reactionsetup

January 16, 2006
15:54

Contents

1 Routines used to set up the reactions file **1**

2 References **6**

3 INDEX **7**

1 Routines used to set up the reactions file

```
$Id: reactionsetup.web,v 1.27 2000/04/01 17:34:02 dstotler Exp $
```

The file `reaction_infile` identified in the `degas2.in` file describes all of the reactions available in DEGAS 2. Each reaction is represented by four separate lines:

1. The full name of the reaction
2. A blank delimited sequence of entries like

   ```
   reagent1 + reagent2 -> product1 + product2 + ...
   ```

   where reagent1, product1, etc. are species symbols. Currently only binary reactions are supported. Most reactions involve one background reagent and one test reagent. This ordering is now enforced by *problemsetup*. Reactions between two background species (e.g., recombination) are also permitted provided at least one of the products is a test species. Such reactions are treated as sources.
3. The path name (relative or absolute) of the netCDF data file providing the reaction rate and collision handling data for this reaction.
4. The reaction type. This is used to group reactions which are sufficiently similar that their products and collisions can be processed using the same code. Currently recognized types include

   chargex charge exchange

   elastic ion - neutral elastic scattering

   dissoc Molecular dissociation and dissociative ionization

   dissoc_rec Molecular dissociative recombination

   ionize Ionization

   ionize_suppress Ionization handled with the technique of suppressed absorption.

   recombination Recombination; is actually a source of neutral test species.

Here are some example reactions:

```
electron impact ionization of hydrogen
hionize e + H *  -> e + H+ + e
../data/hionize.nc
ionize_suppress

electron recombination of hydrogen
hrecombine e + H+ -> H *
../data/hrecombine.nc
recombination

hydrogen charge exchange
hchex H+ + H -> H + H+
../data/hchex.nc
chargex

molecular hydrogen dissociation
h2dis e + H2 -> e + H + H
../data/h2dis.nc
dissoc
```

Again, this input file is read by *datasetup*, translated into DEGAS 2 data, and written into a netCDF file. The corresponding symbolic name in `degas2.in` is `reactionfile`.

Note that adding a new reaction to this file involves two tasks beyond inserting the appropriate lines in `reaction_infile`. First, the netCDF file for the atomic physics data must be generated. The second task would be to write subroutines for setting up the products and handling collisions. This would be necessary only if a new reaction type were being added.

All of the entries in the above example file are considered generic reactions. That is, each reaction specified is available for all isotopically equivalent reagents and products. The code handles automatically the various isotopic possibilities. Reactions which should apply to only specific species (e.g., ion-neutral elastic scattering) can be identified by replacing the "`->`" symbol with "`=>`".

`"reactionsetup.f"` 1 ≡
  @m FILE `'reactionsetup.web'`

The unnamed module.

`"reactionsetup.f"` 1.1 ≡
  ⟨ Functions and Subroutines 1.2 ⟩

Read in data from `reactions_infile`.

```
⟨ Functions and Subroutines 1.2 ⟩ ≡
    subroutine read_reactions
      implicit_none_f77
      rc_common    // Common
      sp_common
      rf_common
      implicit_none_f90
      integer length, p, b, e, i    // Local
      character*LINELEN line
      character*FILELEN file
      character*FILELEN tempfile
      character*rc_type_len type

      ⟨ Memory allocation interface 0 ⟩
      st_decls
      rc_num = 0
  @#if 0
      rd_table_size = 0
  @#endif
      reaction_version = 'unknown'

      tempfile = filenames_array_reaction_infile
      assert(tempfile ≠ char_undef)
      open(unit = diskin, file = tempfile, form = 'formatted', status = 'old')
      assert(read_string(diskin, line, length))
      assert(length ≤ len(line))
      reaction_version = line(:length)
loop1: continue
      if (¬read_string(diskin, line, length))
        goto eof    /* Add next reaction */
      rc_num = rc_num + 1
      var_realloca(reaction_name)
      var_realloca(reaction_sy)
      var_realloca(reaction_emitter)
      var_realloca(reaction_reagent_num)
      var_realloca(reaction_generic)
      var_realloca(reaction_product_num)
      var_realloca(reaction_reagent)
      var_realloca(reaction_product)
      var_realloca(reaction_type)
      var_realloca(reaction_filename)

        /* First line contains the full name of the reaction */
      length = parse_string(line(:length))
      reaction_name_rc_num = line(:length)

      assert(read_string(diskin, line, length))
      length = parse_string(line(:length))    /* Count and identify reagent species */
      p = 0
      assert(next_token(line, b, e, p))
      rc_sy(rc_num) = read_text(line(b:e))
      rc_reagent_num(rc_num) = 0
      rc_emitter(rc_num) = 0
```

```
loop2: continue
    rc_reagent_num(rc_num) = rc_reagent_num(rc_num) + 1
    assert(rc_reagent_num(rc_num) ≤ rc_reagent_max)
    assert(next_token(line, b, e, p))
    rc_reagent(rc_num, rc_reagent_num(rc_num)) = sp_lookup(line(b : e))
    assert(next_token(line, b, e, p))
    if (line(b : e) ≡ '*') then
      assert(rc_emitter(rc_num) ≡ 0)
      rc_emitter(rc_num) = −rc_reagent_num(rc_num)
      assert(next_token(line, b, e, p))
    end if
    if (line(b : e) ≡ '+') then
      goto loop2
    else if (line(b : e) ≡ '->') then
      rc_gen(rc_num) = rc_generic_yes     // Use generic species
    else if (line(b : e) ≡ '=>') then
      rc_gen(rc_num) = rc_generic_no     // Specific species
    else
      assert('Invalid␣token' ≡ '␣')
    end if     /* Count and identify product species */
    rc_product_num(rc_num) = 0
loop3: continue
    rc_product_num(rc_num) = rc_product_num(rc_num) + 1
    assert(rc_product_num(rc_num) ≤ rc_product_max)
    assert(next_token(line, b, e, p))
    rc_product(rc_num, rc_product_num(rc_num)) = sp_lookup(line(b : e))
    if (next_token(line, b, e, p)) then
      if (line(b : e) ≡ '*') then
        assert(rc_emitter(rc_num) ≡ 0)
        rc_emitter(rc_num) = rc_product_num(rc_num)
        if (next_token(line, b, e, p)) then
          assert(line(b : e) ≡ '+')
          goto loop3
        end if
      else
        assert(line(b : e) ≡ '+')
        goto loop3
      end if
    end if
    if (rc_product_num(rc_num) < rc_product_max) then
      do i = rc_product_num(rc_num) + 1, rc_product_max
        rc_product(rc_num, i) = int_unused
      end do
    end if     /* Third line contains the path name for the data file */
    assert(read_string(diskin, file, length))
    assert(length ≤ len(file))

    rc_filename(rc_num) = file

      /* Fourth line gets the reaction type */
    assert(read_string(diskin, type, length))
    assert(length ≤ len(type))
    rc_reaction_type(rc_num) = type
```

```
      /* Get next reaction */
    goto loop1
eof: continue
    close(unit = diskin)
    var_reallocb(reaction_name)
    var_reallocb(reaction_sy)
    var_reallocb(reaction_emitter)
    var_reallocb(reaction_reagent_num)
    var_reallocb(reaction_generic)
    var_reallocb(reaction_product_num)
    var_reallocb(reaction_reagent)
    var_reallocb(reaction_product)
    var_reallocb(reaction_type)
    var_reallocb(reaction_filename)

    return
  end
```

See also section 1.3.

This code is used in section 1.1.

Write out data into netcdf file `reactions.nc`

⟨ Functions and Subroutines 1.2 ⟩ +≡

```
subroutine nc_write_reaction
  implicit_none_f77
  rc_common    // Common
  rf_common
  implicit_none_f90
  integer fileid

  rc_ncdecl
  nc_decls
  st_decls
  character*LINELEN description, program_version
  character*FILELEN tempfile    // local

  program_version = '$Id: reactionsetup.web,v 1.27 2000/04/01 17:34:02 dstotler Exp $'

  tempfile = filenames_array_reactionfile
  assert(tempfile ≠ char_undef)
  fileid = nccreate(tempfile, NC_CLOBBER, nc_stat)

  description = 'Data for reaction in degas 2'
  call ncattputc(fileid, NC_GLOBAL, 'description', NC_CHAR, string_length(description),
      description, nc_stat)

  call ncattputc(fileid, NC_GLOBAL, 'data_version', NC_CHAR, string_length(reaction_version),
      reaction_version, nc_stat)

  call ncattputc(fileid, NC_GLOBAL, 'program_version', NC_CHAR,
      string_length(program_version), program_version, nc_stat)

  rc_ncdef(fileid)
  call ncendef(fileid, nc_stat)
  rc_ncwrite(fileid)
  call ncclose(fileid, nc_stat)

  return
end
```

2 References

3 INDEX

assert: 1.2, 1.3.

b: 1.2.

char_undef: 1.2, 1.3.

datasetup: 1.
description: 1.3.
diskin: 1.2.

e: 1.2.
eof: 1.2.

`file`: 1.2.
FILE: 1.
fileid: 1.3.
FILELEN: 1.2, 1.3.
filenames_array: 1.2, 1.3.
`form`: 1.2.

i: 1.2.
implicit_none_f77: 1.2, 1.3.
implicit_none_f90: 1.2, 1.3.
int_unused: 1.2.

len: 1.2.
length: 1.2.
line: 1.2.
LINELEN: 1.2, 1.3.
loop1: 1.2.
loop2: 1.2.
loop3: 1.2.

NC_CHAR: 1.3.
NC_CLOBBER: 1.3.
nc_decls: 1.3.
NC_GLOBAL: 1.3.
nc_stat: 1.3.
nc_write_reaction: 1.3.
ncattputc: 1.3.
ncclose: 1.3.
nccreate: 1.3.
ncendef: 1.3.
next_token: 1.2.

p: 1.2.
parse_string: 1.2.
problemsetup: 1.
program_version: 1.3.

rc_common: 1.2, 1.3.
rc_emitter: 1.2.
rc_filename: 1.2.
rc_gen: 1.2.
rc_generic_no: 1.2.
rc_generic_yes: 1.2.
rc_ncdecl: 1.3.
rc_ncdef: 1.3.
rc_ncwrite: 1.3.
rc_num: 1.2.
rc_product: 1.2.
rc_product_max: 1.2.
rc_product_num: 1.2.
rc_reaction_type: 1.2.
rc_reagent: 1.2.
rc_reagent_max: 1.2.
rc_reagent_num: 1.2.
rc_sy: 1.2.
rc_type_len: 1.2.
rd_table_size: 1.2.
reaction_emitter: 1.2.
reaction_filename: 1.2.
reaction_generic: 1.2.
reaction_infile: 1.2.
reaction_name: 1.2.
reaction_product: 1.2.
reaction_product_num: 1.2.
reaction_reagent: 1.2.
reaction_reagent_num: 1.2.
reaction_sy: 1.2.
reaction_type: 1.2.
reaction_version: 1.2, 1.3.
reactionfile: 1.3.
read_reactions: 1.2.
read_string: 1.2.
read_text: 1.2.
rf_common: 1.2, 1.3.

sp_common: 1.2.
sp_lookup: 1.2.
st_decls: 1.2, 1.3.
`status`: 1.2.
string_length: 1.3.

tempfile: 1.2, 1.3.
type: 1.2.

`unit`: 1.2.

var_realloca: 1.2.
var_reallocb: 1.2.

⟨ Functions and Subroutines 1.2, 1.3 ⟩ Used in section 1.1.
⟨ Memory allocation interface 0 ⟩ Used in section 1.2.

COMMAND LINE: `"fweave -f -i! -W[ -ykw700 -ytw40000 -j -n/ /u/dstotler/degas2/src/reactionsetup.web"`.

WEB FILE: `"/u/dstotler/degas2/src/reactionsetup.web"`.

CHANGE FILE: `(none)`.

GLOBAL LANGUAGE: FORTRAN.