Callable ADAS

Martin O'Mullane
Department of Physics
University of Strathclyde

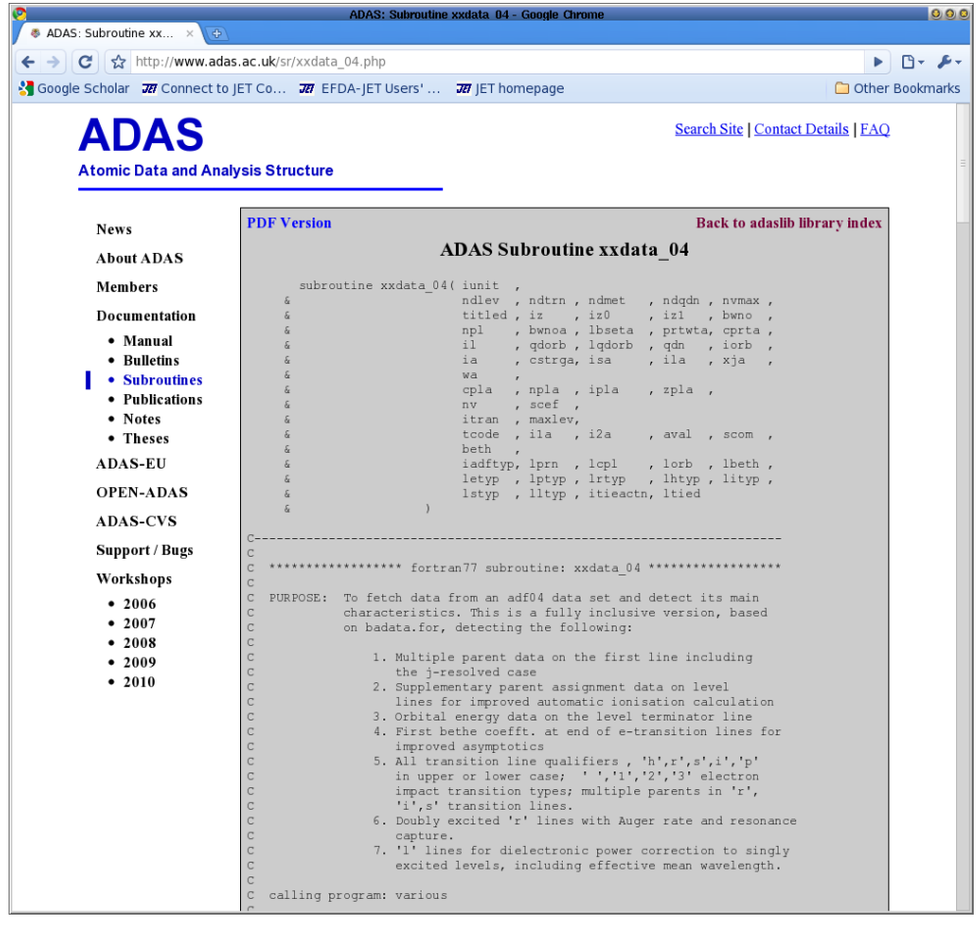
Callable ADAS

ADAS Comes with a Fortran library of over 750 routines:

- all documented in ∼1500 page appendix to the user manual,
- also documented on-line.

Also comes with extensive IDL library:

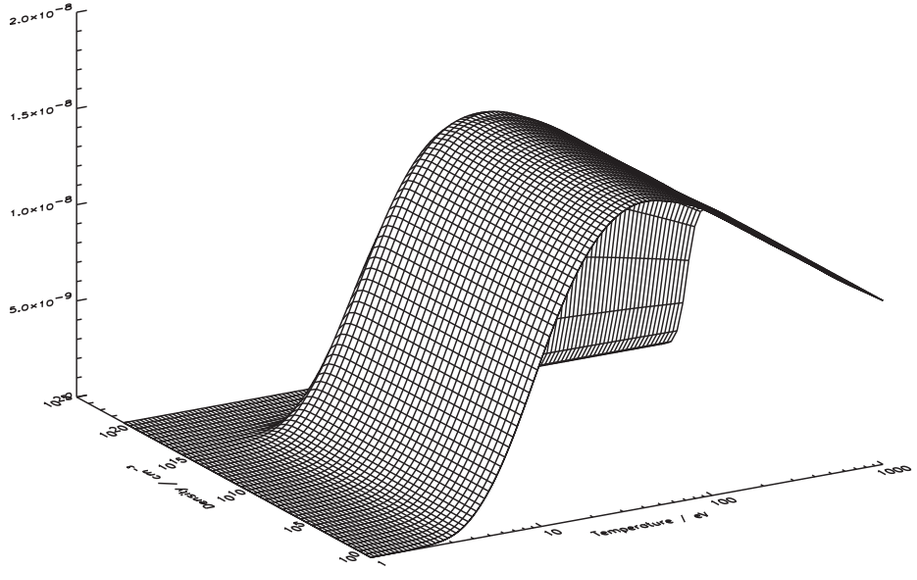
- far easier to use for interactive work at the command line,
- self documenting; almost all routines accept a '`/help`' keyword.



Partial/planned support for C, C++, (Matlab), Perl and Python access.

Reading ADAS data in your own programs

Reading the photon emissivity coefficient for the Ne^{7+} transition: $1\mathrm{s}^2 2\mathrm{p}\ {}^2\mathrm{P} - 1\mathrm{s}^2 2\mathrm{s}\ {}^2\mathrm{S}$

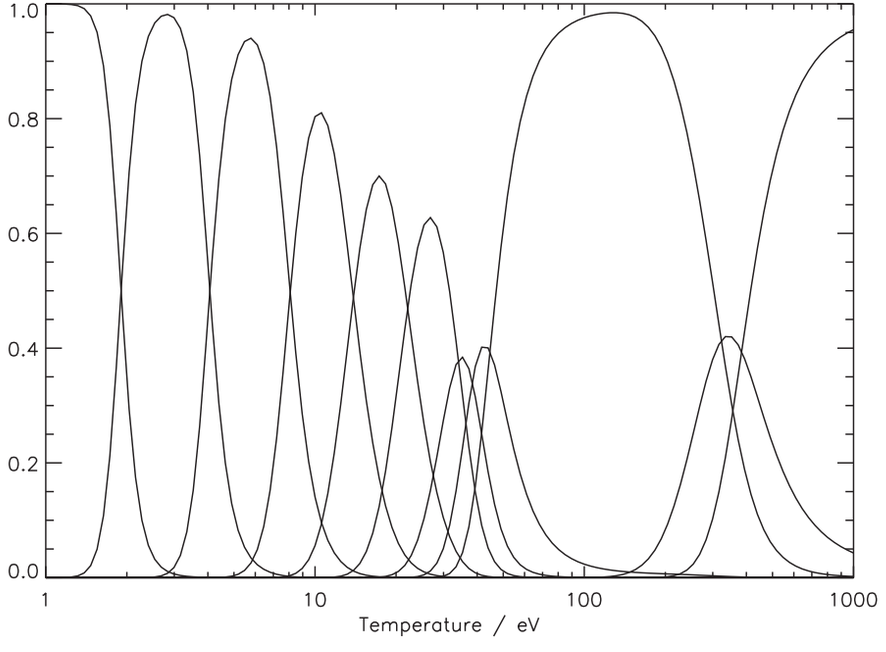



```
temp = adas_vector(low=1,high=1000, num=127)
dens = adas_vector(low=1,high=1e21, num=43)

read_adf15,file='/home/adas/adas/adf15/pec96#ne/pec96#ne_pju#ne7.dat',
           block=1,te=temp,dens=dens,data=data,/all

surface,data,temp,dens,/xlog,/ylog, $
        xtitle='Temperature / eV',ytitle='Density / cm!U-3'
```

Running ADAS codes in your own programs

```
temp = adas_vector(low=1,high=1000, num=127)
dens = adas_vector(low=1,high=1e21, num=43)
```

```
run_adas405,uid  = 'adas', $
            year = 96',    $
            elem = 'ne',   $
            te   = temp,   $
            dens = dens,   $
            frac = frac
```




```
plot, temp, frac.ion[*,0], /xlog, xtitle='Temperature / eV'

for i = 1, 10 do oplot, temp, frac.ion[*,i]
```

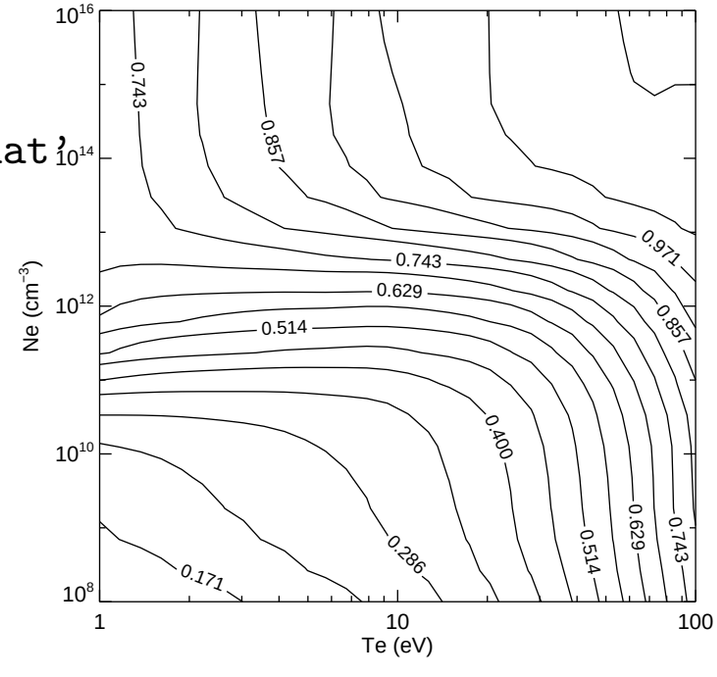
Line ratios without all that clicking!

The spectrum line emissivity is:

$$\begin{aligned} \varepsilon_{j\to i} &= N_e N_\sigma A_{j\to i} F^{exc}_{j\to i} \\ &= N_e N_\sigma \mathcal{PEC}^{exc}_{j\to i} \end{aligned}$$

So a line ratio is the ratio of the photon emissivity coefficients.

```
temp = adas_vector(low=1, high=100, num=30)
dens = adas_vector(low=1e8, high=1e16, num=20)
file = '/home/adas/adas/adf15/pec96#c/pec96#c_vsu#c0.dat'
read_adf15, file=file, block=1, te=temp, dens=dens, $
            data=e1, /all
read_adf15, file=file, block=2, te=temp, dens=dens, $
            data=e2, /all

contour, e1/e2, temp, dens, nlevels=20, /follow,    $
         /xlog, /ylog, $
         xtitle = 'Te (eV)', ytitle='Ne (cm!u-3!n)'
```

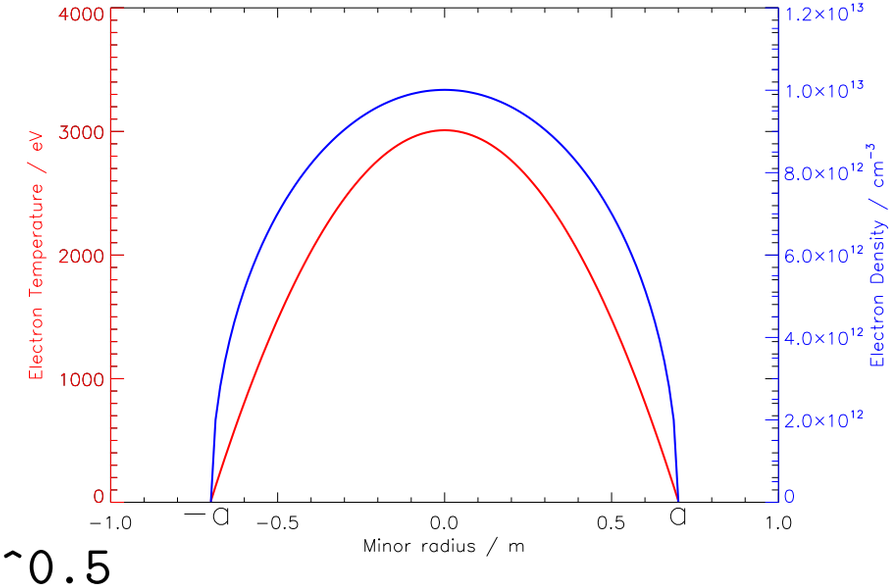


Example: midplane emission through a fusion device

Set up grid of radial points on the midplane:

```
a = 0.7
r = a*(findgen(101)/50.0 - 1.0)
```

Create simplified temperature and density profiles along the midplane:




```
temp = 10.0 + 3e3 * (1 - (r/a)^2)
dens = 1e10 + 1e13 * (1 - (r/a)^2)^0.5
```

Use ADAS to work out fractional ionisation balance along midplane:

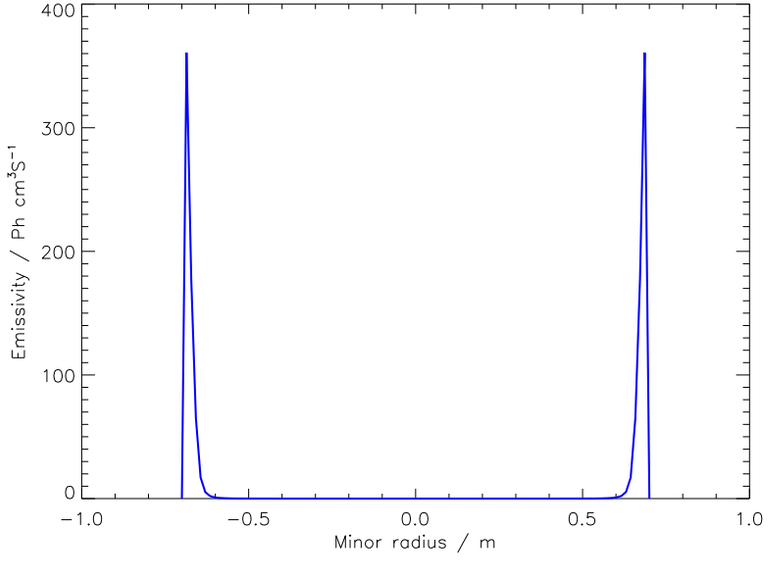
```
run_adas405, uid='adas', year=96, elem='ne', $
             te=temp, dens=dens, frac=frac
```

Read emissivity coefficient for $1s^2 2p\ ^2P - 1s^2 2s\ ^2S$ transition

```
read_adf15,file='/home/adas/adas/adf15/pec96#ne/pec96#ne_pju#ne7.dat'
           te=temp,dens=dens,block=1,data=coeff
```

Calculate the emissivity

```
emissivity = frac.ion[*,7] * $
             coeff * dens
```



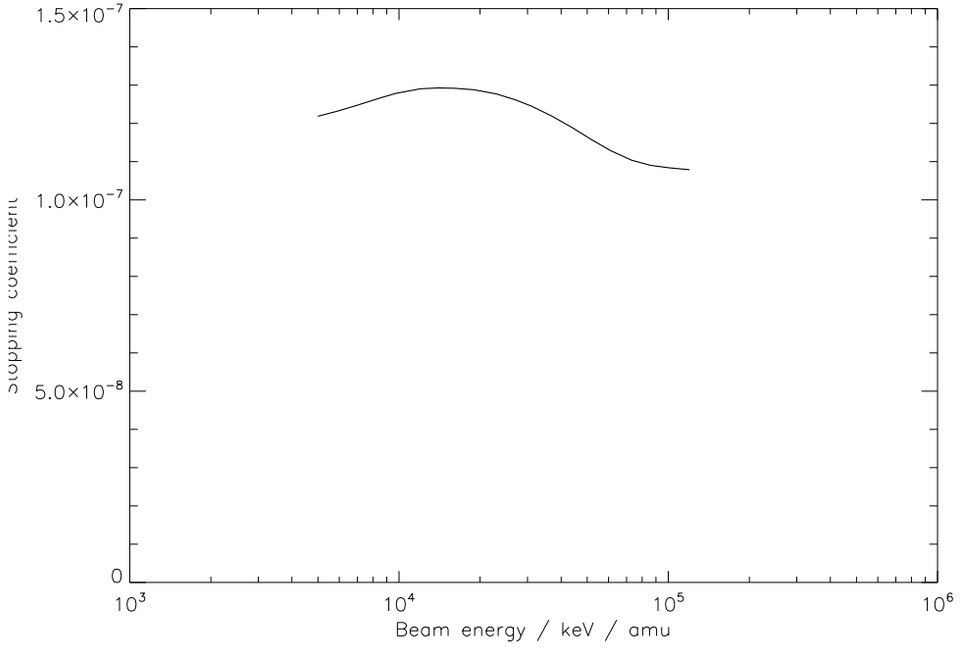
Calculate emission by integration

```
emission = int_tabulated(r,emissivity)
```

This example assumes ionisation balance equilibrium. We also need to adjust for lots of things to do real confrontation, such codes are necessarily complex...

Reading beam attenuation coefficients

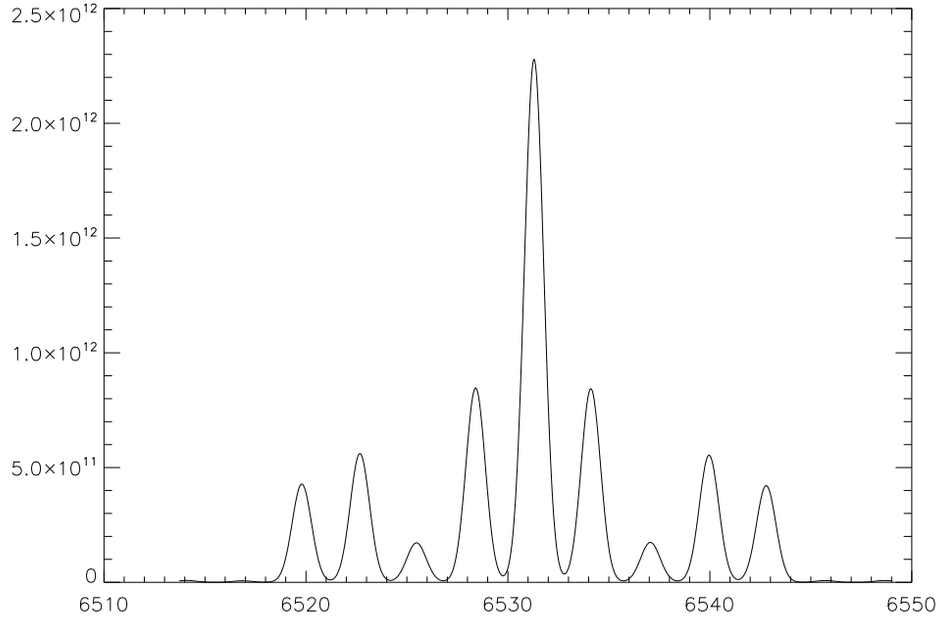
```
files = [ '/home/adas/adas/adf21/bms98#h/bms98#h_h1.dat' , $
          '/home/adas/adas/adf21/bms97#h/bms97#h_c6.dat' , $
          '/home/adas/adas/adf21/bms97#h/bms97#h_be4.dat'  ]

fraction = [0.96, 0.03, 0.01]

temp = fltarr(80) + 2000
dens = fltarr(80) + 1e13
ener = [ 5000,  5900,   7000,   8300, $
          9700, 12000,  14000,  16000, $
         19000, 23000,  27000,  31000, $
         37000, 44000,  52000,  61000, $
         73000, 86000, 101000, 120000  ]
```




```
read_adf21, files=files, data=data, $
            fraction=fraction, te=temp, dens=dens, energy=ener

plot, ener, data, /xlog, ytitle='Stopping coefficient', $
                         xtitle='Beam energy  (keV/amu)'
```

Producing a stark feature

```
pars = afg('stark',/parameters)
pars.broaden = 1
res = afg('stark',calculate=pars)
plot, res.wv, res.intensity
```



BEAM_MASS	FLOAT	2.00000
BEAM_ENERGY	FLOAT	40.0000
BEAM_TE	FLOAT	10.0000
BEAM_DENSITY	FLOAT	1.00000e+10
PLASMA_MASS	FLOAT	2.00000
PLASMA_TE	FLOAT	4440.00
PLASMA_DENSITY	FLOAT	2.50000e+13
PLASMA_ZEFF	FLOAT	2.00000
BEAM_DC_X	FLOAT	0.00000
BEAM_DC_Y	FLOAT	0.00000
BEAM_DC_Z	FLOAT	1.00000
BFIELD_VALUE	FLOAT	3.39150
BFIELD_DC_X	FLOAT	0.788000
BFIELD_DC_Y	FLOAT	0.00530000
BFIELD_DC_Z	FLOAT	0.615200
EFIELD_VALUE	FLOAT	0.00000
EFIELD_DC_X	FLOAT	1.00000
EFIELD_DC_Y	FLOAT	0.00000
EFIELD_DC_Z	FLOAT	0.00000
OBS_DC_X	FLOAT	0.870100
OBS_DC_Y	FLOAT	-0.0470000
OBS_DC_Z	FLOAT	0.490500
OBS_SIGMA	FLOAT	0.510000
OBS_PI	FLOAT	1.00000
BROADEN	INT	1

Fortran Example: Bremsstrahlung emission

Evaluate bremsstrahlung emission using ADAS.

Requires specification of ion and temperature/wavelength:

– We'll use C^{6+} at 3keV and 6000Å for this example.

```
       PROGRAM BREM
C-----------------------------------------------------------------------
C      DEFINE APPROPRIATE VARIABLES
C-----------------------------------------------------------------------
       IMPLICIT  NONE
C-----------------------------------------------------------------------
       INTEGER*4 IZ0  , IZ1
       REAL*8    WAVE , TEV , CONTFF , CONTIN
C-----------------------------------------------------------------------
C      SPECIFY WE'RE DEALING WITH FULLY STRIPPED CARBON
C-----------------------------------------------------------------------
       IZ0=6
       IZ1=6
C-----------------------------------------------------------------------
C      SPECIFY A TEMPERATURE OF 3keV and a WAVELENGTH OF 6000A
C-----------------------------------------------------------------------
       TEV=3000
       WAVE=6000
```

```
C-----------------------------------------------------------------------
C      USE ADAS TO CALCULATE EMISSION
C-----------------------------------------------------------------------
       CALL CONTINUO(WAVE   , TEV    , IZ0  , IZ1  ,
     &               CONTFF , CONTIN               )
C-----------------------------------------------------------------------
C      PRINT THE FREE-FREE EMISSIVITY (Ph cm3 s-1 A-1)
C-----------------------------------------------------------------------
       PRINT *,CONTFF

C-----------------------------------------------------------------------
C      PRINT THE TOTAL EMISSIVITY (Ph cm3 s-1 A-1)
C-----------------------------------------------------------------------
       PRINT *,CONTIN

       END
```

Compile with:

```
g77 brem.for -L/home/sipp/gipp/adas/lib -ladas3xx -ladaslib -o brem.x
```

Run to give the result:

```
4.54015816E-17
4.54120444E-17
```

IDL version: `continuo, 6000.0, 3000.0, 6, 6, contff, contin`

C Example: Bremsstrahlung emission

Direct C translation of previous Fortran example.

```
/*****************************************************************************
 Bremsstrahlung routine is in adaslib library so include appropriate header
*****************************************************************************/
#include "adaslib.h"

int main(int argc, char *argv)
{

/*****************************************************************************
        Declare appropriate variables
*****************************************************************************/
        int iz0, iz1;
        double wave,tev,contff,contin;
/*****************************************************************************
        Specify we're dealing with fully stripped carbon
*****************************************************************************/
        iz0=6;
        iz1=6;

/*****************************************************************************
        Specify a temperature of 3keV and a wavelength of 6000A
*****************************************************************************/
        wave=6000;
        tev=3000;
```

```
/*****************************************************************************
        Use ADAS to calculate emission
*****************************************************************************/
        continuo_(&wave   , &tev    , &iz0  , &iz1  ,&contff , &contin);

/*****************************************************************************
        Print off free-free emissivity (Ph cm3 s-1 A-1)
*****************************************************************************/
        printf("%e\n",contff);

/*****************************************************************************
        Print off total emissivity (Ph cm3 s-1 A-1)
*****************************************************************************/
        printf("%e\n",contin);

        return 0;
}
```

Compile with:

```
g77 -I/home/sipp/gipp/adas/include brem.c
-L/home/sipp/gipp/adas/lib -ladas3xx -ladaslib -o brem.x
```

Note that g77 is used for compilation even though this program is written in C.