Laboratorio di Astrofisica 2

Radial velocities and planet detection

Esperienza di laboratorio

Luca Malavolta

luca.malavolta@unipd.it

RV experience

Sampler: *emcee* (Foreman-Mackey+2013)

Code: PyORBIT (Malavolta+2016)

Model: Keplerian RVs + RV jitter + RV offset + Transit Time

The code is hosted on GitHub

To install the code:

```
$ git clone https://github.com/LucaMalavolta/PyORBIT_students
$ cd PyORBIT_students
$ cd LabAf2_RVdata
```

The code is split in two: the *sampler* part (`*_emcee.py`), and the output analysis (`*_GetResults.py`)

You need to set up three files:

- `STAR_1st_run.yaml`: input parameters for PyORBIT
- `STAR_RVs.dat`: radial velocities of your planet host star
- `tran_1p.dat`: transit time of your planet (from your experience)

Retrieve the RVs from literature

- Go to NASA Exoplanet Archive and look for your star http://exoplanetarchive.ipac.caltech.edu/
- Check the papers with quoted (minimum) mass reference

Planet Parameters						
Planet	M sin(i)		Mass			Reference
	(Jupiter Mass)	(Earth Mass)	(Jupiter Mass)	(Earth Mass)	(Solar Radii)	
b	null	null	null	null	null	Holczer et al. 2016
b	null	null	null	null	$0.0138^{+0.0005}_{-0.0005}$	Morton et al. 2016
b	null	null	null	null	0.01350±0.00024	Van Eylen & Albrecht 2015
b	null	null	$0.0145^{+0.0040}_{-0.0046}$	$4.61^{+1.27}_{-1.46}$	$0.01357^{+0.00045}_{-0.00027}$	Esteves et al. 2015
b	null	null	null	null	null	Hu et al. 2015
b	null	null	0.010±0.002	3.33±0.49	$0.0135^{+0.0003}_{-0.0002}$	Dumusque et al. 2014
b	null	null	0.0145±0.0040	4.60±1.26	0.0134±0.0002	Fogtmann-Schulz et al. 2014
b	null	null	0.014±0.004	$4.56^{+1.17}_{-1.29}$	0.0130±0.0003	Batalha et al. 2011
c	null	null	null	null	null	Holczer et al. 2016
c	null	null	null	null	$0.0213^{+0.0008}_{-0.0007}$	Morton et al. 2016
c	null	null	null	null	0.02129±0.00026	Van Eylen & Albrecht 2015
c	null	null	null	null	null	Kipping et al. 2015
c	null	null	0.054±0.006	17.2±1.9	$0.0216^{+0.0008}_{-0.0004}$	Dumusque et al. 2014
c	null	null	null	null	0.0213±0.0003	Fogtmann-Schulz et al. 2014
c	null	null	<0.063	<20	0.0204±0.0005	Fressin et al. 2011

Setting up the RV file

rv_data_1p.dat

```
6000.009195  22.353737 1.000000  0     0
6001.189929   9.736558 1.000000  0     0
6002.012133   2.839680 1.000000  0     0
6003.035998  -6.573851 1.000000  0     0
6004.030169  -9.105770 1.000000  0     0
6004.926995 -13.803967 1.000000  0     0
6005.991056 -17.539292 1.000000  0     0
6006.919202 -24.898170 1.000000  0     0
6007.969370 -27.433760 1.000000  0     0
```

- Tip: you can use the jitter and offset flag, or use a separate file for each instrument and the flags set to zero

Column	Value
1	RV epochs (Barycentric Julian Date or BJD-2450000)
2	RV measurements [ms^{-1}]
3	RV errors [ms^{-1}]
4	RV jitter flag (+1 for each instrument starting from zero)
5	RV offset flag (+1 for each instrument starting from zero)

Setting up the T0 file

tran_1p.dat

```
0 6067.227011 0.001000
```

Column	Value
1	ID of the transit (starting from 0)
2	Central time of transit (BJD or BJD-2450000, *must be consistent with RV file)*
3	Associated error

- Tip: transit times are associated to a specific planet, name your file properly according if you are working on a multi-planetary system

Setting up the PyORBIT file

FYE: for your experience

HAT-P-12_1st_run.yaml

```yaml
Name: HAT-P-12
Output: HAT-P-12_1st_run
Inputs:
  0:
      File: HAT-P-12_RVs.dat
      Kind: RV
      Models: ['kepler']
Planets:
  0:
    Orbit: keplerian
    Boundaries:
      P: [2.00, 5.00]
      K: [0.00, 100.0]
      e: [0.00, 0.800]
    Priors:
      P: ['Gaussian',3.213, 0.001]
    Starts:
      P: 3.213
    Tcent: tran_1p.dat
    Inclination: [89.0, 0.4]
emcee:
  Nsteps: 4000
  Nburn:   2000
  Npop_mult: 2
  Thin: 1
  Recenter_Bounds: True
Star_Mass: [0.733, 0.018]
```

- **`Name`**: your planet
- **`Output`**: use a different one for each run
- **`Inputs`**: list the files starting from zero. FYE always use **`Kind:RV`** and **`Models:['kepler']`**.
- **`Planets`**: list the planets starting from zero. It is useful to start form the shorter-period ones
  - **`Orbits`**: FYE always use **`keplerian`**
  - **`Boundaries`**: specify the boundaries for Period **`P`**, eccentricity **`e`**, RV semi-amplitude **`K`**
  - **`Priors`**: specify the probability distribution of the parameter prior and its characteristics
  - **`Starts`**: starting points for the MCMC chains. If not specified, the code will use the median point of the boundaries
  - **`Tcent`**: the file with time of transit must be declared here and not in **`Inputs`** because it is specific of this planet
  - **`Inclination`**: if not provided, the code will output the minimum mass

Setting up the PyORBIT file

FYE: for your experience

HAT-P-12_1st_run.yaml

```yaml
Name: HAT-P-12
Output: HAT-P-12_1st_run
Inputs:
  0:
      File: HAT-P-12_RVs.dat
      Kind: RV
      Models: ['kepler']
Planets:
  0:
    Orbit: keplerian
    Boundaries:
      P: [2.00, 5.00]
      K: [0.00, 100.0]
      e: [0.00, 0.800]
    Priors:
      P: ['Gaussian',3.213, 0.001]
    Starts:
      P: 3.213
    Tcent: tran_1p.dat
    Inclination: [89.0, 0.4]
emcee:
  Nsteps: 4000
  Nburn:   2000
  Npop_mult: 2
  Thin: 1
  Recenter_Bounds: True
Star_Mass: [0.733, 0.018]
```

- **`Tref`**: if not provided, the code will choose the mid-point of the first dataset. These value is common to all the datasets
- **`emcee`**: these parameters control the behaviour of the MCMC sample
  - **`Ngen`**: not relevant FYE
  - **`Nsteps`**: total number of steps of each chain
  - **`Nburn`**: number of steps to be removed before computing the posterior. Can be changed after running the code
  - **`Npop_mult`**: the number of *walkers* is given the number of dimensions of the problem multiplied for **`Npop_mult`**. It cannot be <2.
  - **`Thin`**: thinning factor, must be $\geq$ the autocorrelation time
  - **`Recenter_Bounds`**: FYE leave it True
- **`Star_mass`**: used at the moment of extracting the posterior, like **`Nburn`** can be changed after running the sampler.

How to run the code and read the output

Run the sampler:

$ python ../PyORBIT_V3_emcee.py STAR_1st_run.yaml

Wait… then retrieve the results

$ python ../PyORBIT_V3_GetResults.py STAR_1st_run.yaml

```
Acceptance Fraction for all walkers:
[ 0.4184  0.43    0.4031  0.3988  0.4186  0.422
  0.4203  0.4154  0.4077  0.4116  0.4121]

Autocorrelation time all walkers:
[ 20.34390528  18.44974664   0.08539351   2.74776106   1.03565284
   3.11692771   5.83522747]


*************************************************************


rv_data_1p.dat jitter   :   [ 0.22903394]
rv_data_1p.dat offset   :   [ 0.04545595]

Tc  Planet_0
Input Tc:  17.227011   Model Tc:  17.2267066588   Diff:  -0.000304341197904
```

Acceptance fraction for each walker, and auto correlation time for each variable

Jitter and offset for each input dataset

Difference between the observed transit time and the one computed using the median of the posterior of each parameter

How to run the code and read the output

```
LNprob median =  -101.434200071

Tref:  6050.0

***********************************************************

rv_data_1p_jitter 0.229033940343  +\sigma  0.15283748734  -\sigma  0.152816812852
rv_data_1p_offset 0.0454559466181  +\sigma  0.0746873412738  -\sigma  0.0720845439886
P 5.06701108113  +\sigma  0.000225866272818  -\sigma  0.000229172283928
K 5.81972888208  +\sigma  0.00300010027523  -\sigma  0.00291176913467
f 3.91120628389  +\sigma  0.00201791895582  -\sigma  0.00231578687982
ecoso 0.539289961421  +\sigma  0.0010266090312  -\sigma  0.00112318762378
esino -0.257335403406  +\sigma  0.0029949718502  -\sigma  0.00312407942777

***********************************************************

Planet  Planet_0  summary

Period =  33.5214135393  +\sigma  0.00524847540925  -\sigma  0.00532445774006
K      =  56.4823763386  +\sigma  0.117577932854  -\sigma  0.113882546206
phase  =  3.91120628389  +\sigma  0.00201791895582  -\sigma  0.00231578687982
e      =  0.357002179153  +\sigma  0.00111707329864  -\sigma  0.00103469404076 , <  0.357493329021
o      =  -0.44534949671  +\sigma  0.00533587353145  -\sigma  0.00535493195097
Mass_J =  0.8376767796  +\sigma  0.0263621732508  -\sigma  0.0295872512719
Mass_E =  266.23881086  +\sigma  8.37868952431  -\sigma  9.40371607175
Tperi  =  6026.75697425  +\sigma  0.0309939978952  -\sigma  0.0278713087364
Tcent  =  6067.22699784  +\sigma  0.000971585680418  -\sigma  0.000965012226516
a      =  0.203492151063  +\sigma  0.00321515971781  -\sigma  0.00367106034325
Planet  Planet_0  completed
```

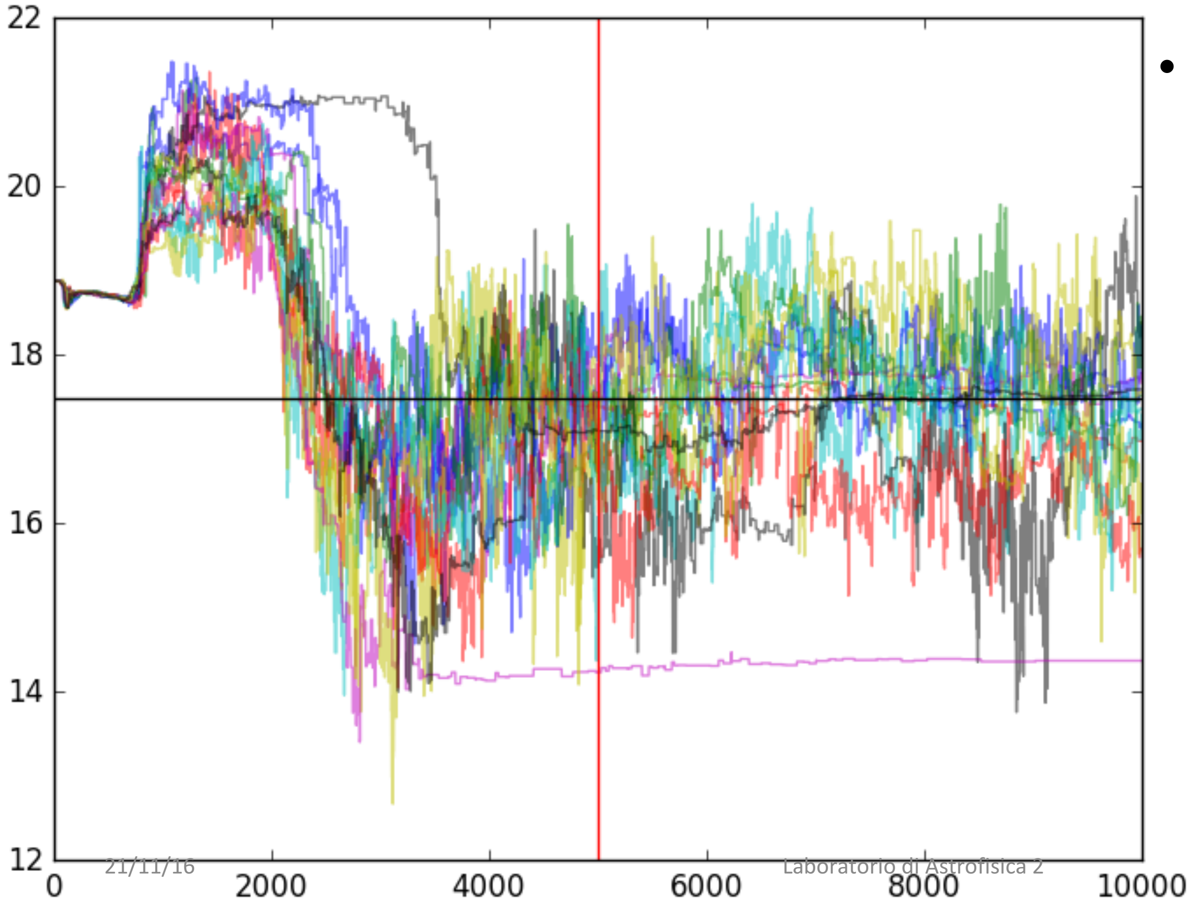
Internal *emcee* variables

Output parameters: median $\pm 1\sigma$ confidence intervals (using 15.865, 50, 84.135 percentiles)

How to run the code and read the output

You need to run the code *twice*!

- Use small values for **`Nburn`** (e.g. 2000) and **`Nsteps`** (e.g. 4000). Set **`Thin`** = 1 (no thinning). Run the sampler and then retrieve the results.
- Observe the chains (**`chain_*png`** files in **`./*_1st_run`**) and choose a proper value for the burn-in and the total length. Take note of the autocorrelation time.



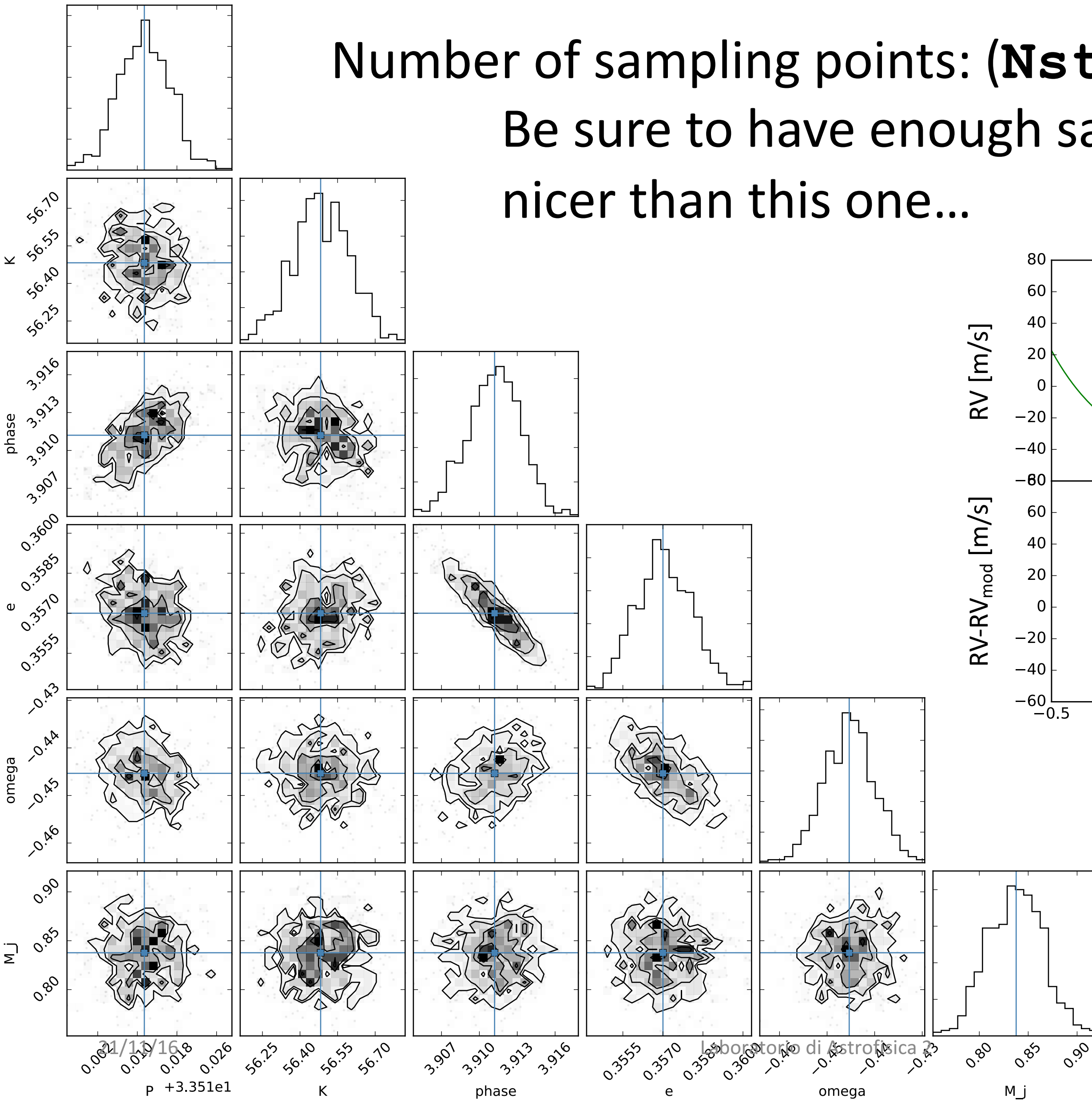
- Open **`*_2nd_run.yaml`** Change **`Nburn`**, **`Nsteps`** and **`Thin`** (= autocorrelation time) and run the code again

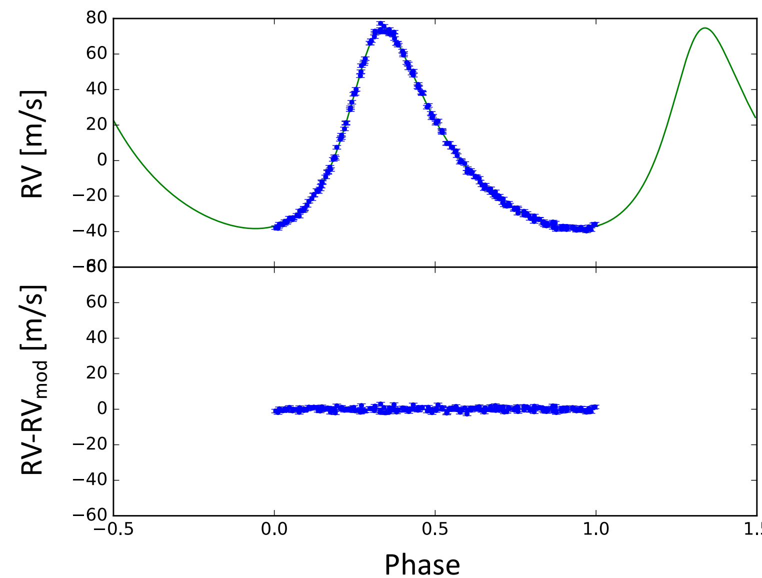
NB: the chains are plotted with the thinning factor already applied

How to run the code and read the output

Number of sampling points: (**`Nsteps-Nburn)/Thin`**)

Be sure to have enough samples to make a plot nicer than this one…





Phase = (t-T$_0$) % P

Plot the result

Inside the `*/_2nd_run/files_plot folder`:

- **`Planet_`*`n`*`_kep.dat`**: epochs and RVs to plot a full RV model curve
- **`Planet_`*`n`*`_pha.dat`**: phase an RVs to plot a RV model curve
- **`Planet_`*`n_dataset`*`.dat_kep.dat`**: RV values for each dataset

Column	Value
1	BJD of the observation (BJD or BJD-2450000, as given in input)
2	Phase of the observation, relative to the period of the planet
3,4	Observed RVs with associated errors
5,6	Observed RVs after removing the offset + a. e.
7,8	Observed RVs after removing offset and all the planets except planet *n* + a. e.
9,10	RV residuals after subtracting the full model + a. e.
11,12	RV model computed at the epochs of the datasert