Photoionization codes

Photoionization codes – first ones

- First models in late 60's by D. Flower, P. Harrington, B. Rubin among others.

- Photoionization codes can be seen as a complete summary of all the physics we know about the interaction between energetic radiation and interstellar gas.

**COMPUTER MODELS OF THE PLANETARY NEBULAE
NGC 7662 AND IC 418**

D. R. FLOWER

Department of Physics, University College, London, England

Discrepancies between the observed and calculated spectra of the nebulae are discussed. There is evidence for the importance of dynamical effects and filamentary structure and possibly for deviations of the flux of the central star of NGC 7662 from black-body values in the far ultra-violet.

Photoionization codes - today

- Today the more used photoionization code is certainly **Cloudy** (G. Ferland, P. van Hoof, R. Porter, R. Williams, W. Henney). It's a 200,000 lines C++ code (with 500,000 lines of data files).

- Some other 1D codes are also used (**PHOTO** by Stasinska, **NEBU** by Péquignot, **MAPPINGS** by Dopita and Binette)

- **MOCASSIN** (Ercolano) is a Monte-Carlo full 3D code

- Cloudy_3D/**pyCloudy** (IDL and Python versions, by Morisset) is a pseudo-3D code.

Photoionization codes – inputs and outputs

Description of the model (inputs)

- Ionizing SED (BB, stel. model, synth. Spectrum.) T^*, L^*, Z^*

- Gas distribution : $nH(r – x,y,z)$

- Chemical composition (spatial var?)

- Dust (spatial var?)

Atomic database

- Cross sections

- Recomb. coeffs.

- Collision strengths

- Einstein coeffs.

- Dust properties

PHOTOIONISATION CODE

- T_e, ne

- X^i / X $\quad\quad$ $(r – x,y,z)$

- Line emissivities

The temperature is an **output**

- Contrary to the H density, which is an input.
- Results of the thermal equilibrium.
- Obtained by GAINS = LOSS.
- GAINS: depend on ionizing photons mean energy (T*).

$$G(H^0) = n_e n_p \alpha_A(H^0, T) \frac{\int_{\nu_0}^{\infty} \frac{J_\nu}{h\nu} h(\nu - \nu_0) a_\nu(H^0) d\nu}{\int_{\nu_0}^{\infty} \frac{J_\nu}{h\nu} a_\nu(H^0) d\nu}$$

The temperature is an **<u>output</u>**

- Contrary to the H density, which is an input.

- Results of the thermal equilibrium.

- Obtained by GAINS = LOSS.

- GAINS: depend on ionizing photons mean energy (T*).

- LOSS: depend on line emission, especially on forbidden lines from metals, thus depend on Z.

$$4\pi\, j_{ul} = n_u A_{ul} h\nu_{ul} = n_l \frac{g_u}{g_l} e^{-h\nu_{ul}/kT} \left[1 + \left(\frac{A_{ul}}{n_e q_{ul}} \right) \right]^{-1} A_{ul} h\nu_{ul}$$
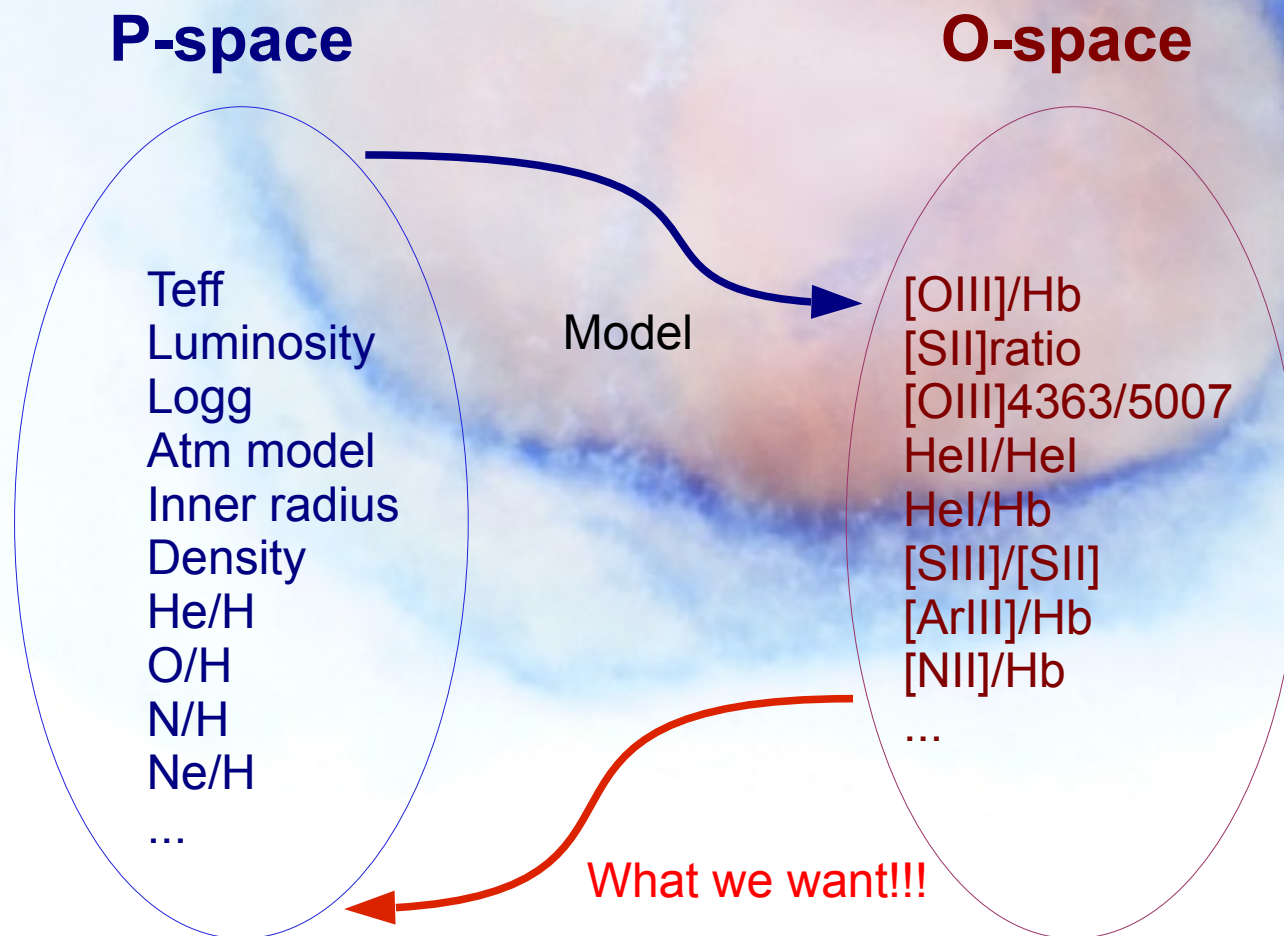
How to make a model ? - basics

Making a model of an given object is **finding** a (unique?) set of **input** parameters that are able to **reproduce all the observables:**

- Emission line ratios (Te- and ne-diagnostics, and also I/Ibeta).

- Absolute value of one emission line intensity.

- Images (shape and angular size – Distance).

- Continuum flux (radio – IR – Opt – UV – X)

- Line profiles, PV-diagrams (velocity field).

How to make a model ? P- and O-spaces

The modeling process can be seen as solving an inverse problem :

P-space

O-space

Teff
Luminosity
Logg
Atm model
Inner radius
Density
He/H
O/H
N/H
Ne/H
...

Model

[OIII]/Hb
[SII]ratio
[OIII]4363/5007
HeII/HeI
HeI/Hb
[SIII]/[SII]
[ArIII]/Hb
[NII]/Hb
...

What we want!!!

No linearity

P-space

Z

U

Semi-empirical analysis of Sloan Digital Sky Survey galaxies
How to distinguish AGN hosts

Grażyna Stasińska,[1]★ Roberto Cid Fernandes,[2] Abílio Mateus,[3] Laerte S[...]
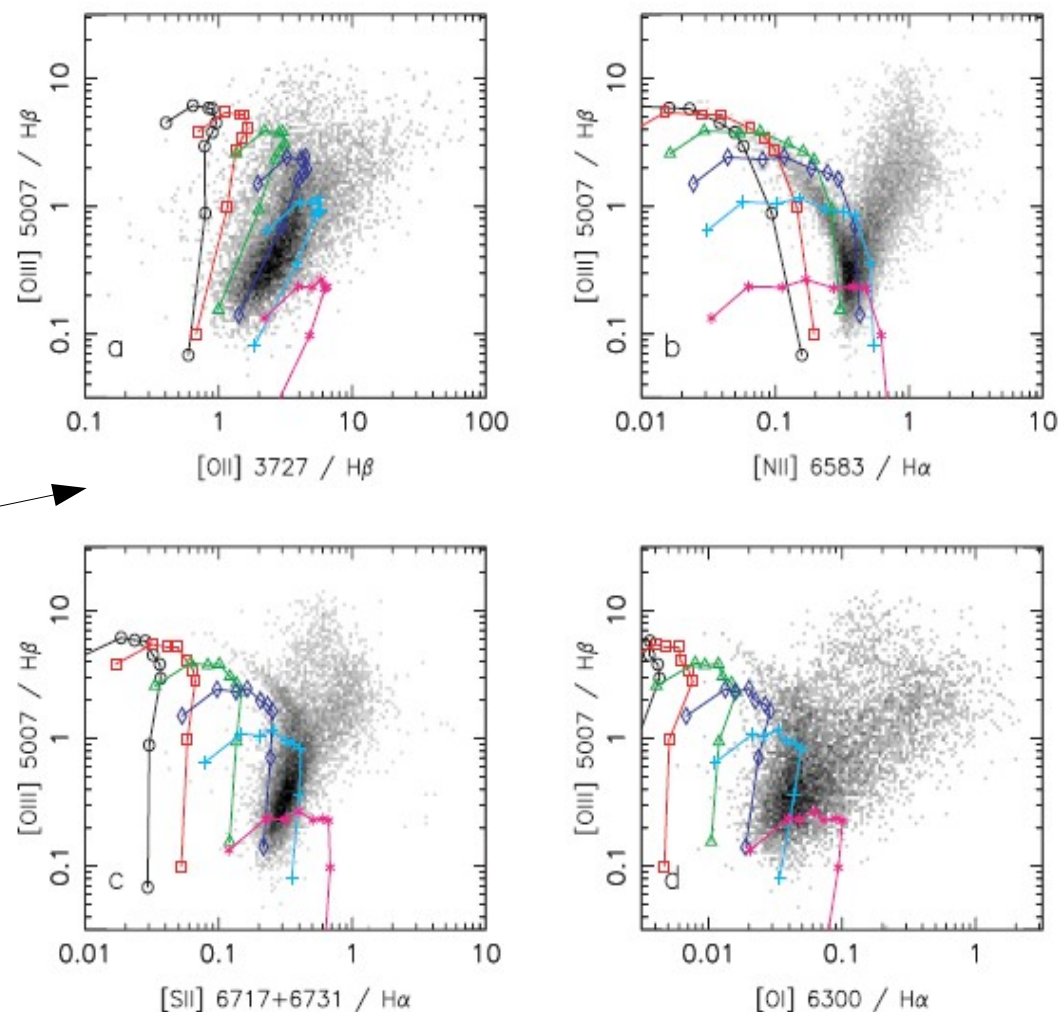and Natalia V. Asari[2]

Figure 2. Sequences of photoionization models with varying metallicities Z and constant ionization parameter U. The symbols on the curves correspond to the location of models with metallicities $Z = 0.1, 0.2, 0.3, 0.4, 0.6, 0.8, 1.0, 1.5$ and $2.5\,Z_\odot$, going from the upper left to the lower right (in panels b, c and d, the lowest metallicity models are actually outside the range of the plots). The values of the ionization parameter U are 10^{-2} (black circles), 5×10^{-3} (red squares), 2×10^{-3} (green triangles), 10^{-3} (blue diamonds), 5×10^{-4} (cyan + signs) and 2×10^{-4} (purple * signs).

Go to Nublado

- https://www.nublado.org/

- Contains downloads for the code (current and previous versions), hotfixes, help.

- https://cloudyastrophysics.groups.io/g/Main is the forum : report problems, ask for help.

- C17 version of Cloudy is described here : https://ui.adsabs.harvard.edu/abs/2017RMxAA ..53..385F/abstract

Short comments
on Cloudy/pyCloudy

Cloudy, short how-to

- Cloudy:
  - C++ code (200,000 lines!)
  - freeware
  - easy to install
  - very well supported by an active group of developers/users.
  - New version every 2-3 years (c17.02 just released)
  - Very detailed documentation (Quickstart and Hazy)

Cloudy, short how-to

- Cloudy.exe < model.in > model.out

- New form : cloudy.exe -p model

- Other test.*** files (radius, physics, ionic abundances, emissivities, etc)

Cloudy input file

- Ionizing source:
  - Shape (T[BB], T, Z, logg[Atm Mod], stellar cluster, power law, AGN)
  - Intensity (Luminosity, logU, Q0)
- Gas properties
  - Distribution (R_in, R_out, nH(r), ff)
  - Metallicity
  - Dust

Cloudy standart output

- The standart output from Cloudy is a collection of information about the model without a simple format, described in the Hazy manual.

Look at the output

- Control that everything is OK : check warnings, number of steps, and <u>why it stops</u> (this can be defined in the input file).

- Look at the mean ionization stages of the principal ions, mean temperature, heating and cooling sources.

- Some line intensities.

- More informations are available in other files.

Cloudy short how-NOT-to

- Use Cloudy to automatically find a solution, but not giving it the way to... by for example giving a small weight to weak lines like... [OIII]4363 !

The rms relative difference between the observed and modeled spectra was determined with

$$\text{rms}^2 = \frac{1}{N} \sum W_i \frac{(O_i - M_i)^2}{O_i^2}, \qquad (2)$$

where N is the number of lines included in the analysis, W_i is a line weight factor, and O_i and M_i are the observed and modeled line intensities. The line weight factor is equal to 1 when $O_i/\text{H}\beta > 0.5$, 0.5 if $O_i/\text{H}\beta \in [0.1, 0.5)$ and 0.25 if $O_i/\text{H}\beta \in [0.01, 0.1)$. Obviously, all hydrogen recombination lines are ex-

Parameter	A 79 Obs.	A 79 Model	BV 5-1 Obs.	BV 5-1 Model	JnEr 1 Obs.	JnEr 1 Model	M 1-41 Obs.	M 1-41 Model	NGC 2818 Obs.	NGC 2818 Model	Sh 1-89 Obs.	Sh 1-89 Model	Sh 2-71E Obs.	Sh 2-71E Model
[O II] $\lambda3727$	511	491	352	367	782	734	290	274	509	498	408	385	179	232
[Ne III] $\lambda3869$	155	155	163	163	115	112	102	101	155	155	108	112	133	133
Hδ $\lambda4101$	30.5	25.8	25.7	25.1	24.5	25.9	25.1	25.4	26.4	25.9	24.5	25.8	23.5	24.7
Hγ $\lambda4340$	48.8	46.8	48.3	46.5	57.0	46.9	46.0	46.7	48.0	46.9	45.2	46.8	43.4	46.3
[O III] $\lambda4363$	9.19	5.21	25.1	18.2	5.42	4.87	17.2	10.2	14.6	8.06	4.07	3.49	21.4	18.8
He I $\lambda4471$	7.19	10.2	6.12	5.99	5.15	7.36	...	4.83	4.40	6.15	4.75	7.16	...	5.96
He II $\lambda4686$	48.6	49.8	66.9	68.5	22.9	22.9	53.2	53.0	32.9	32.7	25.7	23.6	90.8	91.3
[Ar IV] $\lambda4740$	...	0.35	3.89	3.89	...	0.48	6.92	3.00	...	1.92	...	0.29	8.14	8.54
[O III] $\lambda5007$	444	443	951	923	721	820	653	654	808	747	668	691	833	838
[N I] $\lambda5200$	34.9	37.0	19.2	19.6	4.84	5.59	22.5	22.6	22.0	23.2	13.3	13.3	24.3	24.8
[Cl III] $\lambda5518$	...	1.34	2.59	2.36	...	0.87	...	1.65	...	1.21	...	0.72	...	2.13
[Cl III] $\lambda5538$	...	0.99	1.64	1.87	...	0.62	...	1.60	...	0.87	...	0.53	...	1.69
[N II] $\lambda5755$	36.1	34.7	26.4	25.0	7.96	6.94	26.1	22.6	19.3	17.3	7.38	8.19	33.8	34.8
He I $\lambda5876$	26.9	26.8	16.1	16.1	19.4	19.4	13.7	13.8	16.0	16.1	19.2	19.2	16.1	16.1
[O I] $\lambda6300$	98.8	12.5	43.6	10.0	27.3	11.4	43.8	13.7	47.0	16.7	46.3	13.6	25.4	6.58
[S III] $\lambda6312$	...	4.86	7.37	9.51	...	0.52	7.75	5.22	2.83	4.46	...	1.43	9.64	15.4
Hα $\lambda6563$	291	291	299	299	290	290	296	296	289	289	292	292	302	302
[N II] $\lambda6584$	2155	2050	1135	1101	633	584	1010	1011	1175	1191	793	813	1614	1615
He I $\lambda6678$	10.4	7.51	4.73	4.35	...	5.49	3.58	3.54	4.86	4.55	5.56	5.42	6.83	4.35
[S II] $\lambda6717$	145	144	99.0	98.1	16.4	16.8	48.7	48.7	114	115	47.7	48.5	106	107
[S II] $\lambda6731$	117	116	94.9	94.2	11.4	11.8	66.1	66.2	82.5	83.3	40.2	40.8	101	102
[Ar V] $\lambda7006$	...	...	2.10	0.36	...	...	5.80	0.17	...	0.51	...	...	...	1.82
[Ar III] $\lambda7136$	33.5	33.2	44.6	44.6	28.0	27.2	62.1	62.2	31.6	31.4	15.1	15.3	59.1	59.1
$C(H\beta)$	0.37	...	0.80	...	0.21	...	1.99	...	0.20	...	1.00	...	0.67	...
$T(N II)$	10937	10983	12702	12535	10996	10726	12997	12070	10884	10326	8625	8915	12046	12229
$T(O III)$	15529	12272	17528	15183	10524	9791	17461	13638	14599	11891	9862	9338	17284	16117
N_e	194	195	549	552	20	20	2059	2010	31	36	251	250	524	535
rms	...	0.10	...	0.07	...	0.08	...	0.12	...	0.13	...	0.08	...	0.12
$\log[L(H\alpha)]$	...	34.8	...	35.1	...	33.8	...	34.9	...	34.6	...	34.6	...	35.1
$D(L)$	...	6.5	...	7.1	...	0.9	...	1.3	...	3.2	...	1.9	...	5.4
$D(R)$	...	1.8	...	2.7	...	1.3	...	0.4	...	4.9	...	0.8	...	1.0
[C I] $\lambda9850$	...	6.95	...	1.08	...	14.6	...	11.3	...	4.65	...	87.8	...	1.81
[C II] $\lambda2326$	...	56.6	...	30.3	...	71.7	...	281	...	30.7	...	447	...	41.1
[C III] $\lambda1909$	...	106	...	166	...	119	...	1066	...	96.2	...	711	...	406
[C IV] $\lambda1549$	...	4.20	...	45.1	...	8.97	...	133	...	45.1	...	31.4	...	252
[N III] $\lambda1750$	...	72.2	...	264	...	7.53	...	120	...	9.68	...	9.68	...	707
[N III] 57.21 μm	...	295	...	207	...	216	...	67.4	...	361	...	275	...	405
[N IV] $\lambda1486$	...	3.80	...	70.7	...	1.00	...	20.3	...	41.4	...	0.93	...	488
[O IV] 25.88 μm	...	15.9	...	109	...	67.6	...	48.1	...	152	...	66.0	...	177
[Ne IV] $\lambda2424$	...	2.84	...	30.2	...	1.20	...	8.92	...	22.4	...	1.44	...	61.3
[Ne V] 14.32 μm	...	0.17	...	7.59	...	...	...	1.82	...	12.2	...	0.41	...	20.7
[S III] $\lambda9532$	...	131	...	175	...	21.7	...	111	...	134	...	67.5	...	262
[S IV] 10.51 μm	...	12.1	...	63.4	...	5.06	...	28.3	...	58.9	...	19.6	...	150
[Cl IV] $\lambda8047$	...	1.27	...	2.17	...	1.29	...	1.81	...	1.15	...	0.75	...	1.94

Wrong conclusion from wrong assumptions

    4. Models underestimate the observed $T(\text{O\,III})$ temperature in all cases but three (class 1 regions K 3-72, M 3-5, and Wray 16). This has not been observed in other works involving Cloudy modeling of PNs (i.e., van Hoof & Van de Steene 1999; Gonçalves et al. 2007; SDK07). Discrepancies are smaller among most class 1 objects, and in 10 out of 12 cases these differences are within the observational error range reported in B01 and B03. Temperature discrepancies are much larger and beyond the observational error range among class 2 objects. This suggests that this inconsistency may be connected to aperture effects; real slits do not usually contain the entire in-depth extent of the ionized region, but these models cannot account for this and assume full coverage.

pyStuffs

- pyCloudy
  - Python package to manage Cloudy inputs and outputs,
  - Easy grids,
  - Pseudo-3D facility,
  - Related to PyNeb and 3MdB

Why 3D ?
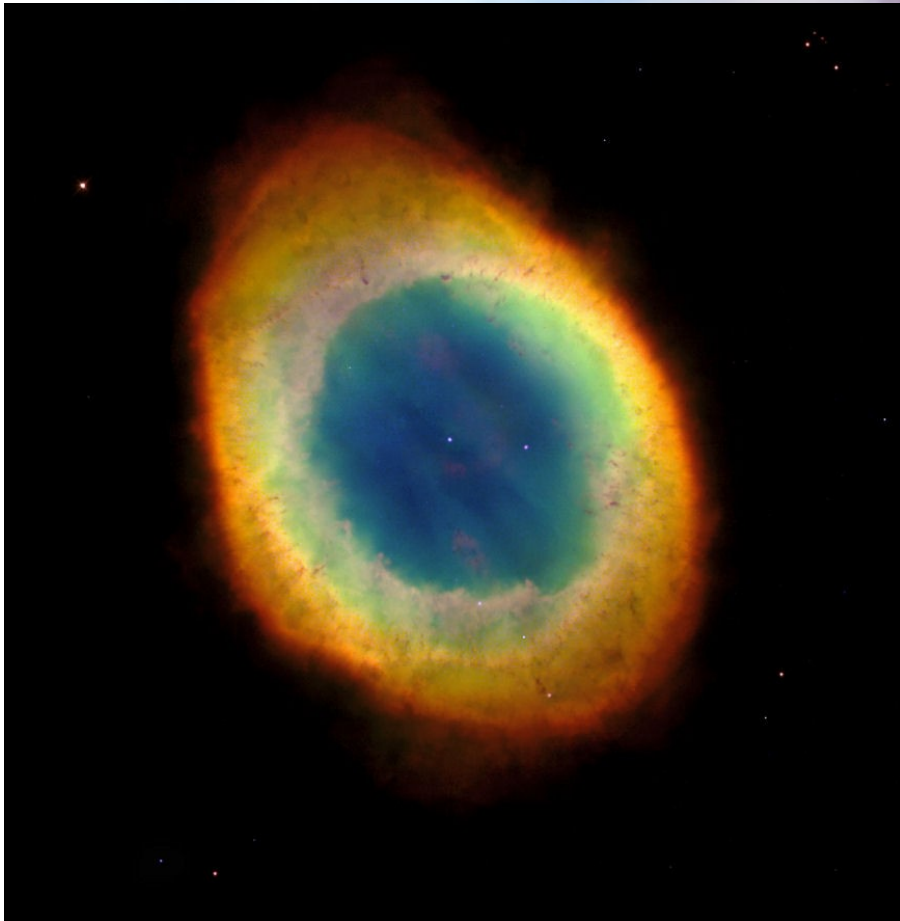
- As we all know, PN are spherical, and HII regions plan parallels :-)

Planetary Nebula IC 418

Hubble Heritage

Not O nor //
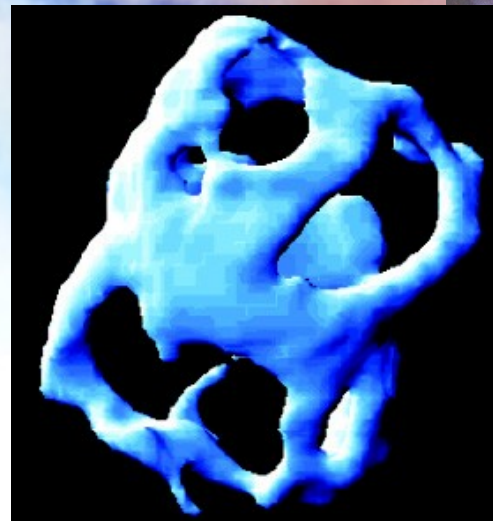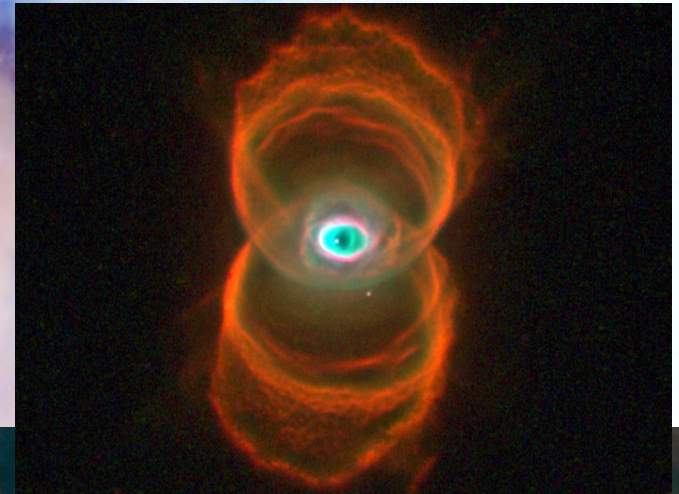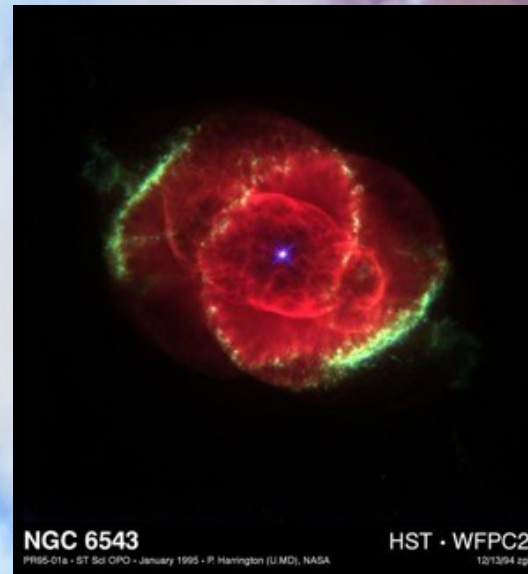
Aperture effects

- Color = dominating emission line.

- Position : different line ratios.

- Consequences on ICFs for example.

MOCASSIN and pyCloudy 3D

- MOCASSIN : A full 3D photoionization code, using Monte-Carlo.

- F90 MPI code, running on cluster.

- Few hours to days to run/converge a model (cluster allocation time policies).

- → need for a quick (but not so dirty) code to obtain « pseudo-3D » models

- Cloudy_3D : we're loosing the « full » 3D, but a few minutes to run:-)

PyCloudy 3D

Modelling of aspherical nebulae – I. A quick pseudo-3D photoionization code

C. Morisset,[1★] G. Stasińska[2] and M. Peña[1]

- Various runs of Cloudy (1D), corresponding to different angles.

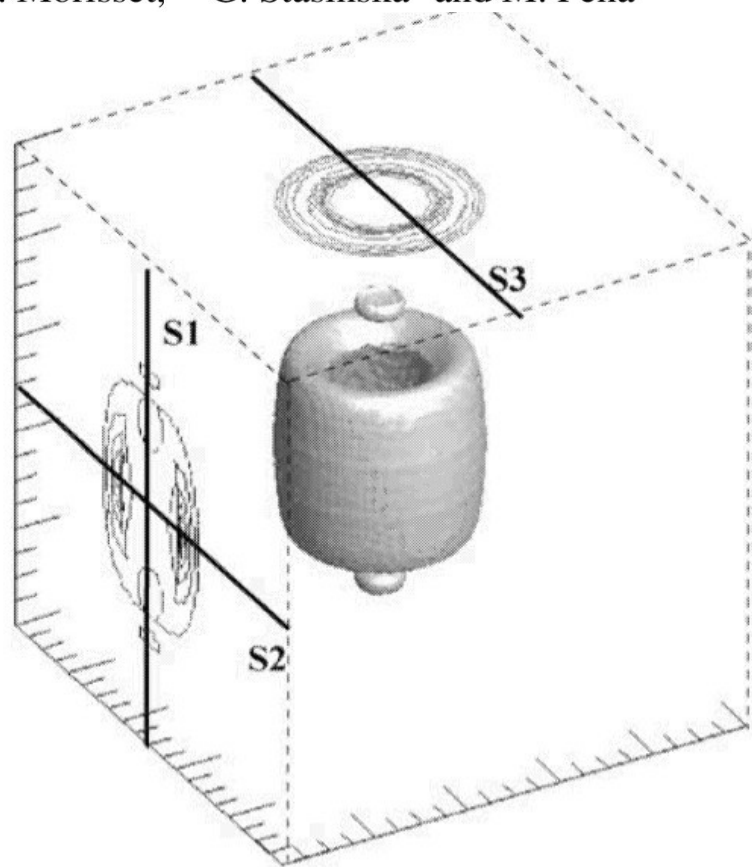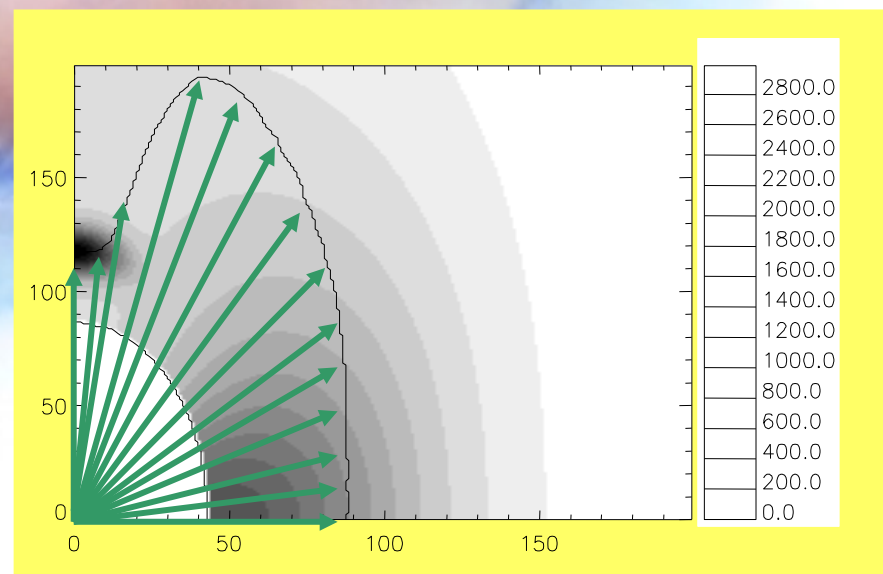- 3D reconstruction in a coordinate cube by interpolation between the 1D runs.

Figure 2. 3D representation of the nebula. An isodensity surface is drawn, showing the equatorial density enhancement and the two polar knots. On the faces we have represented the Hβ surface brightness contours for two orientations of the nebula: on the left side, for the nebula seen with the polar axis parallel to the sky; on the top, for the nebula seen pole-on. The slits used to determine parameters from emission-line ratios are indicated.

PyCloudy 3D: faster but limited ?

- pyCloudy 3D is not a « full » 3D code. It only considers radial radiation (as a combination of 1D runs).
- Limitation : when no-radial radiation dominates a process (e.g. Photoionization).
- 2 cases :
  - Shadows
  - Multiple stars

Otherwise : pseudo 3D is OK !

Most of the « complicated » morphologies are easily modeled with a simple 2- or 3-components **topologically equivalent** model.

Using pyCloudy

- pyCloudy is actually a python library running and dealing with the results of Cloudy.

- First implementation (Cloudy_3D) using IDL.

- The new version (2012) is in **Python**.

- Object-oriented.

- Can also be used for a simple 1D run, or for grids.

Run a model

- A quite simple object is used to define and write the Cloudy input file. It allows to automatically add the various « save » options, with the predefined extensions.

- Only the most common options are predefined, but all the others are also available « by hand ».

Read a model

- Another object reads the outputs of a Cloudy run and fill variables with the radius-dependent parameters.

- Methods are available to compute some integration over the radius or the volume (weighted by different parameters).

- T0(line), T0(ion), t2(line), t2(ion) are computed.

- Model can be cut (Rmin, Rmax).