
00624
          ERRMSG("Too many profiles!");
00625
00626
        NC(nc_inq_dimid(ncid, "NZ", &dimid));
        NC(nc_inq_dimlen(ncid, dimid, &nz));
00627
00628
        if (nz > NZ)
00629
          ERRMSG("Too many profiles!");
00630
00631
        /* Loop over profiles... ∗/
00632
        for (ids = 0; ids < gps->nds; ids++) {
00633
00634
          /* Set profile index... */
          start[0] = (size_t) ids;
count[0] = 1;
00635
00636
00637
          start[1] = 0;
00638
          count[1] = nz;
00639
00640
          /* Set number of altitudes... */
          gps->nz[ids] = (int) nz;
00641
00642
00643
           /* Read data...
00644
          NC(nc_inq_varid(ncid, "time", &varid));
00645
          NC(nc_get_vara_double(ncid, varid, start, count, &gps->time[ids]));
00646
          NC(nc_inq_varid(ncid, "z", &varid));
00647
00648
          NC(nc_get_vara_double(ncid, varid, start, count, gps->z[ids]));
00649
00650
          NC(nc_inq_varid(ncid, "lon", &varid));
00651
          NC(nc_get_vara_double(ncid, varid, start, count, gps->lon[ids]));
00652
          NC(nc_inq_varid(ncid, "lat", &varid));
00653
          NC(nc_get_vara_double(ncid, varid, start, count, gps->lat[ids]));
00654
00655
00656
          NC(nc_inq_varid(ncid, "p", &varid));
00657
          NC(nc_get_vara_double(ncid, varid, start, count, gps->p[ids]));
00658
          NC(nc_inq_varid(ncid, "t", &varid));
00659
00660
          NC(nc_get_vara_double(ncid, varid, start, count, gps->t[ids]));
00661
00662
          NC(nc_inq_varid(ncid, "wv", &varid));
00663
          NC(nc_get_vara_double(ncid, varid, start, count, gps->wv[ids]));
00664
          NC(nc_inq_varid(ncid, "pt", &varid));
00665
00666
          NC(nc_get_vara_double(ncid, varid, start, count, gps->pt[ids]));
00667
          NC(nc_inq_varid(ncid, "th", &varid));
NC(nc_get_vara_double(ncid, varid, start, count, &gps->th[ids]));
00668
00669
00670
00671
00672
        /* Close file... */
```

```
NC(nc_close(ncid));
00674 }
00675
00677
00678 void read_met(
00679
      char *filename,
00680
       met_t * met) {
00681
00682
       char tstr[10];
00683
       int ip, dimid, ncid, varid, year, mon, day, hour;
00684
00685
00686
       size_t np, nx, ny;
00687
00688
       /* Write info... */
        printf("Read meteorological data: %s\n", filename);
00689
00690
00691
        /* Get time from filename... */
00692
        sprintf(tstr, "%.4s", &filename[strlen(filename) - 16]);
00693
        year = atoi(tstr);
00694
        sprintf(tstr, "%.2s", &filename[strlen(filename) - 11]);
00695
        mon = atoi(tstr);
        sprintf(tstr, "%.2s", &filename[strlen(filename) - 8]);
00696
00697
        day = atoi(tstr);
        sprintf(tstr, "%.2s", &filename[strlen(filename) - 5]);
00698
00699
        hour = atoi(tstr);
00700
        time2jsec(year, mon, day, hour, 0, 0, 0, &met->time);
00701
00702
        /* Open netCDF file... */
        NC(nc_open(filename, NC_NOWRITE, &ncid));
00703
00704
        /* Get dimensions... */
NC(nc_inq_dimid(ncid, "lon", &dimid));
00705
00706
00707
        NC(nc_inq_dimlen(ncid, dimid, &nx));
00708
        if (nx > EX)
00709
          ERRMSG("Too many longitudes!");
00710
00711
        NC(nc_inq_dimid(ncid, "lat", &dimid));
00712
        NC(nc_inq_dimlen(ncid, dimid, &ny));
00713
        if (ny > EY)
         ERRMSG("Too many latitudes!");
00714
00715
00716
        NC(nc_inq_dimid(ncid, "lev", &dimid));
00717
        NC(nc_inq_dimlen(ncid, dimid, &np));
00718
           (np > EP)
00719
         ERRMSG("Too many levels!");
00720
00721
        /* Store dimensions... */
00722
        met->np = (int) np;
        met->nx = (int) nx;
00723
00724
        met->ny = (int) ny;
00725
        /* Get horizontal grid... */
NC(nc_inq_varid(ncid, "lon", &varid));
00726
00727
        NC(nc_get_var_double(ncid, varid, met->lon));
NC(nc_inq_varid(ncid, "lat", &varid));
00728
00729
00730
        NC(nc_get_var_double(ncid, varid, met->lat));
00731
       /* Read meteorological data... */
read_met_help(ncid, "t", "T", met, met->t, 1.0);
00732
00733
00734
00735
        /* Read pressure levels from file...
        NC(nc_inq_varid(ncid, "lev", &varid));
NC(nc_get_var_double(ncid, varid, met->p));
00736
00737
        for (ip = 0; ip < met->np; ip++)
  met->p[ip] /= 100.;
00738
00739
00740
00741
        /* Extrapolate data for lower boundary... */
00742
        read_met_extrapolate(met);
00743
00744
        /\star Check ordering of pressure levels... \star/
        for (ip = 1; ip < met->np; ip++)
  if (met->p[ip - 1] < met->p[ip])
00745
00746
00747
            ERRMSG("Pressure levels must be descending!");
00748
00749
        /* Create periodic boundary conditions... */
00750
        read_met_periodic(met);
00751
00752
        /* Close file... */
00753
        NC(nc_close(ncid));
00754 }
00755
00757
00758 void read_met_extrapolate(
00759
       met t * met) {
```

5.8 libgps.c 253

```
00760
00761
        int ip, ip0, ix, iy;
00762
00763
        /* Loop over columns... */
        for (ix = 0; ix < met->nx; ix++)
  for (iy = 0; iy < met->ny; iy++) {
00764
00765
00766
00767
             /* Find lowest valid data point... */
00768
            for (ip0 = met->np - 1; ip0 >= 0; ip0--)
00769
              if (!gsl_finite(met->t[ix][iy][ip0]))
00770
                break:
00771
            /* Extrapolate... */
for (ip = ip0; ip >= 0; ip--)
00772
00773
00774
              met->t[ix][iy][ip] = met->t[ix][iy][ip + 1];
00775
00776 }
00777
00779
00780 void read_met_help(
00781
        int ncid,
        char *varname,
char *varname2,
00782
00783
00784
        met_t * met,
00785
        float dest[EX][EY][EP],
00786
       float scl) {
00787
00788
       static float help[EX * EY * EP];
00789
00790
       int ip, ix, iy, n = 0, varid;
00791
00792
        /\star Check if variable exists... \star/
00793
        if (nc_inq_varid(ncid, varname, &varid) != NC_NOERR)
00794
         if (nc_inq_varid(ncid, varname2, &varid) != NC_NOERR)
00795
            return:
00796
00797
        /* Read data... */
00798
        NC(nc_get_var_float(ncid, varid, help));
00799
00800
        /\star Copy and check data... \star/
00801
        for (ip = 0; ip < met->np; ip++)
         for (iy = 0; iy < met->ny; iy++)
  for (ix = 0; ix < met->nx; ix++) {
    dest[ix][iy][ip] = scl * help[n++];
00802
00803
00804
00805
              if (fabs(dest[ix][iy][ip] / scl) > 1e14)
00806
                dest[ix][iy][ip] = GSL_NAN;
00807
00808 }
00809
00810 /
       *****************************
00811
00812 void read_met_periodic(
00813
       met_t * met) {
00814
00815
        int ip, iv;
00817
        /* Check longitudes... */
00818
       if (!(fabs(met->lon[met->nx - 1] - met->lon[0]
00819
                    + met -> lon[1] - met -> lon[0] - 360) < 0.01))
00820
          return:
00821
00822
        /* Increase longitude counter... */
00823
        if ((++met->nx) > EX)
00824
          ERRMSG("Cannot create periodic boundary conditions!");
00825
00826
       /* Set longitude... */
       met->lon[met->nx - 1] = met->lon[met->nx - 2] + met->lon[1] - met->
00827
      lon[0];
00828
00829
         /\star Loop over latitudes and pressure levels... \star/
00830
        for (iy = 0; iy < met->ny; iy++)
         for (ip = 0; ip < met->np; ip++)
  met->t[met->nx - 1][iy][ip] = met->t[0][iy][ip];
00831
00832
00833 }
00834
00836
00837 void tropopause(
00838
       gps_t * gps) {
00839
00840
       double zmin;
00841
00842
       int ids, iz, iz2, okay;
00843
00844
        /* Loop over profiles... */
       for (ids = 0; ids < gps->nds; ids++) {
00845
```

```
00846
00847
             /* Set default value... */
00848
             gps->th[ids] = GSL_NAN;
00849
00850
             /* Set minimum altitude... */
00851
             zmin =
               8 - 4 * fabs(cos((90 - gps->lat[ids][gps->nz[ids] / 2]) * M_PI / 180));
00853
             /\star Search tropopause (WMO definition)... \star/
00854
             for (iz = 0; iz < gps->nz[ids]; iz++)
  if (gps->z[ids][iz] >= zmin && gps->z[ids][iz] <= 20.0) {</pre>
00855
00856
                  okay = 1;
00857
                  for (iz2 = iz + 1; iz2 < gps->nz[ids]; iz2++)
00858
                    if (gps->z[ids][iz2] - gps->z[ids][iz] <= 2.0)</pre>
00859
00860
                       if (!gsl_finite(gps->t[ids][iz])
00861
                            !gsl_finite(gps->t[ids][iz2]) ||
                            (gps->t[ids][iz2] - gps->t[ids][iz]) / (gps->z[ids][iz2] - gps->z[ids][iz]) < -2.0)
00862
00863
00864
                          okay = 0;
00865
                  if (okay) {
00866
                    gps->th[ids] = gps->z[ids][iz];
00867
                    break;
00868
                 }
00869
00870
         }
00871 }
00872
00874
00875 void write_gps(
00876
         char *filename.
00877
         gps_t * gps) {
00878
00879
          static double help[NDS * NZ];
00880
          int ids, iz, ncid, dimid[2], time_id, z_id, lon_id, lat_id, p_id, t_id,
00881
00882
            pt_id, wv_id, th_id, nzmax = 0;
00884
          /* Create netCDF file...
00885
          printf("Write GPS-RO file: %s\n", filename);
00886
          NC(nc_create(filename, NC_CLOBBER, &ncid));
00887
00888
          /* Set dimensions... */
NC(nc_def_dim(ncid, "NDS", (size_t) gps->nds, &dimid[0]));
00889
          for (ids = 0; ids < gps->nds; ids++)
00890
         nzmax = GSL_MAX(nzmax, gps->nz[ids]);
NC(nc_def_dim(ncid, "NZ", (size_t) nzmax, &dimid[1]));
00891
00892
00893
00894
         /* Add variables... */
add_var(ncid, "time", "s", "time (seconds since 2000-01-01T00:00Z)",
00895
         add_var(ncid, "time", "s", "time (seconds since 2000-01-01T00:00Z)",

NC_DOUBLE, dimid, &time_id, 1);

add_var(ncid, "z", "km", "altitude", NC_FLOAT, dimid, &z_id, 2);

add_var(ncid, "lon", "deg", "longitude", NC_FLOAT, dimid, &lon_id, 2);

add_var(ncid, "lat", "deg", "latitude", NC_FLOAT, dimid, &lat_id, 2);

add_var(ncid, "p", "hPa", "pressure", NC_FLOAT, dimid, &p_id, 2);

add_var(ncid, "t", "K", "temperature", NC_FLOAT, dimid, &t_id, 2);

add_var(ncid, "wv", "ppm", "water vapor volume mixing ratio",
00896
00897
00898
00899
00900
00901
00902
00903
                    NC_FLOAT, dimid, &wv_id, 2);
          add_var(ncid, "pt", "K", "temperature perturbation",
00904
         NC_FLOAT, dimid, &pt_id, 2);
add_var(ncid, "th", "km", "tropopause height", NC_FLOAT, dimid, &th_id, 1);
00905
00906
00907
00908
           * Leave define mode... */
00909
          NC(nc_enddef(ncid));
00910
00911
          /* Write data... */
00912
          NC(nc_put_var_double(ncid, time_id, gps->time));
          NC(nc_put_var_double(ncid, th_id, gps->th));
for (ids = 0; ids < gps->nds; ids++)
00913
00914
            for (iz = 0; iz < gps->nz[ids]; iz++)
00915
00916
               help[ids * nzmax + iz] = gps->z[ids][iz];
00917
          NC(nc_put_var_double(ncid, z_id, help));
          for (ids = 0; ids < gps->nds; ids++)
  for (iz = 0; iz < gps->nz[ids]; iz++)
   help[ids * nzmax + iz] = gps->lon[ids][iz];
00918
00919
00920
          NC(nc_put_var_double(ncid, lon_id, help));
00921
00922
          for (ids = 0; ids < gps->nds; ids++)
00923
            for (iz = 0; iz < gps->nz[ids]; iz++)
00924
              help[ids * nzmax + iz] = gps->lat[ids][iz];
          NC(nc_put_var_double(ncid, lat_id, help));
00925
00926
          for (ids = 0; ids < gps->nds; ids++)
            for (iz = 0; iz < gps->nz[ids]; iz++)
00928
               help[ids * nzmax + iz] = gps->p[ids][iz];
00929
          NC(nc_put_var_double(ncid, p_id, help));
          for (ids = 0; ids < gps->nds; ids++)
  for (iz = 0; iz < gps->nz[ids]; iz++)
    help[ids * nzmax + iz] = gps->t[ids][iz];
00930
00931
00932
```

```
NC(nc_put_var_double(ncid, t_id, help));
        for (ids = 0; ids < gps->nds; ids++)
for (iz = 0; iz < gps->nz[ids]; iz++)
00935
00936
             help[ids * nzmax + iz] = gps->wv[ids][iz];
        NC(nc_put_var_double(ncid, wv_id, help));
00937
        for (ids = 0; ids < gps->nds; ids++)
  for (iz = 0; iz < gps->nz[ids]; iz++)
00938
00940
              help[ids * nzmax + iz] = gps->pt[ids][iz];
00941
        NC(nc_put_var_double(ncid, pt_id, help));
00942
         /* Close file... */
00943
        NC(nc_close(ncid));
00944
00945 }
```

5.9 libgps.h File Reference

Data Structures

struct gps t

GPS-RO profile data.

struct met_t

Meteorological data.

Functions

 void add_var (int ncid, const char *varname, const char *unit, const char *longname, int type, int dimid[], int *varid, int ndims)

Add variable to netCDF file.

void detrend_met (gps_t *gps, char *metbase, double dt_met)

Detrending by means of meteo data.

• void gauss (gps_t *gps, double dx, double dy)

Calculate horizontal Gaussian mean to extract perturbations.

void grid_gps (gps_t *gps, double zmin, double zmax, int nz)

Interpolate GPS data to regular altitude grid.

• void get_met (char *metbase, double dt_met, double t, met_t *met0, met_t *met1)

Get meteorological data for given timestep.

void get_met_help (double t, int direct, char *metbase, double dt_met, char *filename)

Get meteorological data for timestep.

• void intpol_met_3d (float array[EX][EY][EP], int ip, int ix, int iy, double wp, double wx, double wy, double *var)

Linear interpolation of 3-D meteorological data.

• void intpol_met_space (met_t *met, double p, double lon, double lat, double *t)

Spatial interpolation of meteorological data.

• void intpol_met_time (met_t *met0, met_t *met1, double ts, double p, double lon, double lat, double *t)

Temporal interpolation of meteorological data.

void hamming_low_pass (gps_t *gps, double dz)

Apply vertical Hamming filter to extract perturbations.

void hamming_high_pass (gps_t *gps, double dz)

Apply vertical Hamming filter to reduce noise.

• void poly (gps_t *gps, int dim, double zmin, double zmax)

Remove polynomial fit from perturbation profile.

void poly_help (double *xx, double *yy, int n, int dim, double xmin, double xmax)

Auxiliary function for polynomial interpolation.

void read gps prof (char *filename, gps t *gps)

Read GPS-RO profile.

```
    void read_gps (char *filename, gps_t *gps)
```

Read GPS-RO data file.

void read met (char *filename, met t *met)

Read meteorological data file.

void read_met_extrapolate (met_t *met)

Extrapolate meteorological data at lower boundary.

- void read_met_help (int ncid, char *varname, char *varname2, met_t *met, float dest[EX][EY][EP], float scl)

 Read and convert variable from meteorological data file.
- void read_met_periodic (met_t *met)

Create meteorological data with periodic boundary conditions.

void tropopause (gps_t *gps)

Find tropopause height.

void write_gps (char *filename, gps_t *gps)

Write GPS-RO data file.

- 5.9.1 Function Documentation
- 5.9.1.1 void add_var (int *ncid*, const char * *varname*, const char * *unit*, const char * *longname*, int *type*, int *dimid[]*, int * *varid*, int *ndims*)

Add variable to netCDF file.

Definition at line 5 of file libgps.c.

```
00013
00014
       double dp = GSL_NAN;
00015
00016
00017
        /* Define variable... */
00018
       NC(nc_def_var(ncid, varname, type, ndims, dimid, varid));
00019
00020
        /* Set long name... */
00021
       NC(nc_put_att_text(ncid, *varid, "long_name", strlen(longname), longname));
00022
00023
        /* Set units... */
00024
       NC(nc_put_att_text(ncid, *varid, "units", strlen(unit), unit));
00025
00026
        /* Set fill value... */
       NC(nc_put_att_double(ncid, *varid, "_FillValue", type, 1, &dp));
00027
00028 }
```

5.9.1.2 void detrend_met (gps_t * gps, char * metbase, double dt_met)

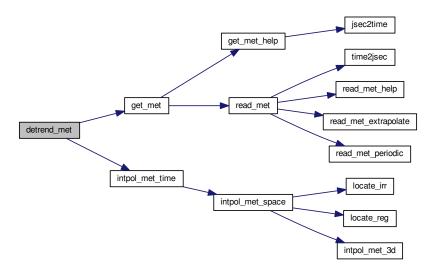
Detrending by means of meteo data.

Definition at line 32 of file libgps.c.

```
00035
                        {
00036
00037
        met_t *met0, *met1;
00038
00039
       double t:
00040
        int ids, iz;
00041
00042
        /* Allocate... */
00043
00044
        ALLOC(met0, met_t, 1);
00045
       ALLOC(met1, met_t, 1);
00046
00047
        /* Loop over profiles... */
00048
       for (ids = 0; ids < gps->nds; ids++) {
00049
```

```
00050
          /* Loop over altitudes... */
00051
          for (iz = 0; iz < gps->nz[ids]; iz++) {
00052
00053
            /\star Get meteorological data... \star/
            get_met(metbase, dt_met, gps->time[ids], met0, met1);
00054
00055
00056
            /* Interpolate meteorological data... */
00057
            intpol_met_time(met0, met1, gps->time[ids], gps->p[ids][iz],
00058
                            gps->lon[ids][iz], gps->lat[ids][iz], &t);
00059
00060
            /* Set perturbation... */
            gps->pt[ids][iz] = gps->t[ids][iz] - t;
00061
00062
00063
00064
00065
        /* Free... */
       free (met0);
00066
00067
       free (met1);
00068 }
```

Here is the call graph for this function:



5.9.1.3 void gauss ($gps_t * gps$, double dx, double dy)

Calculate horizontal Gaussian mean to extract perturbations.

Definition at line 72 of file libgps.c.

```
00075
00076
00077
         double dlat, dlon, w, wsum;
00078
00079
         int ids, ids2, iz;
08000
00081
          /* Loop over profiles... */
         for (ids = 0; ids < gps->nds; ids++) {
00082
00083
           /* Initialize... */
00084
00085
           wsum = 0;
for (iz = 0; iz < gps->nz[ids]; iz++)
00086
00087
              gps->pt[ids][iz] = 0;
00088
           /* Calculate lon-lat standard deviations... */ dlat = dx * 180. / (M_PI * RE) / 2.3548; dlon = dy * 180. / 2.3548
00089
00090
00091
00092
              / (M_PI * RE * cos(gps->lat[ids][gps->nz[ids] / 2] * M_PI / 180.));
```

```
/* Calculate mean temperature... */
for (ids2 = 0; ids2 < gps->nds; ids2++) {
00094
00095
               w = \exp(-0.5 * gsl_pow_2((gps->lon[ids][gps->nz[ids] / 2])
00096
                          - gps->lon[ids2][gps->nz[ids2] / 2]) / dlon)
- 0.5 * gsl_pow_2((gps->lat[ids][gps->nz[ids] / 2]
00097
00098
00099
00100
                                                   gps->lat[ids2][gps->nz[ids2] / 2]) / dlat));
               wsum += w;
for (iz = 0; iz < gps->nz[ids]; iz++)
00101
00102
                  gps \rightarrow pt[ids][iz] += w * gps \rightarrow t[ids2][iz];
00103
00104
00105
00106
             /* Normalize... */
00107
             if (wsum > 0)
               for (iz = 0; iz < gps->nz[ids]; iz++)
   gps->pt[ids][iz] = gps->t[ids][iz] - gps->pt[ids][iz] / wsum;
00108
00109
00110
00111 }
```

5.9.1.4 void grid_gps (gps t * gps, double zmin, double zmax, int nz)

Interpolate GPS data to regular altitude grid.

Definition at line 115 of file libgps.c.

```
00119
00120
00121
       double lat[NZ], lon[NZ], p[NZ], pt[NZ], t[NZ], wv[NZ], z[NZ];
00122
00123
       int ids, iz, iz2;
00124
00125
       /* Check number of altitudes... */
00126
       if (nz > NZ)
00127
        ERRMSG("Too many altitudes!");
00128
00129
       /* Loop over profiles... */
00130
       for (ids = 0; ids < gps->nds; ids++) {
00131
00132
         /* Loop over altitudes... */
00133
         for (iz = 0; iz < nz; iz++) {</pre>
00134
00135
           /* Set altitude... */
          z[iz] = LIN(0.0, zmin, nz - 1.0, zmax, (double) iz);
00136
00137
00138
           /* Get index... */
00139
          iz2 = locate_irr(gps->z[ids], gps->nz[ids], z[iz]);
00140
00141
           /* Interpolate... */
          00142
00143
          lat[iz] = LIN(gps->z[ids][iz2], gps->lat[ids][iz2], gps->z[ids][iz2 + 1], z[iz]);
00144
00145
          00146
00147
00148
          00150
00151
          00152
00153
00154
00155
00156
         /* Copy data... */
         gps->nz[ids] = nz;
for (iz = 0; iz < nz; iz++) {</pre>
00157
00158
          gps->z[ids][iz] = z[iz];
gps->lon[ids][iz] = lon[iz];
00159
00160
          gps->lat[ids][iz] = lat[iz];
00161
          gps->p[ids][iz] = p[iz];
gps->t[ids][iz] = t[iz];
00162
00163
          gps->wv[ids][iz] = wv[iz];
gps->pt[ids][iz] = pt[iz];
00164
00165
00166
00167
      }
00168 }
```

Here is the call graph for this function:



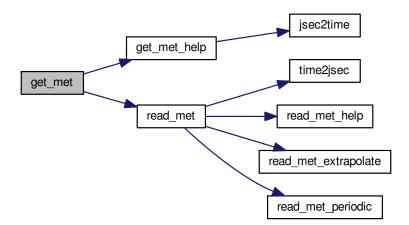
5.9.1.5 void get_met (char * metbase, double dt_met , double t, met_t * met0, met_t * met1)

Get meteorological data for given timestep.

Definition at line 172 of file libgps.c.

```
00177
00178
00179
        char filename[LEN];
00180
00181
        static int init;
00182
00183
        /* Init... */
00184
        if (!init) {
00185
          init = 1;
00186
          get_met_help(t, -1, metbase, dt_met, filename);
read_met(filename, met0);
00187
00188
00189
00190
          get_met_help(t + 1.0, 1, metbase, dt_met, filename);
00191
          read_met(filename, met1);
00192
00193
00194
        /* Read new data... */
00195
        if (t > met1->time) {
00196
        memcpy(met0, met1, sizeof(met_t));
00197
          get_met_help(t, 1, metbase, dt_met, filename);
00198
          read_met(filename, met1);
00199
00200 }
```

Here is the call graph for this function:



5.9.1.6 void get_met_help (double t, int direct, char * metbase, double dt_met, char * filename)

Get meteorological data for timestep.

Definition at line 204 of file libgps.c.

```
00209
00210
00211
        double t6, r;
00212
00213
        int year, mon, day, hour, min, sec;
00214
00215
        /\star Round time to fixed intervals... \star/
00216
        if (direct == -1)
00217
          t6 = floor(t / dt_met) * dt_met;
00218
        else
00219
          t6 = ceil(t / dt_met) * dt_met;
00220
00221
         /* Decode time... */
00222
        jsec2time(t6, &year, &mon, &day, &hour, &min, &sec, &r);
00223
        /* Set filename... */
sprintf(filename, "%s_%d_%02d_%02d_%02d.nc", metbase, year, mon, day, hour);
00224
00225
00226 }
```

Here is the call graph for this function:



5.9.1.7 void intpol_met_3d (float array[EX][EY][EP], int ip, int ix, int iy, double wp, double wx, double wy, double * var)

Linear interpolation of 3-D meteorological data.

Definition at line 230 of file libgps.c.

```
00238
                    {
00239
00240
       double aux00, aux01, aux10, aux11;
00241
00242
       /* Interpolate vertically... */
00243
       aux00 = wp * (array[ix][iy][ip] - array[ix][iy][ip + 1])
00244
         + array[ix][iy][ip + 1];
       00245
00246
00247
00248
         + array[ix + 1][iy][ip + 1];
00249
       aux11 = wp * (array[ix + 1][iy + 1][ip] - array[ix + 1][iy + 1][ip + 1])
00250
         + array[ix + 1][iy + 1][ip + 1];
00251
00252
       /* Interpolate horizontally... */
      aux00 = wy * (aux00 - aux01) + aux01;

aux11 = wy * (aux10 - aux11) + aux11;
00253
00254
00255
       *var = wx * (aux00 - aux11) + aux11;
00256 }
```

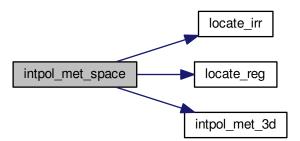
5.9.1.8 void intpol_met_space (met_t * met, double p, double lon, double lat, double * t)

Spatial interpolation of meteorological data.

Definition at line 260 of file libgps.c.

```
00265
00266
00267
           double wp, wx, wy;
00268
00269
           int ip, ix, iy;
00270
           /* Check longitude... */
if (met->lon[met->nx - 1] > 180 && lon < 0)
00271
00272
00273
             lon += 360;
00274
          /* Get indices... */
ip = locate_irr(met->p, met->np, p);
00275
00276
           ix = locate_reg(met->lon, met->nx, lon);
00278
           iy = locate_reg(met->lat, met->ny, lat);
00279
          /* Get weights... */
wp = (met->p[ip + 1] - p) / (met->p[ip + 1] - met->p[ip]);
wx = (met->lon[ix + 1] - lon) / (met->lon[ix + 1] - met->lon[ix]);
wy = (met->lat[iy + 1] - lat) / (met->lat[iy + 1] - met->lat[iy]);
00280
00281
00282
00284
00285
            /* Interpolate... */
00286
           intpol_met_3d(met->t, ip, ix, iy, wp, wx, wy, t);
00287 }
```

Here is the call graph for this function:



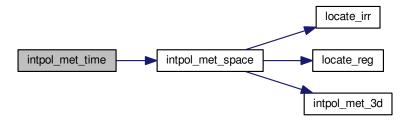
5.9.1.9 void intpol_met_time (met_t * met0, met_t * met1, double ts, double p, double lon, double lat, double * t)

Temporal interpolation of meteorological data.

Definition at line 291 of file libgps.c.

```
{
00299
00300
         double t0, t1, wt;
00301
         /* Spatial interpolation... */
intpol_met_space(met0, p, lon, lat, &t0);
00302
00303
00304
         intpol_met_space(met1, p, lon, lat, &t1);
00305
         /* Get weighting factor... */ wt = (met1->time - ts) / (met1->time - met0->time);
00306
00307
00308
00309
         /* Interpolate... */
00310
         *t = wt * (t0 - t1) + t1;
00311 }
```

Here is the call graph for this function:



5.9.1.10 void hamming_low_pass (gps_t * gps, double dz)

Apply vertical Hamming filter to extract perturbations.

Definition at line 315 of file libgps.c.

```
00317
00318
00319
        double ham[NZ], wsum;
00320
00321
        int ids, iham, iz, nham;
00322
00323
        /* Loop over profiles... */
00324
        for (ids = 0; ids < gps->nds; ids++) {
00325
00326
          /* Calculate Hamming window coefficients... */
          00327
00328
          nham = GSL_MAX(GSL_MIN(nham, NZ), 2);
for (iham = 0; iham < nham; iham++)</pre>
00329
00330
00331
            ham[iham] = 0.54 + 0.46 * cos(M_PI * iham / (nham - 1.0));
00332
00333
          /* Loop over altitudes... */
00334
          for (iz = 0; iz < gps->nz[ids]; iz++) {
00335
            /* Initialize... */
gps->pt[ids][iz] = ham[0] * gps->t[ids][iz];
00336
00337
00338
            wsum = ham[0];
00339
00340
            /\star Loop over filter window... \star/
00341
            for (iham = 1; iham < nham; iham++) {</pre>
00342
              /* Check array range... */
if (iz - iham < 0 || iz + iham >= gps->nz[ids])
00343
00344
00345
                continue;
00346
              /* Check temperature value... */
if (!gsl_finite(gps->t[ids][iz - iham]) ||
00347
00348
                  !gsl_finite(gps->t[ids][iz + iham]))
00349
00350
                continue;
00351
00352
              /\star Check for tropopause... \star/
00353
              if (gsl\_finite(gps->th[ids]) \&\& gps->th[ids] > 0)
                00354
00355
                     || (gps->z[ids][iz] <= gps->th[ids]
00356
00357
                         && gps->z[ids][iz + iham] > gps->th[ids]))
00358
                  continue;
00359
00360
              /\star Apply Hamming filter... \star/
              gps->pt[ids][iz]
00361
                += ham[iham] * (gps->t[ids][iz - iham] + gps->t[ids][iz + iham]);
00362
00363
              wsum += 2 * ham[iham];
00364
00365
00366
            /* Calculate perturbation... */
            gps->pt[ids][iz] = gps->t[ids][iz] - gps->pt[ids][iz] / wsum;
00367
00368
00369
       }
00370 }
```

5.9.1.11 void hamming_high_pass (gps_t * gps, double dz)

Apply vertical Hamming filter to reduce noise.

Definition at line 374 of file libgps.c.

```
00376
                     {
00377
00378
        double ham[NZ], pt[NZ], wsum;
00380
        int ids, iham, iz, nham;
00381
00382
        /* Loop over profiles... */
        for (ids = 0; ids < gps->nds; ids++) {
00383
00384
00385
           /* Calculate Hamming window coefficients... */
          nham = (int) (dz / fabs((gps->z[ids][0] - gps->z[ids][gps->nz[ids] - 1])
/ (gps->nz[ids] - 1.0)) + 0.5);
00386
00387
00388
          nham = GSL\_MAX(GSL\_MIN(nham, NZ), 2);
          for (iham = 0; iham < nham; iham++)
ham[iham] = 0.54 + 0.46 * cos(M_PI * iham / (nham - 1.0));</pre>
00389
00390
00391
00392
           /* Loop over altitudes... */
00393
           for (iz = 0; iz < gps->nz[ids]; iz++) {
00394
00395
             /* Initialize... */
00396
             pt[iz] = ham[0] * gps->pt[ids][iz];
00397
             wsum = ham[0];
00398
00399
             /* Loop over filter window... */
00400
             for (iham = 1; iham < nham; iham++) {</pre>
00401
00402
              /* Check array range... */
              if (iz - iham < 0 || iz + iham >= gps->nz[ids])
00403
00404
                continue;
00405
00406
               /* Check temperature value... */
               if (!gsl_finite(gps->t[ids][iz - iham]) ||
00407
00408
                   !gsl_finite(gps->t[ids][iz + iham]))
00409
                continue:
00410
00411
               /* Apply Hamming filter... */
              pt[iz]
00412
00413
                 += ham[iham] * (gps->pt[ids][iz - iham] + gps->pt[ids][iz + iham]);
               wsum += 2 * ham[iham];
00414
00415
            }
00416
00417
             /* Normalize... */
00418
            pt[iz] /= wsum;
00419
00420
00421
          /\star Set perturbation... \star/
          for (iz = 0; iz < gps->nz[ids]; iz++)
   gps->pt[ids][iz] = pt[iz];
00422
00424
00425 }
```

5.9.1.12 void poly (gps_t * gps, int dim, double zmin, double zmax)

Remove polynomial fit from perturbation profile.

Definition at line 429 of file libgps.c.

```
00433
00434
00435
         double bg[NZ];
00436
00437
         int ids, iz;
00438
00439
          /* Loop over profiles... ∗/
00440
         for (ids = 0; ids < gps->nds; ids++) {
00441
           /* Set profile... */
for (iz = 0; iz < gps->nz[ids]; iz++)
bg[iz] = gps->pt[ids][iz];
00442
00443
00444
00445
```

Here is the call graph for this function:



5.9.1.13 void poly_help (double *xx, double *yy, int n, int dim, double xmin, double xmax)

Auxiliary function for polynomial interpolation.

Definition at line 457 of file libgps.c.

```
00463
00464
00465
         gsl_multifit_linear_workspace *work;
00466
         gsl_matrix *cov, *X;
00467
         gsl_vector *c, *x, *y;
00468
00469
         double chisq, xx2[NZ], yy2[NZ];
00470
00471
         size_t i, i2, n2 = 0;
00473
         /* Check for nan... */
00474
         for (i = 0; i < (size_t) n; i++)</pre>
          if (xx[i] \ge xmin & xx[i] \le xmax & gsl_finite(yy[i])) {
00475
00476
             xx2[n2] = xx[i];
00477
              yy2[n2] = yy[i];
00478
             n2++;
00479
00480
         if ((int) n2 < dim) {</pre>
         for (i = 0; i < (size_t) n; i++)
    yy[i] = GSL_NAN;</pre>
00481
00482
00483
           return;
00484
00485
         /* Allocate...
00486
         work = gsl_multifit_linear_alloc((size_t) n2, (size_t) dim);
00487
         cov = gsl_matrix_alloc((size_t) dim, (size_t) dim);
00488
         X = gsl_matrix_alloc((size_t) n2, (size_t) dim);
c = gsl_vector_alloc((size_t) dim);
00489
00490
         x = gsl_vector_alloc((size_t) n2);
00491
00492
         y = gsl_vector_alloc((size_t) n2);
00493
         /* Compute polynomial fit... */
for (i = 0; i < (size_t) n2; i++)</pre>
00494
00495
          gsl_vector_set(x, i, xx2[i]);
gsl_vector_set(y, i, yy2[i]);
for (i2 = 0; i2 < (size_t) dim; i2++)
00496
00497
00498
00499
              gsl_matrix_set(X, i, i2, pow(gsl_vector_get(x, i), (double) i2));
00500
         gsl_multifit_linear(X, y, c, cov, &chisq, work);
for (i = 0; i < (size_t) n; i++)</pre>
00501
00502
           yy[i] = gsl_poly_eval(c->data, (int) dim, xx[i]);
00503
00504
00505
00506
         gsl_multifit_linear_free(work);
00507
         gsl_matrix_free(cov);
00508
         gsl_matrix_free(X);
gsl_vector_free(c);
00509
         gsl_vector_free(x);
00511
         gsl_vector_free(y);
00512 }
```

```
5.9.1.14 void read_gps_prof ( char * filename, gps_t * gps )
```

Read GPS-RO profile.

Definition at line 516 of file libgps.c.

```
00518
00519
00520
          char bad[10];
00521
          double t0, t1, zmin = 1e100, zmax = -1e100;
00522
00523
00524
          int ncid, dimid, varid;
00526
          size t iz, nz;
00527
          /* Open netCDF file... */ printf("Read GPS-RO profile: %s\n", filename);
00528
00529
00530
          NC (nc_open(filename, NC_NOWRITE, &ncid));
          /* Get dimensions... */
NC(nc_inq_dimid(ncid, "MSL_alt", &dimid));
00532
00533
          NC(nc_inq_dimlen(ncid, dimid, &nz));
00534
00535
          gps->nz[gps->nds] = (int) nz;
if (nz > NZ)
00536
00537
             ERRMSG("Too many altitudes!");
00538
          /\star Check data quality flag... \star/
00539
          NC(nc_get_att_text(ncid, NC_GLOBAL, "bad", bad));
if (bad[0] != '0') {
00540
00541
00542
            NC(nc_close(ncid));
00543
            return;
00544
00545
          /* Get time... */
00546
          \label{eq:ncd_ncd} $$NC(nc\_get\_att\_double(ncid, NC\_GLOBAL, "start\_time", &t0)); $$NC(nc\_get\_att\_double(ncid, NC\_GLOBAL, "stop\_time", &t1)); $$gps->time[gps->nds] = 0.5 * (t0 + t1) - 630720000.0;
00547
00548
00549
00550
00551
00552
          NC(nc_inq_varid(ncid, "MSL_alt", &varid));
          NC(nc_get_var_double(ncid, varid, gps->z[gps->nds]));
NC(nc_inq_varid(ncid, "Lon", &varid));
00553
00554
          NC(nc_get_var_double(ncid, varid, gps->lon[gps->nds]));
NC(nc_inq_varid(ncid, "Lat", &varid));
NC(nc_get_var_double(ncid, varid, gps->lat[gps->nds]));
00555
00557
00558
          NC(nc_inq_varid(ncid, "Pres", &varid));
          NC(nc_get_var_double(ncid, varid, gps->p[gps->nds]));
NC(nc_inq_varid(ncid, "Temp", &varid));
00559
00560
          NC(nc_get_var_double(ncid, varid, gps->t[gps->nds]));
if (nc_ing_varid(ncid, "Vp", &varid) == NC_NOERR)
00561
00562
00563
             NC(nc_get_var_double(ncid, varid, gps->wv[gps->nds]));
00564
00565
          /* Check altitude range... */
          for (iz = 0; iz < nz; iz++)
  if (gps->p[gps->nds][iz] != -999 && gps->t[gps->nds][iz] != -999) {
00566
00567
               zmin = GSL_MIN(zmin, gps->z[gps->nds][iz]);
zmax = GSL_MAX(zmax, gps->z[gps->nds][iz]);
00568
00569
00570
00571
          if (zmin > 5 || zmax < 35) {
00572
            NC(nc_close(ncid));
00573
            return:
00574
00575
00576
          /* Check data... */
00577
          for (iz = 0; iz < nz; iz++)
00578
             if (gps \rightarrow lon[gps \rightarrow nds][iz] == -999 \mid \mid
                  | Ggps-710H[gps-7nds][iz] == -999 || | |
| gps-71gps-7nds][iz] == -999 ||
| gps-71gps-7nds][iz] == -999 || gps-7wv[gps-7nds][iz] == -999) ||
00579
00580
00581
00582
                gps->lon[gps->nds][iz] = GSL_NAN;
00583
                gps->lat[gps->nds][iz] = GSL_NAN;
00584
                gps->p[gps->nds][iz] = GSL_NAN;
00585
                gps->t[gps->nds][iz] = GSL_NAN;
                gps->wv[gps->nds][iz] = GSL_NAN;
00586
00587
00588
00589
          /* Convert temperature...
00590
          for (iz = 0; iz < nz; iz++)</pre>
00591
             gps->t[gps->nds][iz] += 273.15;
00592
00593
          /* Convert water vapor... */
00594
          for (iz = 0; iz < nz; iz++)
```

5.9.1.15 void read_gps (char * filename, gps_t * gps_)

Read GPS-RO data file.

Definition at line 607 of file libgps.c.

```
00609
00610
00611
        int ids, ncid, dimid, varid;
00612
00613
        size_t start[2], count[2], nds, nz;
00614
        /* Read netCDF file... */
printf("Read GPS-RO file: %s\n", filename);
00615
00616
00617
        NC(nc open(filename, NC NOWRITE, &ncid));
00618
        /* Get dimensions... */
NC(nc_inq_dimid(ncid, "NDS", &dimid));
00619
00620
00621
        NC(nc_inq_dimlen(ncid, dimid, &nds));
        gps->nds = (int) nds;
if (nds > NDS)
00622
00623
           ERRMSG("Too many profiles!");
00624
00625
00626
        NC(nc_inq_dimid(ncid, "NZ", &dimid));
00627
        NC(nc_inq_dimlen(ncid, dimid, &nz));
        if (nz > NZ)
00628
          ERRMSG("Too many profiles!");
00629
00630
00631
        /* Loop over profiles... ∗/
00632
        for (ids = 0; ids < gps->nds; ids++) {
00633
00634
          /* Set profile index... */
          start[0] = (size_t) ids;
count[0] = 1;
00635
00636
          start[1] = 0;
00637
00638
          count[1] = nz;
00639
00640
           /\star Set number of altitudes... \star/
00641
          gps->nz[ids] = (int) nz;
00642
00643
           /* Read data... */
00644
           NC(nc_inq_varid(ncid, "time", &varid));
00645
          NC(nc_get_vara_double(ncid, varid, start, count, &gps->time[ids]));
00646
           NC(nc_inq_varid(ncid, "z", &varid));
00647
00648
          NC(nc_get_vara_double(ncid, varid, start, count, gps->z[ids]));
00649
          NC(nc_inq_varid(ncid, "lon", &varid));
NC(nc_get_vara_double(ncid, varid, start, count, gps->lon[ids]));
00650
00651
00652
          NC(nc_inq_varid(ncid, "lat", &varid));
NC(nc_get_vara_double(ncid, varid, start, count, gps->lat[ids]));
00653
00654
00655
00656
           NC(nc_inq_varid(ncid, "p", &varid));
00657
           NC(nc_get_vara_double(ncid, varid, start, count, gps->p[ids]));
00658
00659
           NC(nc_inq_varid(ncid, "t", &varid));
00660
          NC(nc_get_vara_double(ncid, varid, start, count, gps->t[ids]));
00661
           NC(nc_inq_varid(ncid, "wv", &varid));
00662
00663
           NC(nc_get_vara_double(ncid, varid, start, count, gps->wv[ids]));
00664
          NC(nc_inq_varid(ncid, "pt", &varid));
00665
00666
          NC(nc_get_vara_double(ncid, varid, start, count, gps->pt[ids]));
00667
           NC(nc_inq_varid(ncid, "th", &varid));
00668
00669
          NC(nc_get_vara_double(ncid, varid, start, count, &gps->th[ids]));
00670
00671
00672
         /* Close file... */
00673
        NC(nc_close(ncid));
00674 }
```

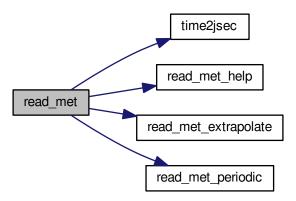
```
5.9.1.16 void read_met ( char * filename, met_t * met )
```

Read meteorological data file.

Definition at line 678 of file libgps.c.

```
00680
00681
00682
        char tstr[10]:
00683
        int ip, dimid, ncid, varid, year, mon, day, hour;
00684
00686
        size_t np, nx, ny;
00687
00688
        /* Write info... */
        printf("Read meteorological data: %s\n", filename);
00689
00690
00691
        /* Get time from filename... */
00692
        sprintf(tstr, "%.4s", &filename[strlen(filename) - 16]);
00693
        year = atoi(tstr);
         sprintf(tstr, "%.2s", &filename[strlen(filename) - 11]);
00694
00695
        mon = atoi(tstr);
00696
        sprintf(tstr, "%.2s", &filename[strlen(filename) - 8]);
00697
        day = atoi(tstr);
00698
        sprintf(tstr, "%.2s", &filename[strlen(filename) - 5]);
00699
        hour = atoi(tstr);
00700
        time2jsec(year, mon, day, hour, 0, 0, 0, &met->time);
00701
00702
        /* Open netCDF file... */
00703
        NC(nc_open(filename, NC_NOWRITE, &ncid));
00704
        /* Get dimensions... */
NC(nc_inq_dimid(ncid, "lon", &dimid));
00705
00706
00707
        NC(nc_inq_dimlen(ncid, dimid, &nx));
00708
            (nx > EX)
00709
           ERRMSG("Too many longitudes!");
00710
00711
         NC(nc_inq_dimid(ncid, "lat", &dimid));
00712
        NC(nc_inq_dimlen(ncid, dimid, &ny));
00713
         if (ny > EY)
00714
           ERRMSG("Too many latitudes!");
00715
00716
        NC(nc_inq_dimid(ncid, "lev", &dimid));
00717
        NC(nc_inq_dimlen(ncid, dimid, &np));
        if (np > EP)
00718
00719
          ERRMSG("Too many levels!");
00720
00721
        /* Store dimensions... */
        met->np = (int) np;
met->nx = (int) nx;
00722
00723
        met->ny = (int) ny;
00724
00725
        /* Get horizontal grid... */
NC(nc_inq_varid(ncid, "lon", &varid));
NC(nc_get_var_double(ncid, varid, met->lon));
NC(nc_inq_varid(ncid, "lat", &varid));
00726
00727
00728
00729
00730
        NC(nc_get_var_double(ncid, varid, met->lat));
00731
00732
        /* Read meteorological data... */
read_met_help(ncid, "t", "T", met, met->t, 1.0);
00733
00734
00735
            Read pressure levels from file...
00736
        NC(nc_inq_varid(ncid, "lev", &varid));
00737
        \label{local_nc_delta} \mbox{NC (nc\_get\_var\_double (ncid, varid, met->p));}
        for (ip = 0; ip < met->np; ip++)
  met->p[ip] /= 100.;
00738
00739
00740
00741
        /* Extrapolate data for lower boundary... */
00742
        read_met_extrapolate(met);
00743
00744
        /\star Check ordering of pressure levels... \star/
00745
        for (ip = 1; ip < met->np; ip++)
  if (met->p[ip - 1] < met->p[ip])
00746
00747
             ERRMSG("Pressure levels must be descending!");
00748
         /* Create periodic boundary conditions... */
00749
00750
        read_met_periodic(met);
00751
00752
         /* Close file... */
00753
        NC(nc_close(ncid));
00754 }
```

Here is the call graph for this function:



5.9.1.17 void read_met_extrapolate (met_t * met)

Extrapolate meteorological data at lower boundary.

Definition at line 758 of file libgps.c.

```
00759
00760
         int ip, ip0, ix, iy;
00763
         /* Loop over columns... */
00764
         for (ix = 0; ix < met->nx; ix++)
00765
          for (iy = 0; iy < met->ny; iy++) {
00766
             /* Find lowest valid data point... */
for (ip0 = met->np - 1; ip0 >= 0; ip0--)
00767
00768
00769
               if (!gsl_finite(met->t[ix][iy][ip0]))
00770
                 break;
00771
00772
             /* Extrapolate... */
for (ip = ip0; ip >= 0; ip--)
00773
00774
               met->t[ix][iy][ip] = met->t[ix][iy][ip + 1];
00775
00776 }
```

5.9.1.18 void read_met_help (int ncid, char * varname, char * varname2, met_t * met, float dest[EX][EY][EP], float scl)

Read and convert variable from meteorological data file.

Definition at line 780 of file libgps.c.

```
00786
00787
00788
        static float help[EX * EY * EP];
00790
        int ip, ix, iy, n = 0, varid;
00791
00792
        /\star Check if variable exists... \star/
        if (nc_inq_varid(ncid, varname, &varid) != NC_NOERR)
00793
         if (nc_inq_varid(ncid, varname2, &varid) != NC_NOERR)
00794
00795
           return;
00796
```

```
/* Read data...
00798
          NC(nc_get_var_float(ncid, varid, help));
00799
00800
          /* Copy and check data... */
          for (ip = 0; ip < met->np; ip++)
  for (iy = 0; iy < met->ny; iy++)
   for (ix = 0; ix < met->nx; ix++) {
00801
00802
00804
                 dest[ix][iy][ip] = scl * help[n++];
00805
                 if (fabs(dest[ix][iy][ip] / scl) > 1e14)
00806
                    dest[ix][iy][ip] = GSL_NAN;
               }
00807
00808 }
```

5.9.1.19 void read_met_periodic (met_t * met)

Create meteorological data with periodic boundary conditions.

Definition at line 812 of file libgps.c.

```
00814
00815
        int ip, iy;
00816
00817
        /* Check longitudes... */
        if (!(fabs(met->lon[met->nx - 1] - met->lon[0]
                     + met->lon[1] - met->lon[0] - 360) < 0.01))
00820
00821
        /\star Increase longitude counter... \star/
00822
00823
        if ((++met->nx) > EX)
00824
          ERRMSG("Cannot create periodic boundary conditions!");
00825
        /* Set longitude... */
00826
00827
        met \rightarrow lon[met \rightarrow nx - 1] = met \rightarrow lon[met \rightarrow nx - 2] + met \rightarrow lon[1] - met \rightarrow
      lon[0];
00828
00829
         /* Loop over latitudes and pressure levels... */
        for (iy = 0; iy < met->ny; iy++)
00831
          for (ip = 0; ip < met->np; ip++)
00832
             met->t[met->nx - 1][iy][ip] = met->t[0][iy][ip];
00833 }
```

5.9.1.20 void tropopause (gps_t * gps)

Find tropopause height.

Definition at line 837 of file libgps.c.

```
00838
00839
00840
         double zmin;
00841
00842
        int ids, iz, iz2, okay;
00843
00844
        /* Loop over profiles... */
00845
         for (ids = 0; ids < gps->nds; ids++) {
00846
00847
           /* Set default value... */
00848
           gps->th[ids] = GSL_NAN;
00849
00850
           /* Set minimum altitude... */
00851
           zmin =
00852
            8 - 4 * fabs(cos((90 - gps->lat[ids][gps->nz[ids] / 2]) * M_PI / 180));
00853
00854
           /\star Search tropopause (WMO definition)... \star/
00855
           for (iz = 0; iz < gps->nz[ids]; iz++)
  if (gps->z[ids][iz] >= zmin && gps->z[ids][iz] <= 20.0) {</pre>
00856
               okay = 1;
00858
               for (iz2 = iz + 1; iz2 < gps->nz[ids]; iz2++)
00859
                  if (gps->z[ids][iz2] - gps->z[ids][iz] <= 2.0)</pre>
                    if (!gsl_finite(gps->t[ids][iz]) ||
00860
                         !gsl_finite(gps->t[ids][iz2]) ||
00861
                        (gps->t[ids][iz2] - gps->t[ids][iz])
/ (gps->z[ids][iz2] - gps->z[ids][iz]) < -2.0)</pre>
00862
00863
00864
                      okay = 0;
               if (okay) {
00865
00866
                  gps->th[ids] = gps->z[ids][iz];
00867
                 break:
00868
00869
00870
        }
00871 }
```

```
5.9.1.21 void write_gps ( char * filename, gps_t * gps )
```

Write GPS-RO data file.

Definition at line 875 of file libgps.c.

```
00877
00878
00879
         static double help[NDS * NZ];
00880
00881
         int ids, iz, ncid, dimid[2], time_id, z_id, lon_id, lat_id, p_id, t_id,
           pt_id, wv_id, th_id, nzmax = 0;
00883
          /* Create netCDF file...
00884
          printf("Write GPS-RO file: %s\n", filename);
00885
          NC(nc_create(filename, NC_CLOBBER, &ncid));
00886
00887
00888
             Set dimensions...
00889
          NC(nc_def_dim(ncid, "NDS", (size_t) gps->nds, &dimid[0]));
00890
          for (ids = 0; ids < gps->nds; ids++)
          nzmax = GSL_MAX(nzmax, gps->nz[ids]);
NC(nc_def_dim(ncid, "NZ", (size_t) nzmax, &dimid[1]));
00891
00892
00893
          /* Add variables... */
add_var(ncid, "time", "s", "time (seconds since 2000-01-01T00:00Z)",
00894
00895
         add_var(ncid, "time", "s", "time (seconds since 2000-01-01T00:00Z)",

NC_DOUBLE, dimid, &time_id, 1);

add_var(ncid, "z", "km", "altitude", NC_FLOAT, dimid, &z_id, 2);

add_var(ncid, "lon", "deg", "longitude", NC_FLOAT, dimid, &lon_id, 2);

add_var(ncid, "lat", "deg", "latitude", NC_FLOAT, dimid, &lat_id, 2);

add_var(ncid, "p", "hPa", "pressure", NC_FLOAT, dimid, &p_id, 2);

add_var(ncid, "t", "K", "temperature", NC_FLOAT, dimid, &t_id, 2);

add_var(ncid, "wv", "ppm", "water vapor volume mixing ratio",

NC_FLOAT_ dimid_ fame_id 2);
00896
00897
00898
00899
00900
00901
00902
         00903
00904
00905
00906
00907
00908
          /* Leave define mode... */
00909
          NC(nc_enddef(ncid));
00910
00911
          /* Write data... */
00912
          NC(nc_put_var_double(ncid, time_id, gps->time));
          NC(nc_put_var_double(ncid, th_id, gps->th));
00913
00914
          for (ids = 0; ids < gps->nds; ids++)
            for (iz = 0; iz < gps->nz[ids]; iz++)
00915
00916
              help[ids * nzmax + iz] = gps->z[ids][iz];
          NC(nc_put_var_double(ncid, z_id, help));
for (ids = 0; ids < gps->nds; ids++)
00917
00918
            for (iz = 0; iz < gps->nz[ids]; iz++)
00919
00920
               help[ids * nzmax + iz] = gps->lon[ids][iz];
00921
          NC(nc_put_var_double(ncid, lon_id, help));
00922
          for (ids = 0; ids < gps->nds; ids++)
00923
           for (iz = 0; iz < gps->nz[ids]; iz++)
              help[ids * nzmax + iz] = gps->lat[ids][iz];
00924
          NC(nc_put_var_double(ncid, lat_id, help));
00925
00926
          for (ids = 0; ids < gps->nds; ids++)
00927
           for (iz = 0; iz < gps->nz[ids]; iz++)
00928
              help[ids * nzmax + iz] = gps->p[ids][iz];
          NC(nc_put_var_double(ncid, p_id, help));
00929
00930
          for (ids = 0; ids < gps->nds; ids++)
            for (iz = 0; iz < gps->nz[ids]; iz++)
00931
              help[ids * nzmax + iz] = gps->t[ids][iz];
00932
00933
          NC(nc_put_var_double(ncid, t_id, help));
          for (ids = 0; ids < gps->nds; ids++)
  for (iz = 0; iz < gps->nz[ids]; iz++)
00934
00935
              help[ids * nzmax + iz] = gps->wv[ids][iz];
00936
         NC(nc_put_var_double(ncid, wv_id, help));
for (ids = 0; ids < gps->nds; ids++)
00937
00938
00939
          for (iz = 0; iz < gps->nz[ids]; iz++)
00940
              help[ids * nzmax + iz] = gps->pt[ids][iz];
00941
         NC(nc_put_var_double(ncid, pt_id, help));
00942
00943
          /* Close file... */
00944
          NC(nc_close(ncid));
00945 }
```

5.10 libgps.h 271

Here is the call graph for this function:



5.10 libgps.h

```
00001 #include <netcdf.h>
00002 #include <gsl/gsl_multifit.h>
00003 #include <gsl/gsl_poly.h>
00004 #include "jurassic.h"
00005
00006 /*
00007
        Dimensions...
80000
00009
00011 #define EP 73
00012
00014 #define EX 721
00015
00017 #define EY 361
00018
00020 #define NDS 10000
00021
00023 #define NZ 5000
00024
00025 /* --
00026
        Macros...
00027
00028
00030 #define NC(cmd) {
       if((cmd)!=NC_NOERR)
00031
00032
           ERRMSG(nc_strerror(cmd));
00033
00034
00035 /* -----
00036
         Structs...
00037
00038
00040 typedef struct {
00041
00043
        int nds;
00044
00046
        int nz[NDS];
00047
00049
        double time[NDS];
00050
00052
        double z[NDS][NZ];
00053
00055
        double lon[NDS][NZ];
00056
00058
        double lat[NDS][NZ];
00059
00061
        double p[NDS][NZ];
00062
00064
        double t[NDS][NZ];
00065
00067
        double wv[NDS][NZ];
00068
00070
        double pt[NDS][NZ];
00071
00073
        double th[NDS];
00074
00075 } gps_t;
00076
00078 typedef struct { 00079
00081
        double time:
00082
00084
        int nx;
```

```
00085
00087
        int ny;
00088
00090
        int np;
00091
00093
        double lon[EX];
00094
00096
        double lat[EY];
00097
00099
        double p[EP];
00100
00102
        float t[EX][EY][EP];
00103
00104 } met_t;
00105
00106 /* -----
00107
         Functions...
00108
00109
00111 void add_var(
00112
      int ncid,
00113
        const char *varname,
        const char *unit,
const char *longname,
00114
00115
00116
        int type,
00117
        int dimid[],
00118
        int *varid,
00119
        int ndims);
00120
00122 void detrend_met(
       gps_t * gps,
char *metbase,
00123
00124
00125
        double dt_met);
00126
00128 void gauss(
00129
        gps_t * gps,
double dx,
00130
00131
        double dy);
00132
00134 void grid_gps(
        gps_t * gps,
double zmin,
double zmax,
00135
00136
00137
00138
        int nz);
00139
00141 void get_met(
00142
       char *metbase,
00143
        double dt_met,
00144
        double t,
00145
        met_t * met0,
00146
        met_t * met1);
00147
00149 void get_met_help(
00150
       double t,
00151
        int direct.
00152
        char *metbase,
00153
        double dt_met,
00154
        char *filename);
00155
00157 void intpol_met_3d(
        float array[EX][EY][EP],
00158
00159
        int ip,
00160
        int ix,
00161
        int iy,
00162
        double wp,
00163
        double wx,
00164
        double wy,
double *var);
00165
00166
00168 void intpol_met_space(
00169
        met_t * met,
00170
        double p,
        double lon, double lat,
00171
00172
00173
        double *t);
00174
00176 void intpol_met_time(
00177
        met_t * met0,
met_t * met1,
00178
00179
        double ts,
00180
        double p, double lon,
00181
00182
        double lat,
00183
        double *t);
00184
00186 void hamming_low_pass(
00187
       qps_t * qps,
```

```
00188
        double dz);
00189
00191 void hamming_high_pass(
00192 gps_t * gps,
        double dz);
00193
00194
00196 void poly(
00197
        gps_t * gps,
00198
        int dim,
00199
        double zmin,
00200
        double zmax);
00201
00203 void poly_help(
00204
       double *xx,
00205
        double *yy,
00206
        int n,
00207
        int dim.
00208
        double xmin,
double xmax);
00209
00210
00212 void read_gps_prof(
00213
        char *filename,
        gps_t * gps);
00214
00215
00217 void read_qps(
00218 char *filename,
00219
        gps_t * gps);
00220
00222 void read_met(
00223 char *filename,
00224 met_t * met);
00225
00227 void read_met_extrapolate(
00228
       met_t * met);
00229
00231 void read_met_help(
00232
        int ncid,
        char *varname,
00234
        char *varname2,
00235
        met_t * met,
00236
        float dest[EX][EY][EP],
00237
        float scl);
00238
00240 void read_met_periodic(
00241 met_t * met);
00242
00244 void tropopause(
00245 gps_t * gps);
00246
00248 void write_qps(
00240 void write_gps(
00249 char *filename,
00250 gps_t * gps);
```

5.11 map.c File Reference

Functions

• int main (int argc, char *argv[])

5.11.1 Function Documentation

5.11.1.1 int main (int *argc*, char * *argv[*])

Definition at line 3 of file map.c.

```
00005

00006

00007 gps_t *gps;

00008

00009 FILE *out;

00010

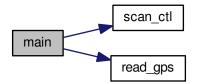
00011 double z;

00012

00013 int ids, iz;
```

```
00014
00015
         /* Allocate... */
00016
         ALLOC(gps, gps_t, 1);
00017
00018
         /* Check arguments... */
00019
         if (argc < 4)
00020
           ERRMSG("Give parameters: <ctl> <gps.nc> <map.tab>");
00021
        /* Get control parameters... */
z = scan_ctl(argc, argv, "Z", -1, "20", NULL);
00022
00023
00024
00025
         /* Read gps data... */
00026
         read_gps(argv[2], gps);
00027
00028
         /* Create output file... */
         printf("Write map data: %s\n", argv[3]);
if (!(out = fopen(argv[3], "w")))
00029
00030
00031
           ERRMSG("Cannot create file!");
00032
00033
         /* Write header... */
00034
         fprintf(out,
                  "# $1 = time [sec]\n"
"# $2 = altitude [km]\n"
"# $3 = longitude [deg]\n"
"# $4 = latitude [deg]\n"
00035
00036
00037
00038
00039
                  "# $5 = pressure [hPa]\n"
00040
                  "# $6 = temperature [K] \n"
                  "# $7 = water vapor vmr [ppm]\n"
"# $8 = temperature perturbation [K]\n"
00041
00042
                  "# $9 = tropopause height [km]\n\n");
00043
00044
00045
         /* Write data... */
         for (ids = 0; ids < gps->nds; ids++)
  for (iz = 0; iz < gps->nz[ids]; iz++)
00046
00047
             00048
00049
00050
00051
00052
00053
               break;
00054
00055
         /* Close file... */
00056
00057
         fclose(out);
00058
00059
         /* Free... */
00060
        free(gps);
00061
         return EXIT_SUCCESS;
00062
00063 }
```

Here is the call graph for this function:



5.12 map.c

```
00001 #include "libgps.h"
00002
00003 int main(
00004 int argc,
00005 char *argv[]) {
```

```
00006
00007
        gps_t *gps;
00008
00009
        FILE *out;
00010
00011
        double z:
00012
00013
        int ids, iz;
00014
00015
         /* Allocate... */
00016
        ALLOC(gps, gps_t, 1);
00017
00018
        /* Check arguments... */
00019
        if (argc < 4)
00020
          ERRMSG("Give parameters: <ctl> <gps.nc> <map.tab>");
00021
        /* Get control parameters... */
z = scan_ctl(argc, argv, "Z", -1, "20", NULL);
00022
00023
00024
00025
        /* Read gps data... */
00026
        read_gps(argv[2], gps);
00027
        /* Create output file... */
printf("Write map data: %s\n", argv[3]);
if (!(out = fopen(argv[3], "w")))
00028
00029
00030
         ERRMSG("Cannot create file!");
00032
00033
        /* Write header... */
        fprintf(out,
    "# $1 = time [sec]\n"
00034
00035
                  "# $2 = altitude [km] \n"
00036
                 "# $3 = longitude [deg]\n"
"# $4 = latitude [deg]\n"
00037
00038
00039
                 "# $5 = pressure [hPa] \n"
                 "# $6 = temperature [K]\n"
"# $7 = water vapor vmr [ppm]\n"
"# $8 = temperature perturbation [K]\n"
00040
00041
00042
                  "# $9 = tropopause height [km]\n\n");
00044
00045
        /* Write data... */
        00046
00047
00048
00049
00050
00051
                        gps->lat[ids][iz], gps->p[ids][iz], gps->t[ids][iz],
00052
                        gps->wv[ids][iz], gps->pt[ids][iz], gps->th[ids]);
00053
               break;
             }
00054
00055
00056
         /* Close file... */
00057
        fclose(out);
00058
00059
        /* Free... */
00060
        free(gps);
00061
        return EXIT_SUCCESS;
00063 }
```

5.13 perturbation.c File Reference

Functions

• int main (int argc, char *argv[])

5.13.1 Function Documentation

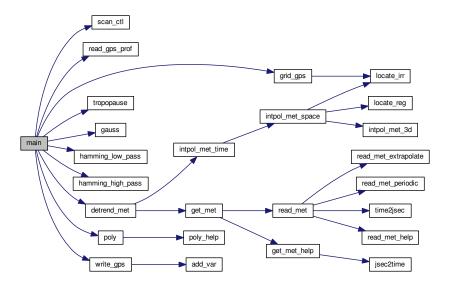
5.13.1.1 int main (int *argc*, char * *argv[*])

Definition at line 7 of file perturbation.c.

```
00009
00010
00011
           gps_t *gps;
00012
00013
          FILE *in:
00014
00015
          char metbase[LEN];
00016
00017
           double dt_met, gauss_dx, gauss_dy, grid_zmin, grid_zmax, ham_dz, ham_dz2,
00018
             poly_zmax, poly_zmin;
00019
00020
          int grid_nz, iarg, poly_dim;
00021
00022
            /* Allocate... */
00023
           ALLOC(gps, gps_t, 1);
00024
00025
           /* Check arguments... */
00026
           if (argc < 4)
00027
              ERRMSG("Give parameters: <ctl> <out.nc> <gps1.nc> [<gps2.nc> ...]");
00028
           /* Get control parameters... */
dt_met = scan_ctl(argc, argv, "DT_MET", -1, "21600", NULL);
00029
          dt_met = scan_ctl(argc, argv, "DT_MET", -1, "21600", NULL);
gauss_dx = scan_ctl(argc, argv, "GAUSS_DX", -1, "-999", NULL);
gauss_dy = scan_ctl(argc, argv, "GAUSS_DX", -1, "-999", NULL);
grid_zmin = scan_ctl(argc, argv, "GRID_ZMIN", -1, "0", NULL);
grid_zmax = scan_ctl(argc, argv, "GRID_ZMAX", -1, "40", NULL);
grid_nz = (int) scan_ctl(argc, argv, "GRID_NZ", -1, "-1", NULL);
ham_dz = scan_ctl(argc, argv, "HAM_DZ", -1, "-999", NULL);
ham_dz2 = scan_ctl(argc, argv, "HAM_DZ2", -1, "-999", NULL);
scan_ctl(argc, argv, "METBASE", -1, "", metbase);
poly_dim = (int) scan_ctl(argc, argv, "POLY_DIM", -1, "5", NULL);
poly_zmin = scan_ctl(argc, argv, "POLY_ZMIN", -1, "0", NULL);
poly_zmax = scan_ctl(argc, argv, "POLY_ZMAX", -1, "40", NULL);
00030
00031
00032
00033
00034
00035
00036
00037
00038
00039
00040
00041
00042
00043
           /* Read individual GPS-RO data files... */
           for (iarg = 3; iarg < argc; iarg++)
  if (!(in = fopen(argv[iarg], "r")))</pre>
00044
00045
00046
                continue;
00047
00048
               fclose(in);
00049
                 read_gps_prof(argv[iarg], gps);
00050
00051
00052
           /* Check number of profiles... */
           if (gps->nds <= 0)</pre>
00053
00054
             ERRMSG("No profiles found!");
00055
00056
           /\star Grid profile... \star/
           if (grid_nz > 0)
00057
00058
             grid_gps(gps, grid_zmin, grid_zmax, grid_nz);
00059
00060
            /* Get tropopause... */
00061
           tropopause(gps);
00062
00063
           /* Get perturbations from horizontal Gaussian mean... */
00064
           if (gauss dx > 0 \&\& gauss dy > 0)
00065
             gauss(gps, gauss_dx, gauss_dy);
00066
00067
           /* Get perturbations from vertical Hamming filter... */
00068
           if (ham_dz > 0)
00069
             hamming_low_pass(gps, ham_dz);
00070
00071
           /* Use vertical Hamming filter to reduce noise... */
00072
           if (ham_dz2 > 0)
00073
             hamming_high_pass(gps, ham_dz2);
00074
00075
           /* Use meteo data for detrending... */
if (metbase[0] != '-')
00076
             detrend_met(gps, metbase, dt_met);
00077
00078
00079
           /* Remove polynomial fit from perturbation profile... */
08000
           if (poly_dim > 0)
00081
             poly(gps, poly_dim, poly_zmin, poly_zmax);
00082
00083
           /* Write GPS-RO data file... */
00084
          write_gps(argv[2], gps);
00085
00086
           /* Free... */
00087
          free(gps);
00088
00089
           return EXIT_SUCCESS;
00090 }
```

5.14 perturbation.c 277

Here is the call graph for this function:



5.14 perturbation.c

```
00001 #include "libgps.h"
00002
00003 /*
00004
                Main...
00005
00006
00007 int main(
80000
              int argc,
00009
              char *argv[]) {
00010
00011
              gps_t *gps;
00012
00013
              FILE *in;
00014
00015
               char metbase[LEN];
00016
00017
               double dt_met, gauss_dx, gauss_dy, grid_zmin, grid_zmax, ham_dz, ham_dz2,
00018
                 poly_zmax, poly_zmin;
00019
00020
              int grid_nz, iarg, poly_dim;
00021
00022
                /* Allocate... */
00023
               ALLOC(gps, gps_t, 1);
00024
00025
               /\star Check arguments... \star/
00026
               if (argc < 4)
00027
                  ERRMSG("Give parameters: <ctl> <out.nc> <gps1.nc> [<gps2.nc> ...]");
00028
00029
              /* Get control parameters... */
dt_met = scan_ctl(argc, argv, "DT_MET", -1, "21600", NULL);
gauss_dx = scan_ctl(argc, argv, "GAUSS_DX", -1, "-999", NULL);
gauss_dy = scan_ctl(argc, argv, "GAUSS_DY", -1, "-999", NULL);
grid_zmin = scan_ctl(argc, argv, "GRID_ZMIN", -1, "0", NULL);
grid_zmax = scan_ctl(argc, argv, "GRID_ZMAX", -1, "40", NULL);
grid_nz = (int) scan_ctl(argc, argv, "GRID_MZ", -1, "-1", NULL);
ham_dz = scan_ctl(argc, argv, "HAM_DZ", -1, "-999", NULL);
ham_dz2 = scan_ctl(argc, argv, "HAM_DZ", -1, "-999", NULL);
scan_ctl(argc, argv, "METBASE", -1, "", metbase);
poly_dim = (int) scan_ctl(argc, argv, "POLY_DIM", -1, "5", NULL);
poly_zmin = scan_ctl(argc, argv, "POLY_ZMIN", -1, "0", NULL);
poly_zmax = scan_ctl(argc, argv, "POLY_ZMAX", -1, "40", NULL);
               /* Get control parameters... */
00030
00031
00032
00033
00034
00035
00036
00037
00038
00039
00040
00041
00042
00043
                /* Read individual GPS-RO data files... */
00044
               for (iarg = 3; iarg < argc; iarg++)
  if (!(in = fopen(argv[iarg], "r")))</pre>
00045
                       continue;
```

```
00047
          else {
00048
          fclose(in);
00049
            read_gps_prof(argv[iarg], gps);
00050
00051
00052
        /* Check number of profiles... */
00053
        if (gps->nds <= 0)
00054
          ERRMSG("No profiles found!");
00055
        /* Grid profile... */
if (grid_nz > 0)
00056
00057
         grid_gps(gps, grid_zmin, grid_zmax, grid_nz);
00058
00059
00060
        /* Get tropopause... */
00061
        tropopause(gps);
00062
        /\star Get perturbations from horizontal Gaussian mean... \star/
00063
00064
        if (gauss_dx > 0 && gauss_dy > 0)
00065
          gauss(gps, gauss_dx, gauss_dy);
00066
00067
        /* Get perturbations from vertical Hamming filter... */
00068
        if (ham_dz > 0)
00069
         hamming_low_pass(gps, ham_dz);
00070
00071
        /* Use vertical Hamming filter to reduce noise... */
00072
        if (ham_dz2 > 0)
00073
          hamming_high_pass(gps, ham_dz2);
00074
        /* Use meteo data for detrending... */ if (metbase[0] != '-')
00075
00076
          detrend_met(gps, metbase, dt_met);
00077
00078
00079
        /\star Remove polynomial fit from perturbation profile... \star/
08000
        if (poly_dim > 0)
00081
         poly(gps, poly_dim, poly_zmin, poly_zmax);
00082
00083
        /* Write GPS-RO data file... */
00084
        write_gps(argv[2], gps);
00085
00086
        /* Free... */
00087
        free(gps);
00088
00089
        return EXIT_SUCCESS;
00090 }
```

5.15 prof.c File Reference

Functions

• int main (int argc, char *argv[])

5.15.1 Function Documentation

5.15.1.1 int main (int argc, char * argv[])

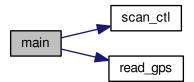
Definition at line 3 of file prof.c.

```
00005
00006
00007
        gps_t *gps;
00008
00009
        FILE *out;
00010
00011
        int ids, iz;
00012
00013
        /* Allocate... */
        ALLOC(gps, gps_t, 1);
00014
00015
00016
        /* Check arguments... */
        if (argc < 4)
00017
00018
         ERRMSG("Give parameters: <ctl> <gps.nc> <prof.tab>");
00019
00020
       /* Get control parameters... */
       ids = (int) scan_ctl(argc, argv, "IDS", -1, "0", NULL);
```

5.16 prof.c 279

```
00022
00023
          /* Read gps data... */
00024
          read_gps(argv[2], gps);
00025
         /* Check profile number... */
if (ids < 0 || ids > gps->nds)
    ERRMSG("Profile not available!");
00026
00027
00028
00029
00030
          /* Create output file... */
          printf("Write profile data: %s\n", argv[3]);
if (!(out = fopen(argv[3], "w")))
    ERRMSG("Cannot create file!");
00031
00032
00033
00034
00035
          /* Write header... */
00036
          fprintf(out,
                    "# $1 = time [sec]\n"
"# $2 = altitude [km]\n"
"# $3 = longitude [deg]\n"
"# $4 = latitude [deg]\n"
00037
00038
00039
00040
00041
                    "# $5 = pressure [hPa]\n"
00042
                    "# $6 = temperature [K] \n"
                    "# $7 = water vapor vmr [ppm]\n"
"# $8 = temperature perturbation [K]\n"
00043
00044
                    "# $9 = tropopause height [km]\n\n");
00045
00046
00047
          /* Write data... */
for (iz = 0; iz < gps->nz[ids]; iz++)
00048
           00049
00050
00051
00052
00053
00054
00055
          /* Close file... */
00056
         fclose(out);
00057
00058
          /* Free... */
         free(gps);
00060
00061
         return EXIT_SUCCESS;
00062 }
```

Here is the call graph for this function:



5.16 prof.c

```
00001 #include "libgps.h"
00002
00003 int main(
00004
        int argc,
00005
       char *argv[]) {
00006
00007
       qps_t *qps;
80000
00009
       FILE *out;
00010
00011
        int ids, iz;
00012
00013
        /* Allocate... */
00014
       ALLOC(gps, gps_t, 1);
```

```
00015
00016
        /* Check arguments... */
00017
        if (argc < 4)
          ERRMSG("Give parameters: <ctl> <gps.nc> <prof.tab>");
00018
00019
00020
        /* Get control parameters... */
        ids = (int) scan_ctl(argc, argv, "IDS", -1, "0", NULL);
00021
00022
00023
        /* Read gps data... */
00024
        read_gps(argv[2], gps);
00025
        /* Check profile number... */
if (ids < 0 || ids > gps->nds)
    ERRMSG("Profile not available!");
00026
00027
00028
00029
00030
        /* Create output file... */
        printf("Write profile data: %s\n", argv[3]);
if (!(out = fopen(argv[3], "w")))
    ERRMSG("Cannot create file!");
00031
00032
00033
00034
00035
        /* Write header... */
        00036
00037
                 "# $2 = altitude [km] \n"
00038
00039
                 "# $3 = longitude [deg]\n"
00040
                 "# $4 = latitude [deg]\n"
00041
                 "# $5 = pressure [hPa] \n"
00042
                 "# $6 = temperature [K] \n"
                 "# $7 = water vapor vmr [ppm]\n"
00043
                 "# $8 = temperature perturbation [K] \n"
00044
00045
                 "# $9 = tropopause height [km]\n\n");
00046
00047
        /* Write data... */
00048
        for (iz = 0; iz < gps->nz[ids]; iz++)
         00049
00050
00051
00052
00053
00054
00055
        /* Close file... */
00056
       fclose(out);
00057
00058
        /* Free... */
00059
        free(gps);
00060
00061
        return EXIT_SUCCESS;
00062 }
```

5.17 response.c File Reference

Functions

int main (int argc, char *argv[])

5.17.1 Function Documentation

5.17.1.1 int main (int argc, char * argv[])

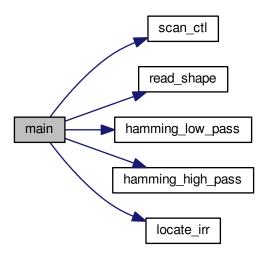
Definition at line 7 of file response.c.

```
00009
00010
00011
        gps_t *gps;
00012
00013
       FILE *out;
00014
00015
        double 1z, ptmax[NZ], var[NZ], w, wmax, se[NZ], sz[NZ], t0 = 10.0,
00016
         grid_zmin, grid_zmax, ham_dz, ham_dz2;
00017
00018
       int idx, iphi, iz, iz2, nphi = 360, sn = 0, grid_nz;
00019
00020
       /* Allocate... */
00021
       ALLOC(gps, gps_t, 1);
```

```
00022
00023
          /* Check arguments... */
00024
         if (argc < 4)
            ERRMSG("Give parameters: <ctl> <sens.tab> <response.tab>");
00025
00026
00027
          /\star Get control parameters... \star/
         /* Get Control parameters... */
grid_zmin = scan_ctl(argc, argv, "GRID_ZMIN", -1, "0", NULL);
grid_zmax = scan_ctl(argc, argv, "GRID_ZMAX", -1, "60", NULL);
grid_nz = (int) scan_ctl(argc, argv, "GRID_NZ", -1, "601", NULL);
ham_dz = scan_ctl(argc, argv, "HAM_DZ", -1, "6.0", NULL);
ham_dz2 = scan_ctl(argc, argv, "HAM_DZ2", -1, "0.4", NULL);
00028
00029
00030
00031
00032
00033
00034
          /* Read vertical sensitivity function... */
00035
         if (argv[2][0] != '-') {
00036
            read_shape(argv[2], sz, se, &sn);
00037
            if (sn > NZ)
               ERRMSG("Too many data points!");
00038
00039
00040
00041
          /* Create output file... */
00042
         printf("Write response data: %s\n", argv[3]);
          if (!(out = fopen(argv[3], "w")))
    ERRMSG("Cannot create file!");
00043
00044
00045
00046
          /* Write header... */
00047
         fprintf(out,
                    "# $1 = vertical wavelength [km]\n"
00048
                    "# $2 = altitude [km] \n"
00049
                    "# $3 = response (amplitude) [%%] \n"
00050
                    "# $4 = response (variance) [%%] \n");
00051
00052
00053
         /* Create profile... */
         gps->nds = 1;
gps->nz[0] = grid_nz;
00054
00055
00056
         for (iz = 0; iz < gps->nz[0]; iz++)
            gps->z[0][iz] =
00057
00058
              LIN(0.0, grid_zmin, grid_nz - 1.0, grid_zmax, (double) iz);
00059
00060
          /* Loop over vertical wavelength...
00061
         for (lz = 0.1; lz <= 20.0; lz += 0.1) {
00062
00063
            /* Write info... */
printf("Calculate %g km...\n", lz);
00064
00065
            /* Initialize... */
for (iz = 0; iz < gps->nz[0]; iz++)
00066
00067
00068
              ptmax[iz] = var[iz] = 0;
00069
            /* Loop over phase... */
for (iphi = 0; iphi < nphi; iphi++) {</pre>
00070
00071
00072
               /* Create profile... */
for (iz = 0; iz < gps->nz[0]; iz++)
00073
00074
00075
                 gps \rightarrow t[0][iz] = 250.0 + t0 * sin(2. * M_PI / 1z * gps \rightarrow z[0][iz]
00076
00077
                                                           2. * M_PI * (double) iphi /
00078
                                                           (double) nphi);
00079
00080
               /* Get perturbations from vertical Hamming filter... */
00081
               if (ham_dz > 0)
00082
                 hamming_low_pass(gps, ham_dz);
00083
00084
               /* Use vertical Hamming filter to reduce noise... */
00085
               if (ham_dz2 > 0)
00086
                 hamming_high_pass(gps, ham_dz2);
00087
00088
               /* Multiply with vertical sensitivity function... */ if (argv[2][0] != '-') {
00089
00090
                 wmax = 0;
00091
                 for (iz2 = 0; iz2 < gps->nz[0]; iz2++) {
00092
                   if (gps->z[0][iz2] < sz[0] || gps->z[0][iz2] > sz[sn - 1])
00093
                      w = 0;
00094
                    else {
                      idx = locate_irr(sz, sn, gps->z[0][iz2]);
00095
00096
00097
                        LIN(sz[idx], se[idx], sz[idx + 1], se[idx + 1], gps \rightarrow z[0][iz2]);
00098
00099
                    gps->pt[0][iz2] *= w;
00100
                    wmax = GSL\_MAX(w, wmax);
00101
                 if (wmax > 0)
  for (iz2 = 0; iz2 < gps->nz[0]; iz2++)
00102
00103
00104
                      gps->pt[0][iz2] /= wmax;
00105
00106
              /* Get response... */
for (iz = 0; iz < gps->nz[0]; iz++) {
00107
00108
```

```
ptmax[iz] = GSL_MAX(ptmax[iz], gps->pt[0][iz]);
var[iz] += gsl_pow_2(gps->pt[0][iz]) / nphi;
00110
00111
         }
00112
00113
         00114
00115
00116
00117
00118
00119
00120
00121
       /* Close file... */
00122
       fclose(out);
00123
00124
       /* Free... */
00125
       free(gps);
00126
00127
       return EXIT_SUCCESS;
00128 }
```

Here is the call graph for this function:



5.18 response.c

```
00001 #include "libgps.h"
00002
00003 /* -----
00004
        Main...
00005
00006
00007 int main(
80000
       int argc,
00009
       char *argv[]) {
00010
00011
        gps_t *gps;
00012
00013
       FILE *out;
00014
00015
        double 1z, ptmax[NZ], var[NZ], w, wmax, se[NZ], sz[NZ], t0 = 10.0,
00016
          grid_zmin, grid_zmax, ham_dz, ham_dz2;
00017
00018
       int idx, iphi, iz, iz2, nphi = 360, sn = 0, grid_nz;
00019
00020
       /* Allocate... */
       ALLOC(gps, gps_t, 1);
00021
```

5.18 response.c 283

```
00022
00023
          /* Check arguments... */
00024
          if (argc < 4)
            ERRMSG("Give parameters: <ctl> <sens.tab> <response.tab>");
00025
00026
00027
          /\star Get control parameters... \star/
         /* Get Control parameters... */
grid_zmin = scan_ctl(argc, argv, "GRID_ZMIN", -1, "0", NULL);
grid_zmax = scan_ctl(argc, argv, "GRID_ZMAX", -1, "60", NULL);
grid_nz = (int) scan_ctl(argc, argv, "GRID_NZ", -1, "601", NULL);
ham_dz = scan_ctl(argc, argv, "HAM_DZ", -1, "6.0", NULL);
ham_dz2 = scan_ctl(argc, argv, "HAM_DZ2", -1, "0.4", NULL);
00028
00029
00030
00031
00032
00033
00034
          /* Read vertical sensitivity function... */
00035
          if (argv[2][0] != '-') {
00036
            read_shape(argv[2], sz, se, &sn);
00037
            if (sn > NZ)
               ERRMSG("Too many data points!");
00038
00039
00040
00041
          /* Create output file... */
00042
          printf("Write response data: %s\n", argv[3]);
          if (!(out = fopen(argv[3], "w")))
    ERRMSG("Cannot create file!");
00043
00044
00045
00046
          /* Write header... */
00047
          fprintf(out,
                    "# $1 = vertical wavelength [km]\n"
00048
                    "# $2 = altitude [km] \n"
00049
                    "# $3 = response (amplitude) [%%] \n"
00050
                    "# $4 = response (variance) [%%]\n");
00051
00052
00053
          /* Create profile... */
         gps->nds = 1;
gps->nz[0] = grid_nz;
00054
00055
00056
          for (iz = 0; iz < gps->nz[0]; iz++)
            gps->z[0][iz] =
00057
00058
               LIN(0.0, grid_zmin, grid_nz - 1.0, grid_zmax, (double) iz);
00059
00060
          /* Loop over vertical wavelength...
00061
          for (lz = 0.1; lz <= 20.0; lz += 0.1) {
00062
00063
            /* Write info... */
printf("Calculate %g km...\n", lz);
00064
00065
            /* Initialize... */
for (iz = 0; iz < gps->nz[0]; iz++)
00066
00067
00068
              ptmax[iz] = var[iz] = 0;
00069
00070
            /* Loop over phase... */
for (iphi = 0; iphi < nphi; iphi++) {</pre>
00071
00072
               /* Create profile... */
for (iz = 0; iz < gps->nz[0]; iz++)
   gps->t[0][iz] = 250.0 + t0 * sin(2. * M_PI / lz * gps->z[0][iz]
00073
00074
00075
00076
00077
                                                            2. * M_PI * (double) iphi /
00078
                                                            (double) nphi);
00079
00080
               /* Get perturbations from vertical Hamming filter... */
00081
               if (ham_dz > 0)
00082
                 hamming_low_pass(gps, ham_dz);
00083
00084
               /* Use vertical Hamming filter to reduce noise... */
00085
               if (ham_dz2 > 0)
00086
                 hamming_high_pass(gps, ham_dz2);
00087
00088
               /* Multiply with vertical sensitivity function... */ if (argv[2][0] != '-') {
00089
00090
                 wmax = 0;
00091
                  for (iz2 = 0; iz2 < gps->nz[0]; iz2++) {
00092
                   if (gps->z[0][iz2] < sz[0] || gps->z[0][iz2] > sz[sn - 1])
00093
                      w = 0;
00094
                    else {
00095
                      idx = locate_irr(sz, sn, qps->z[0][iz2]);
00096
00097
                        LIN(sz[idx], se[idx], sz[idx + 1], se[idx + 1], gps->z[0][iz2]);
00098
00099
                    gps->pt[0][iz2] *= w;
00100
                    wmax = GSL\_MAX(w, wmax);
00101
                 if (wmax > 0)
  for (iz2 = 0; iz2 < gps->nz[0]; iz2++)
00102
00103
00104
                      gps->pt[0][iz2] /= wmax;
00105
00106
               /* Get response... */
for (iz = 0; iz < gps->nz[0]; iz++) {
00107
00108
```

```
ptmax[iz] = GSL_MAX(ptmax[iz], gps->pt[0][iz]);
00110
           var[iz] += gsl_pow_2(gps->pt[0][iz]) / nphi;
00111
         }
00112
        }
00113
        /* Write output... */
00114
        fprintf(out, "\n");
00115
        00116
00117
00118
00119
00120
00121
      /* Close file... */
00122
      fclose(out);
00123
      /* Free... */
00124
00125
      free(gps);
00126
      return EXIT_SUCCESS;
00128 }
```

5.19 variance.c File Reference

Functions

• int main (int argc, char *argv[])

5.19.1 Function Documentation

5.19.1.1 int main (int argc, char * argv[])

Definition at line 20 of file variance.c.

```
00022
00023
00024
            gps_t *gps;
00025
00026
            FILE *in, *out;
00027
00028
             static double lon0, lon1, lat0, lat1, z0, z1, mean[GX][GY][GZ],
               min[GX][GY][GZ], max[GX][GY][GZ], var[GX][GY][GZ],
mtime[GX][GY], glon[GX], glat[GY], gz[GZ], thmean[GX][GY],
tmean[GX][GY][GZ], twmean[GX][GY], se[NZ], sz[NZ], tw, w, wmax, wsum;
00029
00030
00031
00032
00033
             static int iarg, ids, idx, ix, iy, iz, iz2,
00034
                nx, ny, nz, n[GX][GY][GZ], np[GX][GY], sn;
00035
00036
              /* Allocate... */
00037
             ALLOC(gps, gps_t, 1);
00038
00039
              /* Check arguments... */
00040
             if (argc < 5)
00041
                ERRMSG("Give parameters: <ctl> <var.tab> <sens.tab> "
00042
                             "<qps1.nc> [<qps2.nx> ...]");
00043
            /* Get control parameters... */
z0 = scan_ctl(argc, argv, "20", -1, "0", NULL);
z1 = scan_ctl(argc, argv, "21", -1, "60", NULL);
nz = (int) scan_ctl(argc, argv, "NZ", -1, "6", NULL);
lon0 = scan_ctl(argc, argv, "LON0", -1, "-180", NULL);
lon1 = scan_ctl(argc, argv, "LON1", -1, "180", NULL);
nx = (int) scan_ctl(argc, argv, "NX", -1, "72", NULL);
lat0 = scan_ctl(argc, argv, "LAT0", -1, "-90", NULL);
lat1 = scan_ctl(argc, argv, "LAT1", -1, "90", NULL);
ny = (int) scan ctl(argc, argv, "NY", -1, "36", NULL);
00044
00045
00046
00047
00048
00050
00051
00052
            ny = (int) scan_ctl(argc, argv, "NY", -1, "36", NULL);
00053
00054
00055
             /* Check grid dimensions... */
00056
            if (nx < 1 \mid \mid nx > GX)
00057
                ERRMSG("Set 1 <= GX <= MAX!");</pre>
            if (ny < 1 || ny > GY)
   ERRMSG("Set 1 <= GY <= MAX!");
if (nz < 1 || nz > GZ)
00058
00059
00060
00061
                ERRMSG("Set 1 <= GZ <= MAX!");</pre>
00062
```

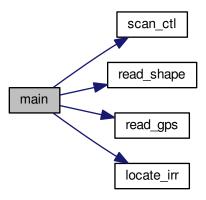
```
/* Read vertical sensitivity function... */
00064
        if (argv[3][0] != '-') {
00065
          read_shape(argv[3], sz, se, &sn);
00066
          if (sn > NZ)
00067
            ERRMSG("Too many data points!");
00068
00069
00070
        /\star Loop over data files... \star/
00071
        for (iarg = 4; iarg < argc; iarg++) {</pre>
00072
00073
          /* Read gps data... */
00074
          if (!(in = fopen(argv[iarg], "r")))
00075
            continue;
00076
00077
            fclose(in);
00078
            read_gps(argv[iarg], gps);
00079
00080
00081
          /* Loop over profiles... */
00082
          for (ids = 0; ids < gps->nds; ids++) {
00083
00084
            /* Check tropopause height...
00085
            if (!gsl_finite(gps->th[ids]))
00086
              continue;
00087
00088
            /* Multiply with vertical sensitivity function... */
00089
            if (argv[3][0] != '-')
00090
              tw = wsum = wmax = 0;
              for (iz2 = 0; iz2 < gps->nz[ids]; iz2++) {
00091
00092
                if (gps->z[ids][iz2] < sz[0] || gps->z[ids][iz2] > sz[sn - 1])
00093
                  w = 0:
00094
                else {
00095
                 idx = locate_irr(sz, sn, gps->z[ids][iz2]);
00096
00097
                    LIN(sz[idx], se[idx], sz[idx + 1], se[idx + 1],
                         gps->z[ids][iz2]);
00098
00099
00100
                if (gsl_finite(gps->t[ids][iz2]) && gps->pt[ids][iz2]) {
00101
                  tw += w * gps->t[ids][iz2];
00102
                  wsum += w;
00103
                   gps \rightarrow pt[ids][iz2] *= w;
00104
                  wmax = GSL_MAX(w, wmax);
00105
                }
00106
00107
              tw /= wsum;
00108
              for (iz2 = 0; iz2 < gps->nz[ids]; iz2++)
00109
                gps->pt[ids][iz2] /= wmax;
00110
00111
00112
            /* Get grid indices... */
            ix = (int) ((gps->lon[ids][gps->nz[ids] / 2] - lon0)
00113
00114
                          (lon1 - lon0) * (double) nx);
00115
            iy = (int) ((gps->lat[ids][gps->nz[ids] / 2] - lat0)
            / (lat1 - lat0) * (double) ny);
if (ix < 0 || ix >= nx || iy < 0 || iy >= ny)
00116
00117
00118
              continue;
00120
            /* Get mean time and tropopause height... */
00121
            mtime[ix][iy] += gps->time[ids];
            thmean[ix][iy] += gps->th[ids];
twmean[ix][iy] += tw;
00122
00123
00124
            np[ix][iy]++;
00125
00126
            /* Loop over altitudes... */
00127
            for (iz2 = 0; iz2 < gps->nz[ids]; iz2++) {
00128
00129
              /* Get grid indices... */
              iz = (int) ((gps->z[ids][iz2] - z0)
00130
                           / (z1 - z0) * (double) nz);
00131
              if (iz < 0 || iz >= nz)
00132
00133
                continue;
00134
00135
              /* Check data... */
00136
              00137
                continue;
00138
00139
              /* Get statistics of perturbations... */
00140
              tmean[ix][iy][iz] += gps->t[ids][iz2];
00141
              mean[ix][iy][iz] += gps->pt[ids][iz2];
              medn[ix][iy][iz] += gsl_pow_2(gps->pt[ids][iz2]);
max[ix][iy][iz] = GSL_MAX(max[ix][iy][iz], gps->pt[ids][iz2]);
00142
00143
              min[ix][iy][iz] = GSL_MIN(min[ix][iy][iz], gps->pt[ids][iz2]);
00144
00145
              n[ix][iy][iz]++;
00146
00147
         }
       }
00148
00149
```

```
/* Analyze results... */
00151
        for (ix = 0; ix < nx; ix++)
00152
           for (iy = 0; iy < ny; iy++) {</pre>
00153
00154
              /* Get mean time and tropopause height... */
00155
              if (np[ix][iy] > 0) {
               mtime[ix][iy] /= (double) np[ix][iy];
00156
00157
                thmean[ix][iy] /= (double) np[ix][iy];
                twmean[ix][iy] /= (double) np[ix][iy];
00158
              } else {
00159
               mtime[ix][iy] = GSL_NAN;
00160
               thmean[ix][iy] = GSL_NAN;
twmean[ix][iy] = GSL_NAN;
00161
00162
00163
00164
00165
              /* Loop over altitudes... */
             for (iz = 0; iz < nz; iz++) {
00166
00167
00168
                /* Get geolocation... */
                gz[iz] = z0 + (iz + 0.5) / (double) nz *(
00169
00170
        z1 - z0);
00171
               glon[ix] = lon0 + (ix + 0.5) / (double) nx *(
        lon1 - lon0);
00172
        \label{eq:glat_glat} \begin{split} &\text{glat[iy] = lat0 + (iy + 0.5) / (double) ny *(lat1 - lat0);} \end{split}
00173
00174
00175
00176
                /\star Get mean perturbation and variance... \star/
00177
                if (n[ix][iy][iz] > 0) {
00178
                  tmean[ix][iy][iz]
00179
                    /= (double) n[ix][iy][iz];
00180
                  mean[ix][iy][iz]
00181
                    /= (double) n[ix][iy][iz];
00182
                  var[ix][iy][iz]
00183
                    = var[ix][iy][iz] / (double) n[ix][iy][iz]
                    - gsl_pow_2(mean[ix][iy][iz]);
00184
00185
                } else {
                 tmean[ix][iy][iz] = GSL_NAN;
mean[ix][iy][iz] = GSL_NAN;
00186
                  var[ix][iy][iz] = GSL_NAN;
min[ix][iy][iz] = GSL_NAN;
00188
00189
                  max[ix][iy][iz] = GSL_NAN;
00190
00191
00192
             }
00193
00194
00195
         /* Create file... */
00196
        printf("Write variance statistics: %s\n", argv[2]);
00197
         if (!(out = fopen(argv[2], "w")))
           ERRMSG("Cannot create file!");
00198
00199
00200
         /* Write header... */
00201
        fprintf(out,
00202
                  "# $1 = time [s] \n"
                  "# $2 = altitude [km]\n"
"# $3 = longitude [deg]\n"
00203
00204
00205
                  "# $4 = latitude [deg] \n"
                  "# $5 = number of profiles\n"
00206
00207
                  "# $6 = number of data points\n"
00208
                  "# $7 = mean perturbation [K] \n"
                  "# $8 = minimum perturbation [K]\n"
00209
                  "# \$9 = \text{maximum perturbation [K]} \n"
00210
                  "# $10 = variance [K^2]\n"
00211
00212
                  "# $11 = mean temperature [K]\n"
00213
                  "# $12 = mean weighted temperature [K] \n"
00214
                  "# $13 = mean tropopause height [km]\n");
00215
00216
        /* Write results... */
00217
        for (iz = 0; iz < nz; iz++) {
          for (iy = 0; iy < ny; iy++) {
00218
             if (iy == 0 || nx > 1)
00219
00220
                fprintf(out, "\n");
             for (ix = 0; ix < nx; ix++)
fprintf(out, "%.2f %g %g %g %d %d %g %g %g %g %g %g %g %g",</pre>
00221
00222
                         ntime[ix][iy], gz[iz], glon[ix], glat[iy],
np[ix][iy], n[ix][iy][iz], mean[ix][iy][iz],
min[ix][iy][iz], max[ix][iy][iz], var[ix][iy][iz],
00223
00224
00225
00226
                         tmean[ix][iy][iz], twmean[ix][iy], thmean[ix][iy]);
00227
00228
00229
        /* Close file... */
00230
00231
        fclose(out);
00232
00233
         /* Free... */
00234
        free(gps);
00235
00236
        return EXIT_SUCCESS;
```

5.20 variance.c 287

```
00237 }
```

Here is the call graph for this function:



5.20 variance.c

```
00001 #include "libgps.h"
00002
00003 /* -----
         Dimensions...
00004
00005
00006
00007 /* Maximum number of longitudes. */
00008 #define GX 360
00009
00010 /\star Maximum number of latitudes. \star/
00011 #define GY 180
00012
00013 /* Maximum number of altitudes. */
00014 #define GZ 50
00015
00016 /* ---
00017
          Main...
00018
00019
00020 int main(
00021
         int argc,
00022
         char *argv[]) {
00023
00024
         gps_t *gps;
00025
00026
         FILE *in, *out;
00027
00028
         static double lon0, lon1, lat0, lat1, z0, z1, mean[GX][GY][GZ],
          min[GX][GY][GZ], max[GX][GY][GZ], var[GX][GY][GZ], mtime[GX][GY], glon[GX], glat[GY], gz[GZ], thmean[GX][GY], tmean[GX][GY], twmean[GX][GY], se[NZ], sz[NZ], tw, w, wmax, wsum;
00029
00030
00031
00032
00033
         static int iarg, ids, idx, ix, iy, iz, iz2,
00034
           nx, ny, nz, n[GX][GY][GZ], np[GX][GY], sn;
00035
00036
          /* Allocate... */
00037
         ALLOC(gps, gps_t, 1);
00038
00039
          /* Check arguments... */
         if (argc < 5)
    ERRMSG("Give parameters: <ctl> <var.tab> <sens.tab> "
    "<gps1.nc> [<gps2.nx> ...]");
00040
00041
00042
00043
         /* Get control parameters... */
z0 = scan_ctl(argc, argv, "Z0", -1, "0", NULL);
00044
00045
```

```
z1 = scan_ctl(argc, argv, "Z1", -1, "60", NULL);
nz = (int) scan_ctl(argc, argv, "NZ", -1, "6", NULL);
lon0 = scan_ctl(argc, argv, "LON0", -1, "-180", NULL);
lon1 = scan_ctl(argc, argv, "LON1", -1, "180", NULL);
nx = (int) scan_ctl(argc, argv, "NX", -1, "72", NULL);
lat0 = scan_ctl(argc, argv, "LAT1", -1, "90", NULL);
lat1 = scan_ctl(argc, argv, "LAT1", -1, "90", NULL);
00047
00048
00049
00050
00051
00053
         ny = (int) scan_ctl(argc, argv, "NY", -1, "36", NULL);
00054
00055
         /\star Check grid dimensions... \star/
00056
         if (nx < 1 \mid \mid nx > GX)
           ERRMSG("Set 1 <= GX <= MAX!");</pre>
00057
           f (ny < 1 || ny > GY)
ERRMSG("Set 1 <= GY <= MAX!");
00058
00059
00060
         if (nz < 1 || nz > GZ)
00061
           ERRMSG("Set 1 <= GZ <= MAX!");</pre>
00062
00063
         /* Read vertical sensitivity function... */
         if (argv[3][0] != '-') {
00064
00065
           read_shape(argv[3], sz, se, &sn);
00066
           if (sn > NZ)
00067
              ERRMSG("Too many data points!");
00068
00069
00070
          /* Loop over data files... */
00071
         for (iarg = 4; iarg < argc; iarg++) {</pre>
00072
00073
            /* Read gps data... */
00074
            if (!(in = fopen(argv[iarg], "r")))
00075
              continue;
00076
            else {
00077
              fclose(in);
00078
              read_gps(argv[iarg], gps);
00079
00080
            /* Loop over profiles... */
00081
00082
            for (ids = 0; ids < gps->nds; ids++) {
00084
               /* Check tropopause height...
00085
              if (!gsl_finite(gps->th[ids]))
00086
00087
00088
               /* Multiply with vertical sensitivity function... */
               if (argv[3][0] != '-') {
00089
                 tw = wsum = wmax = 0;
00090
00091
                 for (iz2 = 0; iz2 < gps->nz[ids]; iz2++) {
00092
                   if (gps->z[ids][iz2] < sz[0] || gps->z[ids][iz2] > sz[sn - 1])
00093
                     w = 0;
00094
                   else {
00095
                     idx = locate irr(sz, sn, qps->z[ids][iz2]);
00097
                        LIN(sz[idx], se[idx], sz[idx + 1], se[idx + 1],
00098
                             gps->z[ids][iz2]);
00099
                   if (gsl_finite(gps->t[ids][iz2]) && gps->pt[ids][iz2]) {
00100
                     tw += w * gps->t[ids][iz2];
00101
                      wsum += w;
00103
                      gps->pt[ids][iz2] *= w;
00104
                      wmax = GSL_MAX(w, wmax);
00105
                   }
00106
                 tw /= wsum;
for (iz2 = 0; iz2 < gps->nz[ids]; iz2++)
  gps->pt[ids][iz2] /= wmax;
00107
00108
00109
00110
00111
               /* Get grid indices... */
00112
              ix = (int) ((gps->lon[ids][gps->nz[ids] / 2] - lon0)
00113
00114
                             / (lon1 - lon0) * (double) nx);
              iy = (int) ((gps->lat[ids][gps->nz[ids] / 2] - lat0)
00115
00116
                               (lat1 - lat0) * (double) ny);
              if (ix < 0 || ix >= nx || iy < 0 || iy >= ny)
00117
00118
                 continue;
00119
00120
               /* Get mean time and tropopause height... */
              mtime[ix][iy] += gps->time[ids];
00121
              thmean[ix][iy] += gps->th[ids];
twmean[ix][iy] += tw;
00122
00123
00124
              np[ix][iy]++;
00125
00126
               /* Loop over altitudes... */
              for (iz2 = 0; iz2 < gps->nz[ids]; iz2++) {
00128
00129
                 /* Get grid indices... */
00130
                 iz = (int) ((gps->z[ids][iz2] - z0)
                 / (z1 - z0) * (double) nz);
if (iz < 0 \mid \mid iz >= nz)
00131
00132
```

5.20 variance.c 289

```
continue;
00134
00135
                 /* Check data... */
00136
                 00137
                   continue;
00138
                 /\star Get statistics of perturbations... \star/
00140
                 tmean[ix][iy][iz] += gps->t[ids][iz2];
00141
                 mean[ix][iy][iz] += gps->pt[ids][iz2];
                var[ix][iy][iz] += gsl_pow_2(gps->pt[ids][iz2]);
max[ix][iy][iz] = GSL_MAX(max[ix][iy][iz], gps->pt[ids][iz2]);
min[ix][iy][iz] = GSL_MIN(min[ix][iy][iz], gps->pt[ids][iz2]);
00142
00143
00144
00145
                n[ix][iy][iz]++;
00146
00147
00148
00149
00150
         /* Analyze results... */
         for (ix = 0; ix < nx; ix++)
           for (iy = 0; iy < ny; iy++) {</pre>
00153
00154
               /\star Get mean time and tropopause height... \star/
00155
               if (np[ix][iy] > 0) {
                time[ix][iy] /= (double) np[ix][iy];
thmean[ix][iy] /= (double) np[ix][iy];
twmean[ix][iy] /= (double) np[ix][iy];
00156
00157
00159
                mtime[ix][iy] = GSL_NAN;
thmean[ix][iy] = GSL_NAN;
00160
00161
                twmean[ix][iy] = GSL_NAN;
00162
00163
00164
00165
              /* Loop over altitudes... */
00166
              for (iz = 0; iz < nz; iz++) {</pre>
00167
                 /* Get geolocation... */ gz[iz] = z0 + (iz + 0.5) / (double) nz *(
00168
00169
         z1 - z0);
00171
                glon[ix] = lon0 + (ix + 0.5) / (double) nx *(
00172
         lon1 - lon0);
00173
         \texttt{glat[iy]} = \texttt{lat0} + (\texttt{iy} + \texttt{0.5}) \ / \ (\texttt{double}) \ \texttt{ny} \ \star (\texttt{lat1} - \texttt{lat0});
00174
00175
00176
                 /\star Get mean perturbation and variance... \star/
00177
                 if (n[ix][iy][iz] > 0) {
00178
                   tmean[ix][iy][iz]
00179
                     /= (double) n[ix][iy][iz];
00180
                   mean[ix][iy][iz]
00181
                     /= (double) n[ix][iy][iz];
00182
                   var[ix][iv][iz]
                     = var[ix][iy][iz] / (double) n[ix][iy][iz]

- gsl_pow_2(mean[ix][iy][iz]);
00183
00184
                 } else {
00185
00186
                   tmean[ix][iy][iz] = GSL_NAN;
                   mean[ix][iy][iz] = GSL_NAN;
00187
                   var[ix][iy][iz] = GSL_NAN;
min[ix][iy][iz] = GSL_NAN;
00188
00190
                   max[ix][iy][iz] = GSL_NAN;
00191
00192
              }
           }
00193
00194
00195
          /* Create file... */
00196
         printf("Write variance statistics: %s\n", argv[2]);
00197
          if (!(out = fopen(argv[2], "w")))
00198
           ERRMSG("Cannot create file!");
00199
00200
         /* Write header... */
00201
         fprintf(out,
                   "# $1 = time [s] \n"
00203
                   "# $2 = altitude [km] \n"
                   "# $3 = longitude [deg] \n"
00204
                   "# $4 = latitude [deg]\n"
00205
                   "# $5 = number of profiles\n"
"# $6 = number of data points\n"
00206
00207
                   "# $7 = mean perturbation [K]\n"
00208
00209
                   "# $8 = minimum perturbation [K]\n"
00210
                   "# $9 = maximum perturbation [K] \n"
                   "# $10 = variance [K^2] \n"
00211
                   "# $11 = mean temperature [K]\n"
00212
                   "# $12 = mean weighted temperature [K]\n"
00213
                   "# $13 = mean tropopause height [km]\n");
00215
00216
         /* Write results... */
         for (iz = 0; iz < nz; iz++) {
  for (iy = 0; iy < ny; iy++) {
    if (iy == 0 || nx > 1)
00217
00218
00219
```

```
00220
00221
00222
00223
00224
00225
00226
00227
00228
00229
      /* Close file... */
00231
     fclose(out);
00232
00233
      /* Free... */
00234
     free(gps);
00235
00236
      return EXIT_SUCCESS;
00237 }
```

Index

add_var	retq_zmin, 8
libgps.c, 228	rett_zmax, 8
libgps.h, 256	rett_zmin, 8
atm2x	tblbase, 7
jurassic.c, 26	window, 7
jurassic.h, 153	write_bbt, 9
atm2x_help	write matrix, 9
jurassic.c, 26	ctm co2
jurassic.h, 153	ctl_t, 7
atm_t, 3	ctm h2o
	-
k, 4	ctl_t, 7
lat, 4	ctm_n2
lon, 4	ctl_t, 7
np, 4	ctm_o2
p, 4	ctl_t, 7
q, 4	ctmco2
t, 4	jurassic.c, <mark>36</mark>
time, 4	jurassic.h, 164
z, 4	ctmh2o
	jurassic.c, 46
brightness	jurassic.h, 174
jurassic.c, 26	ctmn2
jurassic.h, 154	jurassic.c, 58
J	•
cart2geo	jurassic.h, 186
jurassic.c, 27	ctmo2
jurassic.h, 154	jurassic.c, 59
climatology	jurassic.h, 187
jurassic.c, 27	detrend_met
jurassic.h, 154	libgps.c, 229
jurassic.h, 154 copy_atm	-
jurassic.h, 154 copy_atm jurassic.c, 60	libgps.c, 229
jurassic.h, 154 copy_atm jurassic.c, 60 jurassic.h, 188	libgps.c, 229 libgps.h, 256
jurassic.h, 154 copy_atm jurassic.c, 60 jurassic.h, 188 copy_obs	libgps.c, 229 libgps.h, 256 ds
jurassic.h, 154 copy_atm jurassic.c, 60 jurassic.h, 188	libgps.c, 229 libgps.h, 256 ds
jurassic.h, 154 copy_atm jurassic.c, 60 jurassic.h, 188 copy_obs	libgps.c, 229 libgps.h, 256 ds los_t, 13
jurassic.h, 154 copy_atm jurassic.c, 60 jurassic.h, 188 copy_obs jurassic.c, 61	libgps.c, 229 libgps.h, 256 ds los_t, 13 emitter
jurassic.h, 154 copy_atm jurassic.c, 60 jurassic.h, 188 copy_obs jurassic.c, 61 jurassic.h, 188	libgps.c, 229 libgps.h, 256 ds los_t, 13 emitter ctl_t, 6 eps
jurassic.h, 154 copy_atm jurassic.c, 60 jurassic.h, 188 copy_obs jurassic.c, 61 jurassic.h, 188 ctl_t, 5	libgps.c, 229 libgps.h, 256 ds los_t, 13 emitter ctl_t, 6 eps tbl_t, 19
jurassic.h, 154 copy_atm jurassic.c, 60 jurassic.h, 188 copy_obs jurassic.c, 61 jurassic.h, 188 ctl_t, 5 ctm_co2, 7	libgps.c, 229 libgps.h, 256 ds los_t, 13 emitter ctl_t, 6 eps tbl_t, 19 events.c, 20
jurassic.h, 154 copy_atm jurassic.c, 60 jurassic.h, 188 copy_obs jurassic.c, 61 jurassic.h, 188 ctl_t, 5 ctm_co2, 7 ctm_h2o, 7 ctm_n2, 7	libgps.c, 229 libgps.h, 256 ds los_t, 13 emitter ctl_t, 6 eps tbl_t, 19
jurassic.h, 154 copy_atm jurassic.c, 60 jurassic.h, 188 copy_obs jurassic.c, 61 jurassic.h, 188 ctl_t, 5 ctm_co2, 7 ctm_h2o, 7 ctm_n2, 7 ctm_o2, 7	libgps.c, 229 libgps.h, 256 ds los_t, 13 emitter ctl_t, 6 eps tbl_t, 19 events.c, 20 main, 20
jurassic.h, 154 copy_atm jurassic.c, 60 jurassic.h, 188 copy_obs jurassic.c, 61 jurassic.h, 188 ctl_t, 5 ctm_co2, 7 ctm_h2o, 7 ctm_n2, 7 ctm_o2, 7 emitter, 6	libgps.c, 229 libgps.h, 256 ds los_t, 13 emitter ctl_t, 6 eps tbl_t, 19 events.c, 20 main, 20 find_emitter
jurassic.h, 154 copy_atm jurassic.c, 60 jurassic.h, 188 copy_obs jurassic.c, 61 jurassic.h, 188 ctl_t, 5 ctm_co2, 7 ctm_h2o, 7 ctm_n2, 7 ctm_o2, 7 emitter, 6 fov, 8	libgps.c, 229 libgps.h, 256 ds los_t, 13 emitter ctl_t, 6 eps tbl_t, 19 events.c, 20 main, 20 find_emitter jurassic.c, 61
jurassic.h, 154 copy_atm jurassic.c, 60 jurassic.h, 188 copy_obs jurassic.c, 61 jurassic.h, 188 ctl_t, 5 ctm_co2, 7 ctm_h2o, 7 ctm_n2, 7 ctm_o2, 7 emitter, 6 fov, 8 hydz, 7	libgps.c, 229 libgps.h, 256 ds los_t, 13 emitter ctl_t, 6 eps tbl_t, 19 events.c, 20 main, 20 find_emitter jurassic.c, 61 jurassic.h, 189
jurassic.h, 154 copy_atm jurassic.c, 60 jurassic.h, 188 copy_obs jurassic.c, 61 jurassic.h, 188 ctl_t, 5 ctm_co2, 7 ctm_h2o, 7 ctm_n2, 7 ctm_o2, 7 emitter, 6 fov, 8 hydz, 7 nd, 6	libgps.c, 229 libgps.h, 256 ds los_t, 13 emitter ctl_t, 6 eps tbl_t, 19 events.c, 20 main, 20 find_emitter jurassic.c, 61 jurassic.h, 189 formod
jurassic.h, 154 copy_atm jurassic.c, 60 jurassic.h, 188 copy_obs jurassic.c, 61 jurassic.h, 188 ctl_t, 5 ctm_co2, 7 ctm_h2o, 7 ctm_n2, 7 ctm_o2, 7 emitter, 6 fov, 8 hydz, 7 nd, 6 ng, 6	libgps.c, 229 libgps.h, 256 ds los_t, 13 emitter ctl_t, 6 eps tbl_t, 19 events.c, 20 main, 20 find_emitter jurassic.c, 61 jurassic.h, 189 formod jurassic.c, 62
jurassic.h, 154 copy_atm jurassic.c, 60 jurassic.h, 188 copy_obs jurassic.c, 61 jurassic.h, 188 ctl_t, 5 ctm_co2, 7 ctm_h2o, 7 ctm_n2, 7 ctm_o2, 7 emitter, 6 fov, 8 hydz, 7 nd, 6 ng, 6 nu, 7	libgps.c, 229 libgps.h, 256 ds los_t, 13 emitter ctl_t, 6 eps tbl_t, 19 events.c, 20 main, 20 find_emitter jurassic.c, 61 jurassic.h, 189 formod jurassic.c, 62 jurassic.h, 189
jurassic.h, 154 copy_atm jurassic.c, 60 jurassic.h, 188 copy_obs jurassic.c, 61 jurassic.h, 188 ctl_t, 5 ctm_co2, 7 ctm_h2o, 7 ctm_n2, 7 ctm_o2, 7 emitter, 6 fov, 8 hydz, 7 nd, 6 ng, 6 nu, 7 nw, 6	libgps.c, 229 libgps.h, 256 ds los_t, 13 emitter ctl_t, 6 eps tbl_t, 19 events.c, 20 main, 20 find_emitter jurassic.c, 61 jurassic.h, 189 formod jurassic.h, 189 formod_continua
jurassic.h, 154 copy_atm jurassic.c, 60 jurassic.h, 188 copy_obs jurassic.c, 61 jurassic.h, 188 ctl_t, 5 ctm_co2, 7 ctm_h2o, 7 ctm_n2, 7 ctm_o2, 7 emitter, 6 fov, 8 hydz, 7 nd, 6 ng, 6 nu, 7 nw, 6 rayds, 8	libgps.c, 229 libgps.h, 256 ds los_t, 13 emitter ctl_t, 6 eps tbl_t, 19 events.c, 20 main, 20 find_emitter jurassic.c, 61 jurassic.h, 189 formod jurassic.h, 189 formod_continua jurassic.c, 63
jurassic.h, 154 copy_atm jurassic.c, 60 jurassic.h, 188 copy_obs jurassic.h, 188 ctl_t, 5 ctm_co2, 7 ctm_h2o, 7 ctm_n2, 7 ctm_o2, 7 emitter, 6 fov, 8 hydz, 7 nd, 6 ng, 6 nu, 7 nw, 6 rayds, 8 raydz, 8	libgps.c, 229 libgps.h, 256 ds los_t, 13 emitter ctl_t, 6 eps tbl_t, 19 events.c, 20 main, 20 find_emitter jurassic.c, 61 jurassic.h, 189 formod jurassic.c, 62 jurassic.h, 189 formod_continua jurassic.c, 63 jurassic.h, 191
jurassic.h, 154 copy_atm jurassic.c, 60 jurassic.h, 188 copy_obs jurassic.c, 61 jurassic.h, 188 ctl_t, 5 ctm_co2, 7 ctm_h2o, 7 ctm_n2, 7 ctm_o2, 7 emitter, 6 fov, 8 hydz, 7 nd, 6 ng, 6 nu, 7 nw, 6 rayds, 8 raydz, 8 refrac, 7	libgps.c, 229 libgps.h, 256 ds los_t, 13 emitter ctl_t, 6 eps tbl_t, 19 events.c, 20 main, 20 find_emitter jurassic.c, 61 jurassic.h, 189 formod jurassic.h, 189 formod_continua jurassic.h, 189 formod_continua jurassic.h, 191 formod_fov
jurassic.h, 154 copy_atm jurassic.c, 60 jurassic.h, 188 copy_obs jurassic.c, 61 jurassic.h, 188 ctl_t, 5 ctm_co2, 7 ctm_h2o, 7 ctm_n2, 7 ctm_o2, 7 emitter, 6 fov, 8 hydz, 7 nd, 6 ng, 6 nu, 7 nw, 6 rayds, 8 raydz, 8 refrac, 7 retk_zmax, 9	libgps.c, 229 libgps.h, 256 ds los_t, 13 emitter ctl_t, 6 eps tbl_t, 19 events.c, 20 main, 20 find_emitter jurassic.c, 61 jurassic.h, 189 formod jurassic.c, 62 jurassic.h, 189 formod_continua jurassic.c, 63 jurassic.h, 191
jurassic.h, 154 copy_atm jurassic.c, 60 jurassic.h, 188 copy_obs jurassic.c, 61 jurassic.h, 188 ctl_t, 5 ctm_co2, 7 ctm_h2o, 7 ctm_n2, 7 ctm_o2, 7 emitter, 6 fov, 8 hydz, 7 nd, 6 ng, 6 nu, 7 nw, 6 rayds, 8 raydz, 8 refrac, 7	libgps.c, 229 libgps.h, 256 ds los_t, 13 emitter ctl_t, 6 eps tbl_t, 19 events.c, 20 main, 20 find_emitter jurassic.c, 61 jurassic.h, 189 formod jurassic.h, 189 formod_continua jurassic.h, 189 formod_continua jurassic.h, 191 formod_fov
jurassic.h, 154 copy_atm jurassic.c, 60 jurassic.h, 188 copy_obs jurassic.c, 61 jurassic.h, 188 ctl_t, 5 ctm_co2, 7 ctm_h2o, 7 ctm_n2, 7 ctm_o2, 7 emitter, 6 fov, 8 hydz, 7 nd, 6 ng, 6 nu, 7 nw, 6 rayds, 8 raydz, 8 refrac, 7 retk_zmax, 9	libgps.c, 229 libgps.h, 256 ds los_t, 13 emitter ctl_t, 6 eps tbl_t, 19 events.c, 20 main, 20 find_emitter jurassic.c, 61 jurassic.h, 189 formod jurassic.h, 189 formod_continua jurassic.c, 63 jurassic.h, 191 formod_fov jurassic.c, 64
jurassic.h, 154 copy_atm jurassic.c, 60 jurassic.h, 188 copy_obs jurassic.c, 61 jurassic.h, 188 ctl_t, 5 ctm_co2, 7 ctm_h2o, 7 ctm_n2, 7 ctm_o2, 7 emitter, 6 fov, 8 hydz, 7 nd, 6 ng, 6 nu, 7 nw, 6 rayds, 8 raydz, 8 refrac, 7 retk_zmax, 9 retk_zmin, 9	libgps.c, 229 libgps.h, 256 ds los_t, 13 emitter ctl_t, 6 eps tbl_t, 19 events.c, 20 main, 20 find_emitter jurassic.c, 61 jurassic.h, 189 formod jurassic.c, 62 jurassic.h, 189 formod_continua jurassic.c, 63 jurassic.h, 191 formod_fov jurassic.h, 191
jurassic.h, 154 copy_atm jurassic.c, 60 jurassic.h, 188 copy_obs jurassic.c, 61 jurassic.h, 188 ctl_t, 5 ctm_co2, 7 ctm_h2o, 7 ctm_n2, 7 ctm_o2, 7 emitter, 6 fov, 8 hydz, 7 nd, 6 ng, 6 nu, 7 nw, 6 rayds, 8 raydz, 8 refrac, 7 retk_zmax, 9 retk_zmin, 9 retp_zmax, 8	libgps.c, 229 libgps.h, 256 ds los_t, 13 emitter ctl_t, 6 eps tbl_t, 19 events.c, 20 main, 20 find_emitter jurassic.c, 61 jurassic.h, 189 formod jurassic.c, 62 jurassic.h, 189 formod_continua jurassic.c, 63 jurassic.h, 191 formod_fov jurassic.h, 191 formod_pencil

formod_srcfunc	libgps.c, 233
jurassic.c, 67	libgps.h, 260
jurassic.h, 194	intpol_met_time
fov	libgps.c, 234
ctl t, 8	libgps.h, 261
Cii_i, O	intpol_tbl
gauss	•
libgps.c, 230	jurassic.c, 71
libgps.h, 257	jurassic.h, 199
.	intpol_tbl_eps
geo2cart	jurassic.c, 73
jurassic.c, 67	jurassic.h, 201
jurassic.h, 195	intpol_tbl_u
get_met	jurassic.c, 74
libgps.c, 231	jurassic.h, <mark>201</mark>
libgps.h, 259	
get_met_help	jsec2time
libgps.c, 232	jurassic.c, 74
libgps.h, 259	jurassic.h, <mark>202</mark>
gps_t, 9	jurassic.c, 23
lat, 11	atm2x, 26
lon, 11	atm2x_help, 26
nds, 10	brightness, 26
nz, 10	cart2geo, 27
	climatology, 27
p, 11	copy_atm, 60
pt, 11	copy_obs, 61
t, 11	
th, 11	ctmco2, 36
time, 10	ctmh2o, 46
wv, 11	ctmn2, 58
z, 10	ctmo2, 59
grid_gps	find_emitter, 61
libgps.c, 230	formod, 62
libgps.h, 258	formod_continua, 63
	formod_fov, 64
hamming_high_pass	formod_pencil, 65
libgps.c, 235	formod_srcfunc, 67
libgps.h, 262	geo2cart, 67
hamming_low_pass	hydrostatic, 68
libgps.c, 235	idx2name, 69
libgps.h, 262	init_tbl, 69
hydrostatic	intpol atm, 71
jurassic.c, 68	intpol_tbl, 71
-	intpol_tbl_eps, 73
jurassic.h, 195	intpol_tbl_u, 74
hydz	jsec2time, 74
ctl_t, 7	-
iduOnama	kernel, 75
idx2name	locate_irr, 77
jurassic.c, 69	locate_reg, 77
jurassic.h, 196	locate_tbl, 78
init_tbl	obs2y, <mark>78</mark>
jurassic.c, 69	planck, 78
jurassic.h, 196	raytrace, 79
intpol_atm	read_atm, 81
jurassic.c, 71	read_ctl, 82
jurassic.h, 198	read_matrix, 83
intpol_met_3d	read_obs, 84
libgps.c, 233	read_shape, 85
libgps.h, 260	refractivity, 85
intpol_met_space	scan_ctl, 85
pomot_opaso	55a.i_6ti, 65

tangent_point, 86	y2obs, 220
time2jsec, 87	
timer, 87	k
write_atm, 88	atm_t, 4
write_matrix, 89	los_t, 13
write_obs, 91	kernel
x2atm, 92	jurassic.c, 75
x2atm_help, 93	jurassic.h, 202
y2obs, 93	lat
jurassic.h, 150	atm t, 4
atm2x, 153	gps t, 11
atm2x_help, 153	los_t, 13
brightness, 154	met t, 15
cart2geo, 154	libgps.c, 227
climatology, 154	add_var, 228
copy_atm, 188	detrend met, 229
copy_obs, 188	gauss, 230
ctmco2, 164	get_met, 231
ctmh2o, 174	get_met_help, 232
ctmn2, 186	grid_gps, 230
ctmo2, 187	hamming_high_pass, 235
find_emitter, 189	hamming_low_pass, 235
formod, 189	intpol_met_3d, 233
formod_continua, 191	intpol_met_space, 233
formod_fov, 191	intpol_met_time, 234
formod_pencil, 193	poly, 236
formod_srcfunc, 194	poly_help, 237
geo2cart, 195	read_gps, 239
hydrostatic, 195	read_gps_prof, 237
idx2name, 196	read_met, 239
init_tbl, 196	read_met_extrapolate, 241
intpol_atm, 198	read_met_help, 241
intpol_tbl, 199	read_met_periodic, 242
intpol_tbl_eps, 201	tropopause, 242
intpol_tbl_u, 201	write_gps, 242
jsec2time, 202	libgps.h, 255
kernel, 202 locate irr, 204	add_var, 256
locate_reg, 204	detrend_met, 256
locate_tbl, 205	gauss, 257
obs2y, 205	get_met, 259
planck, 205	get_met_help, 259
raytrace, 206	grid_gps, 258
read atm, 208	hamming_high_pass, 262
read_ctl, 209	hamming_low_pass, 262 intpol met 3d, 260
read matrix, 210	intpol_met_space, 260
read_obs, 211	intpol_met_space, 200
read_shape, 212	poly, 263
refractivity, 212	poly_help, 264
scan_ctl, 212	read_gps, 266
tangent_point, 213	read_gps_prof, 264
time2jsec, 214	read_gps_prof, 204
timer, 214	read_met_extrapolate, 268
write_atm, 215	read_met_help, 268
write_matrix, 216	read_met_periodic, 269
write_obs, 218	tropopause, 269
x2atm, 219	write_gps, 269
x2atm_help, 220	locate_irr
	

	jurassic.c, 77		tbl t, 19
	jurassic.h, 204	nu	
loca	te_reg		ctl_t, 7
.000	jurassic.c, 77		tbl_t, 19
	jurassic.h, 204	מאו	ω_ι, 13
looo	te_tbl	nw	atl t C
ioca			ctl_t, 6
	jurassic.c, 78	nx	
	jurassic.h, 205		met_t, 14
lon		ny	
	atm_t, 4		met_t, 15
	gps_t, 11	nz	
	los_t, 12		gps_t, 10
	met_t, 15		
los_	t, 12	obs	2y
	ds, 13		jurassic.c, 78
	k, 13		jurassic.h, 205
	lat, 13	obs	_t, 15
	lon, 12	0.00_	nr, 16
	np, 12		obslat, 17
	p, 13		obslat, 17
	q, 13		obsidit, 17
	t, 13		
	tsurf, 13		rad, 18
	u, 13		tau, 17
			time, 16
	z, 12		tplat, 17
maiı	2		tplon, 17
IIIaii			tpz, 17
	events.c, 20		vplat, 17
	map.c, 273		vplon, 17
	perturbation.c, 275		
	•		vpz, 17
	prof.c, 278	obsl	vpz, 17 lat
	prof.c, 278 response.c, 280	obsl	lat
	prof.c, 278		at obs_t, 17
map	prof.c, 278 response.c, 280	obsl	lat obs_t, 17 lon
map	prof.c, 278 response.c, 280 variance.c, 284	obsl	obs_t, 17 lon obs_t, 17
	prof.c, 278 response.c, 280 variance.c, 284 o.c, 273		at obs_t, 17 lon obs_t, 17 z
	prof.c, 278 response.c, 280 variance.c, 284 o.c, 273 main, 273	obsl	obs_t, 17 lon obs_t, 17
	prof.c, 278 response.c, 280 variance.c, 284 o.c, 273 main, 273 _t, 14	obsi	at obs_t, 17 lon obs_t, 17 z
	prof.c, 278 response.c, 280 variance.c, 284 o.c, 273 main, 273 _t, 14 lat, 15 lon, 15	obsl	dat obs_t, 17 don obs_t, 17 z obs_t, 16
	prof.c, 278 response.c, 280 variance.c, 284 o.c, 273 main, 273 _t, 14 lat, 15	obsi	dat obs_t, 17 don obs_t, 17 z obs_t, 16 atm_t, 4
	prof.c, 278 response.c, 280 variance.c, 284 o.c, 273 main, 273 _t, 14 lat, 15 lon, 15 np, 15 nx, 14	obsi	at obs_t, 17 lon obs_t, 17 z obs_t, 16 atm_t, 4 gps_t, 11
	prof.c, 278 response.c, 280 variance.c, 284 o.c, 273 main, 273 _t, 14 lat, 15 lon, 15 np, 15 nx, 14 ny, 15	obsi	at obs_t, 17 lon obs_t, 17 z obs_t, 16 atm_t, 4 gps_t, 11 los_t, 13
	prof.c, 278 response.c, 280 variance.c, 284 o.c, 273 main, 273 _t, 14 lat, 15 lon, 15 np, 15 nx, 14 ny, 15 p, 15	obsi	at obs_t, 17 lon obs_t, 17 z obs_t, 16 atm_t, 4 gps_t, 11 los_t, 13 met_t, 15
	prof.c, 278 response.c, 280 variance.c, 284 o.c, 273 main, 273 t, 14 lat, 15 lon, 15 np, 15 nx, 14 ny, 15 p, 15 t, 15	obsi	at obs_t, 17 lon obs_t, 17 z obs_t, 16 atm_t, 4 gps_t, 11 los_t, 13
	prof.c, 278 response.c, 280 variance.c, 284 o.c, 273 main, 273 _t, 14 lat, 15 lon, 15 np, 15 nx, 14 ny, 15 p, 15	obsi obsi p	at obs_t, 17 lon obs_t, 17 z obs_t, 16 atm_t, 4 gps_t, 11 los_t, 13 met_t, 15
met	prof.c, 278 response.c, 280 variance.c, 284 o.c, 273 main, 273 t, 14 lat, 15 lon, 15 np, 15 nx, 14 ny, 15 p, 15 t, 15	obsi obsi p	at obs_t, 17 lon obs_t, 17 z obs_t, 16 atm_t, 4 gps_t, 11 los_t, 13 met_t, 15 tbl_t, 19
	prof.c, 278 response.c, 280 variance.c, 284 o.c, 273 main, 273 _t, 14 lat, 15 lon, 15 np, 15 nx, 14 ny, 15 p, 15 t, 15 time, 14	obsi obsi p	at obs_t, 17 lon obs_t, 17 z obs_t, 16 atm_t, 4 gps_t, 11 los_t, 13 met_t, 15 tbl_t, 19 urbation.c, 275 main, 275
met <u>i</u>	prof.c, 278 response.c, 280 variance.c, 284 o.c, 273 main, 273 t, 14 lat, 15 lon, 15 np, 15 nx, 14 ny, 15 p, 15 t, 15	obsi obsi p	at obs_t, 17 lon obs_t, 17 z obs_t, 16 atm_t, 4 gps_t, 11 los_t, 13 met_t, 15 tbl_t, 19 urbation.c, 275 main, 275 ock
met	prof.c, 278 response.c, 280 variance.c, 284 o.c, 273 main, 273 _t, 14 lat, 15 lon, 15 np, 15 nx, 14 ny, 15 p, 15 t, 15 time, 14	obsi obsi p	at obs_t, 17 lon obs_t, 17 z obs_t, 16 atm_t, 4 gps_t, 11 los_t, 13 met_t, 15 tbl_t, 19 urbation.c, 275 main, 275 ock jurassic.c, 78
met nd	prof.c, 278 response.c, 280 variance.c, 284 o.c, 273 main, 273 _t, 14 lat, 15 lon, 15 np, 15 nx, 14 ny, 15 p, 15 t, 15 time, 14	obsize p pert	at obs_t, 17 don obs_t, 17 z obs_t, 16 atm_t, 4 gps_t, 11 los_t, 13 met_t, 15 tbl_t, 19 urbation.c, 275 main, 275 ock jurassic.c, 78 jurassic.h, 205
met <u>i</u>	prof.c, 278 response.c, 280 variance.c, 284 o.c, 273 main, 273 _t, 14 lat, 15 lon, 15 np, 15 nx, 14 ny, 15 p, 15 t, 15 time, 14 ctl_t, 6 gps_t, 10	obsi obsi p	at obs_t, 17 don obs_t, 17 z obs_t, 16 atm_t, 4 gps_t, 11 los_t, 13 met_t, 15 tbl_t, 19 urbation.c, 275 main, 275 ock jurassic.c, 78 jurassic.h, 205
nd nds	prof.c, 278 response.c, 280 variance.c, 284 o.c, 273 main, 273 _t, 14 lat, 15 lon, 15 np, 15 nx, 14 ny, 15 p, 15 t, 15 time, 14	obsize p pert	obs_t, 17 lon obs_t, 17 z obs_t, 17 z obs_t, 16 atm_t, 4 gps_t, 11 los_t, 13 met_t, 15 tbl_t, 19 urbation.c, 275 main, 275 ock jurassic.h, 205 // libgps.c, 236
met nd	prof.c, 278 response.c, 280 variance.c, 284 b.c, 273 main, 273 _t, 14 lat, 15 lon, 15 np, 15 nx, 14 ny, 15 p, 15 t, 15 time, 14 ctl_t, 6 gps_t, 10 ctl_t, 6	obsize p perturbation poly	at obs_t, 17 lon obs_t, 17 z obs_t, 16 atm_t, 4 gps_t, 11 los_t, 13 met_t, 15 tbl_t, 19 urbation.c, 275 main, 275 ock jurassic.c, 78 jurassic.h, 205 libgps.c, 236 libgps.h, 263
nd nds	prof.c, 278 response.c, 280 variance.c, 284 b.c, 273 main, 273 _t, 14 lat, 15 lon, 15 np, 15 nx, 14 ny, 15 p, 15 t, 15 time, 14 ctl_t, 6 atm_t, 4	obsize p perturbation poly	obs_t, 17 lon obs_t, 17 z obs_t, 16 atm_t, 4 gps_t, 11 los_t, 13 met_t, 15 tbl_t, 19 urbation.c, 275 main, 275 lock jurassic.c, 78 jurassic.h, 205 libgps.c, 236 libgps.h, 263 y_help
nd nds	prof.c, 278 response.c, 280 variance.c, 284 o.c, 273 main, 273 _t, 14 lat, 15 lon, 15 np, 15 nx, 14 ny, 15 p, 15 t, 15 time, 14 ctl_t, 6 atm_t, 4 los_t, 12	obsize p perturbation poly	obs_t, 17 lon obs_t, 17 z obs_t, 16 atm_t, 4 gps_t, 11 los_t, 13 met_t, 15 tbl_t, 19 urbation.c, 275 main, 275 ock jurassic.c, 78 jurassic.h, 205 libgps.c, 236 libgps.h, 263 y_help libgps.c, 237
nd nds	prof.c, 278 response.c, 280 variance.c, 284 o.c, 273 main, 273 _t, 14 lat, 15 lon, 15 np, 15 nx, 14 ny, 15 p, 15 t, 15 time, 14 ctl_t, 6 atm_t, 4 los_t, 12 met_t, 15	obsize p pertiplant poly	at obs_t, 17 lon obs_t, 17 lon obs_t, 17 z obs_t, 16 atm_t, 4 gps_t, 11 los_t, 13 met_t, 15 tbl_t, 19 urbation.c, 275 main, 275 lock jurassic.c, 78 jurassic.h, 205 libgps.c, 236 libgps.h, 263 help libgps.c, 237 libgps.h, 264
nd nds ng	prof.c, 278 response.c, 280 variance.c, 284 o.c, 273 main, 273 _t, 14 lat, 15 lon, 15 np, 15 nx, 14 ny, 15 p, 15 t, 15 time, 14 ctl_t, 6 atm_t, 4 los_t, 12	obsize p pertiplant poly	at obs_t, 17 lon obs_t, 17 lon obs_t, 17 z obs_t, 16 atm_t, 4 gps_t, 11 los_t, 13 met_t, 15 tbl_t, 19 urbation.c, 275 main, 275 lock jurassic.c, 78 jurassic.h, 205 libgps.c, 236 libgps.h, 263 y_help libgps.c, 237 libgps.h, 264 .c, 278
nd nds	prof.c, 278 response.c, 280 variance.c, 284 o.c, 273 main, 273 _t, 14 lat, 15 lon, 15 np, 15 nx, 14 ny, 15 p, 15 t, 15 time, 14 ctl_t, 6 atm_t, 4 los_t, 12 met_t, 15 tbl_t, 19	obside ob	at obs_t, 17 lon obs_t, 17 lon obs_t, 17 z obs_t, 16 atm_t, 4 gps_t, 11 los_t, 13 met_t, 15 tbl_t, 19 urbation.c, 275 main, 275 lock jurassic.c, 78 jurassic.h, 205 libgps.c, 236 libgps.h, 263 help libgps.c, 237 libgps.h, 264
nd nds ng	prof.c, 278 response.c, 280 variance.c, 284 o.c, 273 main, 273 _t, 14 lat, 15 lon, 15 np, 15 nx, 14 ny, 15 p, 15 t, 15 time, 14 ctl_t, 6 atm_t, 4 los_t, 12 met_t, 15	obsize p pertiplant poly	at obs_t, 17 lon obs_t, 17 lon obs_t, 17 z obs_t, 16 atm_t, 4 gps_t, 11 los_t, 13 met_t, 15 tbl_t, 19 urbation.c, 275 main, 275 lock jurassic.c, 78 jurassic.h, 205 libgps.c, 236 libgps.h, 263 y_help libgps.c, 237 libgps.h, 264 .c, 278
nd nds ng	prof.c, 278 response.c, 280 variance.c, 284 o.c, 273 main, 273 _t, 14 lat, 15 lon, 15 np, 15 nx, 14 ny, 15 p, 15 t, 15 time, 14 ctl_t, 6 atm_t, 4 los_t, 12 met_t, 15 tbl_t, 19	obside ob	at obs_t, 17 lon obs_t, 17 lon obs_t, 17 z obs_t, 16 atm_t, 4 gps_t, 11 los_t, 13 met_t, 15 tbl_t, 19 urbation.c, 275 main, 275 lock jurassic.c, 78 jurassic.h, 205 libgps.c, 236 libgps.h, 263 y_help libgps.c, 237 libgps.h, 264 .c, 278

q	retp_zmin
atm_t, 4	ctl_t, 8
los_t, 13	retq_zmax
rad	ctl_t, 8
	retq_zmin
obs_t, 18	ctl_t, 8
rayds	rett_zmax
ctl_t, 8	ctl_t, 8
raydz	rett_zmin
ctl_t, 8	ctl_t, 8
raytrace	
jurassic.c, 79	scan_ctl
jurassic.h, 206	jurassic.c, 85
read_atm	jurassic.h, 212
jurassic.c, 81	sr
jurassic.h, 208	tbl_t, 19
read_ctl	st
jurassic.c, 82	tbl_t, 19
jurassic.h, 209	_ ′
read_gps	t
libgps.c, 239	atm_t, 4
libgps.h, 266	gps_t, 11
read_gps_prof	los_t, 13
libgps.c, 237	met_t, 15
libgps.h, 264	tbl_t, 19
read_matrix	tangent_point
jurassic.c, 83	jurassic.c, 86
	-
jurassic.h, 210	jurassic.h, 213
read_met	tau
libgps.c, 239	obs_t, 17
libgps.h, 266	tbl_t, 18
libgps.h, 266 read_met_extrapolate	tbl_t, 18 eps, 19
libgps.h, 266 read_met_extrapolate libgps.c, 241	tbl_t, 18 eps, 19 np, 19
libgps.h, 266 read_met_extrapolate libgps.c, 241 libgps.h, 268	tbl_t, 18 eps, 19 np, 19 nt, 19
libgps.h, 266 read_met_extrapolate libgps.c, 241 libgps.h, 268 read_met_help	tbl_t, 18 eps, 19 np, 19
libgps.h, 266 read_met_extrapolate libgps.c, 241 libgps.h, 268 read_met_help libgps.c, 241	tbl_t, 18 eps, 19 np, 19 nt, 19
libgps.h, 266 read_met_extrapolate libgps.c, 241 libgps.h, 268 read_met_help	tbl_t, 18 eps, 19 np, 19 nt, 19 nu, 19
libgps.h, 266 read_met_extrapolate libgps.c, 241 libgps.h, 268 read_met_help libgps.c, 241	tbl_t, 18 eps, 19 np, 19 nt, 19 nu, 19 p, 19
libgps.h, 266 read_met_extrapolate libgps.c, 241 libgps.h, 268 read_met_help libgps.c, 241 libgps.h, 268	tbl_t, 18 eps, 19 np, 19 nt, 19 nu, 19 p, 19 sr, 19
libgps.h, 266 read_met_extrapolate libgps.c, 241 libgps.h, 268 read_met_help libgps.c, 241 libgps.h, 268 read_met_periodic	tbl_t, 18 eps, 19 np, 19 nt, 19 nu, 19 p, 19 sr, 19 st, 19
libgps.h, 266 read_met_extrapolate libgps.c, 241 libgps.h, 268 read_met_help libgps.c, 241 libgps.h, 268 read_met_periodic libgps.c, 242	tbl_t, 18 eps, 19 np, 19 nt, 19 nu, 19 p, 19 sr, 19 st, 19 t, 19
libgps.h, 266 read_met_extrapolate libgps.c, 241 libgps.h, 268 read_met_help libgps.c, 241 libgps.h, 268 read_met_periodic libgps.c, 242 libgps.h, 269	tbl_t, 18 eps, 19 np, 19 nt, 19 nu, 19 p, 19 sr, 19 st, 19 t, 19 u, 19
libgps.h, 266 read_met_extrapolate libgps.c, 241 libgps.h, 268 read_met_help libgps.c, 241 libgps.h, 268 read_met_periodic libgps.c, 242 libgps.h, 269 read_obs jurassic.c, 84	tbl_t, 18
libgps.h, 266 read_met_extrapolate libgps.c, 241 libgps.h, 268 read_met_help libgps.c, 241 libgps.h, 268 read_met_periodic libgps.c, 242 libgps.h, 269 read_obs jurassic.c, 84 jurassic.h, 211	tbl_t, 18
libgps.h, 266 read_met_extrapolate libgps.c, 241 libgps.h, 268 read_met_help libgps.c, 241 libgps.h, 268 read_met_periodic libgps.c, 242 libgps.h, 269 read_obs jurassic.c, 84 jurassic.h, 211 read_shape	tbl_t, 18
libgps.h, 266 read_met_extrapolate libgps.c, 241 libgps.h, 268 read_met_help libgps.c, 241 libgps.h, 268 read_met_periodic libgps.c, 242 libgps.h, 269 read_obs jurassic.c, 84 jurassic.h, 211 read_shape jurassic.c, 85	tbl_t, 18
libgps.h, 266 read_met_extrapolate libgps.c, 241 libgps.h, 268 read_met_help libgps.c, 241 libgps.h, 268 read_met_periodic libgps.c, 242 libgps.h, 269 read_obs jurassic.c, 84 jurassic.h, 211 read_shape jurassic.c, 85 jurassic.h, 212	tbl_t, 18
libgps.h, 266 read_met_extrapolate libgps.c, 241 libgps.h, 268 read_met_help libgps.c, 241 libgps.h, 268 read_met_periodic libgps.c, 242 libgps.h, 269 read_obs jurassic.c, 84 jurassic.h, 211 read_shape jurassic.c, 85 jurassic.h, 212 refrac	tbl_t, 18
libgps.h, 266 read_met_extrapolate libgps.c, 241 libgps.h, 268 read_met_help libgps.c, 241 libgps.h, 268 read_met_periodic libgps.c, 242 libgps.h, 269 read_obs jurassic.c, 84 jurassic.h, 211 read_shape jurassic.c, 85 jurassic.h, 212 refrac ctl_t, 7	tbl_t, 18
libgps.h, 266 read_met_extrapolate libgps.c, 241 libgps.h, 268 read_met_help libgps.c, 241 libgps.h, 268 read_met_periodic libgps.c, 242 libgps.h, 269 read_obs jurassic.c, 84 jurassic.h, 211 read_shape jurassic.c, 85 jurassic.h, 212 refrac ctl_t, 7 refractivity	tbl_t, 18
libgps.h, 266 read_met_extrapolate libgps.c, 241 libgps.h, 268 read_met_help libgps.c, 241 libgps.h, 268 read_met_periodic libgps.c, 242 libgps.h, 269 read_obs jurassic.c, 84 jurassic.h, 211 read_shape jurassic.c, 85 jurassic.h, 212 refrac ctl_t, 7 refractivity jurassic.c, 85	tbl_t, 18
libgps.h, 266 read_met_extrapolate libgps.c, 241 libgps.h, 268 read_met_help libgps.c, 241 libgps.h, 268 read_met_periodic libgps.c, 242 libgps.h, 269 read_obs jurassic.c, 84 jurassic.h, 211 read_shape jurassic.c, 85 jurassic.h, 212 refrac ctl_t, 7 refractivity jurassic.c, 85 jurassic.h, 212	tbl_t, 18
libgps.h, 266 read_met_extrapolate libgps.c, 241 libgps.h, 268 read_met_help libgps.c, 241 libgps.h, 268 read_met_periodic libgps.c, 242 libgps.h, 269 read_obs jurassic.c, 84 jurassic.h, 211 read_shape jurassic.c, 85 jurassic.h, 212 refrac ctl_t, 7 refractivity jurassic.h, 212 response.c, 280	tbl_t, 18
libgps.h, 266 read_met_extrapolate libgps.c, 241 libgps.h, 268 read_met_help libgps.c, 241 libgps.h, 268 read_met_periodic libgps.c, 242 libgps.h, 269 read_obs jurassic.c, 84 jurassic.h, 211 read_shape jurassic.c, 85 jurassic.h, 212 refrac ctl_t, 7 refractivity jurassic.h, 212 response.c, 280 main, 280	tbl_t, 18
libgps.h, 266 read_met_extrapolate libgps.c, 241 libgps.h, 268 read_met_help libgps.c, 241 libgps.h, 268 read_met_periodic libgps.c, 242 libgps.h, 269 read_obs jurassic.c, 84 jurassic.h, 211 read_shape jurassic.c, 85 jurassic.h, 212 refrac ctl_t, 7 refractivity jurassic.h, 212 response.c, 280 main, 280 retk_zmax	tbl_t, 18
libgps.h, 266 read_met_extrapolate libgps.c, 241 libgps.h, 268 read_met_help libgps.c, 241 libgps.h, 268 read_met_periodic libgps.c, 242 libgps.h, 269 read_obs jurassic.c, 84 jurassic.h, 211 read_shape jurassic.c, 85 jurassic.h, 212 refrac ctl_t, 7 refractivity jurassic.h, 212 response.c, 280 main, 280 retk_zmax ctl_t, 9	tbl_t, 18
libgps.h, 266 read_met_extrapolate libgps.c, 241 libgps.h, 268 read_met_help libgps.c, 241 libgps.h, 268 read_met_periodic libgps.c, 242 libgps.h, 269 read_obs jurassic.c, 84 jurassic.h, 211 read_shape jurassic.c, 85 jurassic.h, 212 refrac ctl_t, 7 refractivity jurassic.h, 212 response.c, 280 main, 280 retk_zmax ctl_t, 9 retk_zmin	tbl_t, 18
libgps.h, 266 read_met_extrapolate libgps.c, 241 libgps.h, 268 read_met_help libgps.c, 241 libgps.h, 268 read_met_periodic libgps.c, 242 libgps.h, 269 read_obs jurassic.c, 84 jurassic.h, 211 read_shape jurassic.c, 85 jurassic.h, 212 refrac ctl_t, 7 refractivity jurassic.h, 212 response.c, 280 main, 280 retk_zmax ctl_t, 9 retk_zmin ctl_t, 9	tbl_t, 18
libgps.h, 266 read_met_extrapolate libgps.c, 241 libgps.h, 268 read_met_help libgps.c, 241 libgps.h, 268 read_met_periodic libgps.c, 242 libgps.h, 269 read_obs jurassic.c, 84 jurassic.h, 211 read_shape jurassic.c, 85 jurassic.h, 212 refrac ctl_t, 7 refractivity jurassic.h, 212 response.c, 280 main, 280 retk_zmax ctl_t, 9 retk_zmin ctl_t, 9 retp_zmax	tbl_t, 18
libgps.h, 266 read_met_extrapolate libgps.c, 241 libgps.h, 268 read_met_help libgps.c, 241 libgps.h, 268 read_met_periodic libgps.c, 242 libgps.h, 269 read_obs jurassic.c, 84 jurassic.h, 211 read_shape jurassic.c, 85 jurassic.h, 212 refrac ctl_t, 7 refractivity jurassic.h, 212 response.c, 280 main, 280 retk_zmax ctl_t, 9 retk_zmin ctl_t, 9	tbl_t, 18

```
tpz
    obs_t, 17
tropopause
     libgps.c, 242
     libgps.h, 269
tsurf
     los_t, 13
u
     los_t, 13
     tbl_t, 19
variance.c, 284
     main, 284
vplat
     obs_t, 17
vplon
     obs_t, 17
vpz
    obs_t, 17
window
     ctl_t, 7
write_atm
    jurassic.c, 88
    jurassic.h, 215
write_bbt
    ctl_t, 9
write_gps
     libgps.c, 242
     libgps.h, 269
write_matrix
     ctl_t, 9
    jurassic.c, 89
    jurassic.h, 216
write_obs
    jurassic.c, 91
    jurassic.h, 218
W۷
     gps_t, 11
x2atm
     jurassic.c, 92
    jurassic.h, 219
x2atm_help
    jurassic.c, 93
    jurassic.h, 220
y2obs
    jurassic.c, 93
    jurassic.h, 220
     atm_t, 4
     gps_t, 10
     los_t, 12
```