Bayesian Parameter Inference and Model Comparison: the Eclipse of 1919

Alan Heavens, ICIC, Imperial College London
a.heavens@imperial.ac.uk
DISCNET Summer School 25 July 2019
Instructions and ancillary files are at `https://github.com/afheavens/Data-Analysis-Workshop`

1 Testing gravity with gravitational lensing

The aim of this exercise is to determine whether Einstein's General Theory of Relativity or Newton's theory of gravity is favoured by data from photographic plates taken during Eddington's 1919 eclipse expedition. Light from stars that passes close to the Sun is deflected by measurable amounts, and during an eclipse, this light can be detected and the stars' displacements measured, when compared with photographs taken when the Sun is far away. Einstein's theory predicts that the light will be bent by twice as large an angle (1.75 arcseconds for light grazing the Sun) as Newton's theory.

There are four exercises here, and any subset can be attempted. I would be surprised (pleasantly!) if anyone can do all four in the day. If you have written an MCMC code before, I suggest that you skip Project 1.

1. Write a Monte Carlo Markov Chain (MCMC) code to infer the bending angle (at the limb of the Sun)
2. Use the MCMC chains to compute the Bayesian Evidence for Newton's theory, and for Einstein's theory, and hence the odds ratio.
3. Take the MCMC chains, and use machine learning techniques to learn approximately the form of the posterior probability, and compute the odds analytically from the fit.
4. Estimate the Bayes Factor for the two gravity models by estimating the Savage-Dickey density ratio from the chains.

Pre-computed MCMC chains are available if you want to skip the first task. For projects 2-4, the output is the Bayes Factor. You should get the same answer from all methods.

2 Background

General Relativity predicts that light passing a mass M at distance r will be bent through an angle

$$\theta_{\rm GR}(r) = \frac{4GM}{rc^2}, \tag{1}$$

whereas an argument based on Newtonian gravity gives half this:

$$\theta_{\rm N}(r) = \frac{2GM}{rc^2}. \tag{2}$$

We can either treat this as a *parameter inference* problem, modelling the bending as

$$\theta(r) = \frac{\alpha GM}{rc^2}, \tag{3}$$

and inferring α, or as a *model comparison* problem (see later). For the first project, we do the former.

The data *model* looks very simple, with a single *parameter* α, as it is fine to assume that GM, c and r are known perfectly, from measurements of planet orbits etc. However, the experiment has some other 'nuisance' parameters as well, which we investigate below.

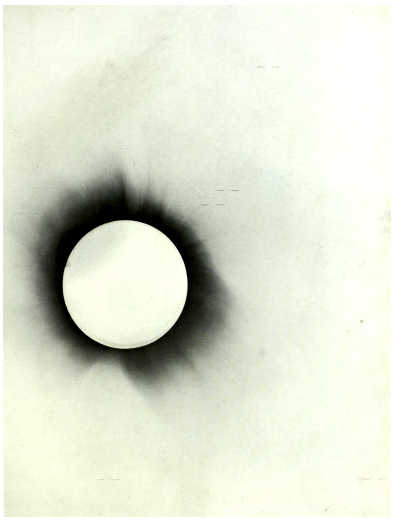

Figure 1: One of the photographs from Eddington's expedition.

3 Data

The Dyson, Eddington and Davidson 1920 paper (Phil. Trans. R. Society, 220, 571) is open access at
`http://rsta.royalsocietypublishing.org/content/roypta/220/571-581/291.full.pdf`
and contains the data we need. Some figures in this document are taken from there.

4 Data model and model parameters

A displacement of the star images will be caused not just by bending of light, but by changes in the scale of the photographic plate (caused e.g. by changes in temperature), rotation of the plate with respect to the comparison plate, and offsets. We are not very interested in these effects, which introduce *nuisance parameters*, but we need to include them. The data model is then given by the equations in Fig 2, i.e.

$$\begin{aligned} Dx &= ax + by + c + \alpha E_x \\ Dy &= dx + ey + f + \alpha E_y. \end{aligned} \tag{4}$$

Thus there are 4 parameters for the displacement in x, and 4 for the y displacement. Only the light bending parameter, α is common. $E_{x,y}$ are the theoretical displacements if the bending strength is $\alpha = 1$. The data use units measured on the plate, and you will do the same. Dyson et al. do not give errors in the displacements. Assume they are 0.05 in the units given.

The positions of the stars are shown in Fig. 3, or you can read them in the table in Fig. 2, which also gives the expected bend angle (at a distance of 50') if $\alpha = 1$. This is in a strange unit, so your number α will need to be rescaled to translate it to arcseconds at the limb. **To translate the inferred value of α from the units they used to the bending angle for light grazing the Sun, in arcsec, you need to multiply it by 19.8.**

The displacements are measured and shown in Fig. 4, for 7 plates (I,II,...VIII; VI was not used.)

However, there is a complication, in that, for technical reasons, it was not possible to compare an eclipse plate directly with the comparison plate, taken with the Sun elsewhere in the sky. Instead, both plates were compared with a reference (so-called 'scale') plate, so to these displacements it is necessary to add the displacements of the comparison plate to the scale plate. These are given in Fig. 5. So, for example, if you are using Plate II, and comparison plate 14_{2a}, the displacement that should be entered into your code for Dx for star 11 is $(-1.416{+}1.500){+}(-0.478{+}0.552)$.

19. The values of Dx and Dy were equated to expressions of the form

$$ax + by + c + a\mathrm{E}_x \ (= \mathrm{D}x)$$

and

$$dx + ey + f + a\mathrm{E}_y \ (= \mathrm{D}y),$$

where x, y are the co-ordinates of the stars given in Table I., and E_x, E_y are coefficients of the gravitational displacement.

The quantities c and f are corrections to zero, depending on the setting of the scale plate on the plate measured, a and e are differences of scale value, while b and d depend mainly on the orientation of the two plates. The quantity a denotes the deflection at unit distance (*i.e.*, 50′ from the sun's centre), so that $a\mathrm{E}_x$ and $a\mathrm{E}_y$ are the deflection in R.A. and Decl. respectively of a star whose co-ordinates are x and y.

The left-hand sides of the equation for the seven stars shown are :—

No.	Right Ascension.	Declination.
11	$c-0{\cdot}160b-1{\cdot}261a-0{\cdot}587x$	$f-1{\cdot}261d-0{\cdot}160e+0{\cdot}036x$
5	$c-1{\cdot}107b-0{\cdot}160a-0{\cdot}557x$	$f-0{\cdot}160d-1{\cdot}107e-0{\cdot}789x$
4	$c+0{\cdot}472b+0{\cdot}334a-0{\cdot}186x$	$f+0{\cdot}334d+0{\cdot}472e+1{\cdot}336x$
3	$c+0{\cdot}360b+0{\cdot}348a-0{\cdot}222x$	$f+0{\cdot}348d+0{\cdot}360e+1{\cdot}574x$
6	$c+1{\cdot}099b+0{\cdot}587a+0{\cdot}080x$	$f+0{\cdot}587d+1{\cdot}099e+0{\cdot}726x$
10	$c+1{\cdot}321b+0{\cdot}860a+0{\cdot}158x$	$f+0{\cdot}860d+1{\cdot}321e+0{\cdot}589x$
2	$c-0{\cdot}328b+1{\cdot}079a+1{\cdot}540x$	$f+1{\cdot}079d-0{\cdot}328e-0{\cdot}156x$

Figure 2: The data model.

TABLE I.

No.	Names.	Photog. Mag.	Co-ordinates. Unit = 50′.		Gravitational displacement.			
					Sobral.		Principe.	
			x.	y.	x.	y.	x.	y.
		m.			″	″	″	″
1	B.D., 21°, 641	7·0	+0·026	−0·200	−1·31	+0·20	−1·04	+0·09
2	Piazzi, IV, 82	5·8	+1·079	−0·328	+0·85	−0·09	+1·02	−0·16
3	κ^2 Tauri	5·5	+0·348	+0·360	−0·12	+0·87	−0·28	+0·81
4	κ^1 Tauri	4·5	+0·334	+0·472	−0·10	+0·73	−0·21	+0·70
5	Piazzi, IV, 61	6·0	−0·160	−1·107	−0·31	−0·43	−0·31	−0·38
6	υ Tauri	4·5	+0·587	+1·099	+0·04	+0·40	+0·01	+0·41
7	B.D., 20°, 741	7·0	−0·707	−0·864	−0·38	−0·20	−0·35	−0·17
8	B.D., 20°, 740	7·0	−0·727	−1·040	−0·33	−0·22	−0·29	−0·20
9	Piazzi, IV, 53	7·0	−0·483	−1·303	−0·26	−0·30	−0·26	−0·27
10	72 Tauri	5·5	+0·860	+1·321	+0·09	+0·32	+0·07	+0·34
11	66 Tauri	5·5	−1·261	−0·160	−0·32	+0·02	−0·30	+0·01
12	53 Tauri	5·5	−1·311	−0·918	−0·28	−0·10	−0·26	−0·09
13	B.D., 22°, 688	8·0	+0·089	+1·007	−0·17	+0·40	−0·14	+0·39

Figure 3: The x, y positions of the stars measured. Note that only seven stars are used, numbers $11, 5, 4, 3, 6, 10, 2$.

4.1 Data to use

I recommend that (at least to start) you use Dx only, from Plate II only, and use comparison plate 14_{2a}. Then it will be easier to compare results amongst yourselves.

5 Project 1: MCMC analysis of the eclipse data

For this exercise, you can either write your own MCMC code (using either the Metropolis-Hastings algorithm or Hamiltonian Monte Carlo), for which some details are provided here, or, if you prefer, to use a statistical programming language such as Stan to do the hard work for you.

TABLE II.—Eclipse Plates—Scale.

No. of Star.	I.		II.		III.		IV.		V.		VII.		VIII.	
	Dx.	Dy.	Dx.	Dy.	Dx.	Dy.	Dx.	Dy.	Dx.	Dy.	Dx.	Dy.	Dx.	Dy.
	r	r	r	r	r	r	r	r	r	r	r	r	r	r
11	−1·411	−0·554	−1·416	−1·324	+0·592	+0·956	+0·563	+1·238	+0·406	+0·970	−1·456	+0·964	−1·285	−1·195
5	−1·048	−0·338	−1·221	−1·312	+0·756	+0·843	+0·683	+1·226	+0·468	+0·861	−1·267	+0·777	−1·152	−1·332
4	−1·216	+0·114	−1·054	−0·944	+0·979	+1·172	+0·849	+1·524	+0·721	+1·167	−1·028	+1·142	−0·927	−0·930
3	−1·237	+0·150	−1·079	−0·862	+0·958	+1·244	+0·861	+1·587	+0·733	+1·234	−1·010	+1·185	−0·897	−0·894
6	−1·342	+0·124	−1·012	−0·932	+1·052	+1·197	+0·894	+1·564	+0·798	+1·130	−0·888	+1·125	−0·838	−0·937
10	−1·289	+0·205	−0·999	−0·948	+1·157	+1·211	+0·934	+1·522	+0·864	+1·119	−0·820	+1·072	−0·768	−0·964
2	−0·789	+0·109	−0·733	−1·019	+1·256	+0·924	+1·177	+1·373	+0·995	+0·935	−0·768	+0·892	−0·585	−1·166
	−1·500*	−0·554	−1·500	−1·324	+0·500	+0·843	+0·500	+1·226	+0·400	+0·861	−1·500	+0·777	−1·300	−1·322

Figure 4: The displacements Dx and Dy, as measured on 7 plates (I,II,... VII), with respect to a reference 'scale' plate. Note that you have to subtract the number at the bottom of each column - e.g. add 1.5 to the numbers in the first column.

COMPARISON Plates—Scale.

No. of Star.	14_{2x}.		14_{2y}.		15_1.		15_2.		17_1.		17_2.		18_2.	
	Dx.	Dy.	Dx.	Dy.	Dx.	Dy.	Dx.	Dy.	Dx.	Dy.	Dx.	Dy.	Dx.	Dy.
	r	r	r	r	r	r	r	r	r	r	r	r	r	r
11	−0·478	−0·109	+0·967	+1·170	+1·098	+1·228	+0·725	+0·830	−1·073	−1·330	+1·242	−0·302	−1·188	−1·572
5	−0·544	−0·204	+1·013	+1·192	+0·899	+1·232	+0·692	+0·938	−1·072	−1·075	+1·161	−0·224	−1·195	−1·432
4	−0·368	−0·136	+1·030	+1·249	+1·133	+1·086	+0·725	+0·854	−1·296	−1·031	+1·354	−0·281	−1·165	−1·454
3	−0·350	−0·073	+1·044	+1·305	+1·164	+1·114	+0·732	+0·893	−1·278	−1·014	+1·342	−0·261	−1·178	−1·394
6	−0·317	−0·144	+0·980	+1·319	+1·244	+1·012	+0·714	+0·824	−1·375	−1·052	+1·363	−0·390	−1·165	−1·473
10	−0·272	−0·146	+0·997	+1·327	+1·249	+0·960	+0·722	+0·831	−1·424	−1·038	+1·370	−0·423	−1·164	−1·476
2	−0·396	−0·182	+1·102	+1·289	+0·969	+1·052	+0·734	+0·941	−1·236	−0·909	+1·278	−0·328	−1·164	−1·335
	−0·552*	−0·206	+0·967	+1·170	+0·899	+0·960	+0·690	+0·824	−1·424	−1·330	+1·161	−0·423	−1·195	−1·572

* The numbers −1·500, −0·554, &c., given below the line, were taken out to make the values of Dx, Dy small and positive for arithmetical convenience.

Figure 5: The displacements of the comparison plate to the scale plate. As before, subtract the number at the bottom of each column.

5.1 Write your own MCMC

Write an MCMC code to sample from the posterior joint probability of a, b, c and α, using the Dx displacement data for Plate II, and comparison plate 14_{2a}. As mentioned, the units are weird, so multiply α by 19.8 to translate it to the bending inferred at the limb of the Sun (in arcsec), and compare with the Newtonian result (0.875) and the GR result (1.75). Note that Dyson et al. make a correction of about 10% because of errors arising from measuring indirectly the displacements, but we will ignore that here.

- Assume uniform priors on the parameters (so you compute only the likelihood). What do you assume for the form of the likelihood?
- You might like to start with a very simple 'top-hat' proposal distribution, where the new point is selected from a 4D 'rectangular' region centred on the old point. For this you will need a simple random number generator. Alternatively use a gaussian for each parameter.
- Explore visually the chain when you have (a) a very small proposal distribution, and (b) a very large proposal distribution, for a maximum of 1000 trials. What do you conclude?
- Show how the acceptance probability changes as you change the size of the proposal distribution from very small (say 0.0001) to very large (say 10).
- Once you have settled on a 'reasonable' proposal distribution, compute the average value of the parameters under the posterior distribution, and their variances and covariance.
- Marginalise over the nuisance parameters and plot the posterior distribution of α.
- Now see what happens if you don't consider the nuisance parameters (i.e. set them all to zero, or include a gaussian prior on each one, that keeps them close to zero). What would you conclude about the light bending at the limb of the Sun, and what would be your conclusion about gravity physics?

6 Extensions

- Include some more data, e.g. Dy for Plate II, or more plates.
- Extend to perform Hamiltonian Monte Carlo. You might like to try to compare the performance of MCMC and HMC; you will need to decide what the right criterion is.
- On another day, write and apply a Gelman-Rubin convergence test. Convergence tests are really necessary in practice.

For HMC, the algorithm is (from Hajian 2006):

Hamiltonian Monte Carlo

1: initialize $\mathbf{x}_{(0)}$
2: for i = 1 to $N_{samples}$
3: $\mathbf{u} \sim \mathcal{N}(0,1)$ (Normal distribution)
4: $(\mathbf{x}^*_{(0)}, \mathbf{u}^*_{(0)}) = (\mathbf{x}_{(i-1)}, \mathbf{u})$
5: for j = 1 to N
6: make a leapfrog move: $(\mathbf{x}^*_{(j-1)}, \mathbf{u}^*_{(j-1)}) \rightarrow (\mathbf{x}^*_{(j)}, \mathbf{u}^*_{(j)})$
7: end for
8: $(\mathbf{x}^*, \mathbf{u}^*) = (\mathbf{x}_{(N)}, \mathbf{u}_{(N)})$
9: draw $\alpha \sim$ Uniform(0,1)
10: if $\alpha < \min\{1, e^{-(H(\mathbf{x}^*,\mathbf{u}^*)-H(\mathbf{x},\mathbf{u}))}\}$
11: $\mathbf{x}_{(i)} = \mathbf{x}^*$
12: else
13: $\mathbf{x}_{(i)} = \mathbf{x}_{(i-1)}$
14: end for

$H = -\ln L + K$, where $K = \mathbf{u} \cdot \mathbf{u}/2$. One approach is to approximate U by a bivariate gaussian with covariances estimated from the MCMC code (or you can try computing the derivative exactly):

$$U = \frac{1}{2}(\theta - \theta_0)_\alpha C^{-1}_{\alpha\beta}(\theta - \theta_0)_\beta. \tag{5}$$

You can evolve the system with the Euler method (not recommended for stability reasons), or the leapfrog algorithm:

$$\begin{aligned} u_i\left(t + \frac{\epsilon}{2}\right) &= u_i(t) - \frac{\epsilon}{2}\left(\frac{\partial U}{\partial x_i}\right)_{\mathbf{x}_{(t)}} \\ x_i(t+\epsilon) &= x_i(t) + \epsilon u_i\left(t + \frac{\epsilon}{2}\right) \\ u_i(t+\epsilon) &= u_i\left(t + \frac{\epsilon}{2}\right) - \frac{\epsilon}{2}\left(\frac{\partial U}{\partial x_i}\right)_{\mathbf{x}_{(t+\epsilon)}}. \end{aligned} \tag{6}$$

Issues to consider are how many integration steps per point in the chain, and how big those steps are. For some discussion, see Hajian (2006), astroph/0608679.

If you wish to use the chains for projects 2 and/or 3, then you should save the chain in an ascii file, with each row containing (assuming that you are using as your data Dx only)

$$w, -\ln L, \alpha, a, b, c$$

Note the minus sign in front of $\ln L$. The weight is an integer, and should be 1 unless the point in the chain is repeated, in which case it increases by one for every rejection. The MCEvidence code in Project 2 will fail if you repeat identical lines in the file (for rejected points); you have to change the weight instead.

6.1 Alternative: PyStan

As an alternative, you can use Stan, and its python interface PyStan, to do the job quite painlessly. Stan uses a sophisticated HMC sampler. You specify the data, the parameters, and the model, and read or specify the data using python, send it to Stan, and then extract the chain from Stan back into python, where it can be post-processed. For details of PyStan and Stan, see https://pystan.readthedocs.io/en/latest/

If you do MCMC this way, then if you want to set the nuisance parameters to zero, it is much easier simply to specify a very tight prior on them with Stan, rather than changing the number of parameters, e.g. adding a line

$$\mathrm{a} \sim \mathrm{normal}(0.0, 0.001);$$

to force a to be (essentially) zero.

7 Project 2: Computing Bayesian Evidence from the MCMC chains

The denominator in the form of Bayes' theorem used for parameter inference is $Z \equiv p(\mathrm{data}|\mathrm{model})$ and normalises the posterior. It is given by

$$Z = \int d\theta \, p(\mathbf{d}|\theta, M) \, p(\theta|M)$$

where θ are the model parameters, **d** the data, and the model dependence M is explicitly included.

The posterior probability of a model is proportional to Z (assuming uniform priors on the models considered), so the Bayes Factor $B = Z_{\mathrm{Einstein}}/Z_{\mathrm{Newton}}$ is the posterior relative probability of the two models. For some discussion and notes on Bayesian Evidence, see the ICIC workshop notes from Roberto Trotta at

`https://www.imperial.ac.uk/media/imperial-college/research-centres-and-groups/astrophysics/public/icic/data-analysis-workshop/2018/Model_comparison_TrottaSept_2018.pdf`

7.1 Bayesian Evidence from MCMC chains

There are specialised software packages, such as MultiNest and PolyChord, that are designed to compute Z, but one can also compute it from MCMC chains that have been created to perform parameter inference.

An MCMC chain is designed to deliver a set of samples with a density n that is proportional to the target distribution (in this case the posterior). To compute the Bayesian Evidence, it is necessary to integrate the joint distribution $p(\theta, \mathbf{d})$ over the parameters, and this requires the constant of proportionality to be known. Nearest-neighbour distances can be used to estimate the Bayesian Evidence (and indeed to estimate the full posterior for Z). MCEvidence is a code that does this. See https://arxiv.org/pdf/1704.03472 for details.

The python code can be installed using instructions at

https://github.com/yabebalFantaye/MCEvidence

To use it, you will need to prepare two chains for the Newtonian model N, and two for the Einstein model E, or pick up pre-prepared chains from the website. The filenames need to end in 0 and 1 (to conform with chains produced using CosmoMC), and there are chains of different length available: Examples are:

```
Eclipse_chain_100000E_0
Eclipse_chain_100000E_1
Eclipse_chain_100000N_0
Eclipse_chain_100000N_1
```

which each have length 100000. Other lengths are available. A call to MCEvidence can take the form

```
chainfile_E='Eclipse_chain_100000E_*'

MLE_E=MCEvidence(chainfile_E,split=True,kmax=0,verbose=1,priorvolume=1.,thinlen=0.5,
burnlen=0,debug=False).evidence()

Z_Einstein=math.exp(MLE_E[0])
```

8 Project 3: Gaussian Mixture Models for approximating the posterior

Here we will use sklearn's gaussian mixture model (GMM) python routines to approximate the posterior distribution described by your chains (or those provided on the website) by a mixture of multivariate gaussians. Then, for the simplest case of a single gaussian, the Bayes factor will be computed from the approximate posterior distribution. The best routine is

`sklearn.mixture.BayesianGaussianMixture`

You may wish to start by reading in only two columns of the chains, and plotting a scatter plot with the output of the GMM superimposed, to check it makes sense. Then you can fit a 4D gaussian for α, a, b, c. See Fig. 6. With a

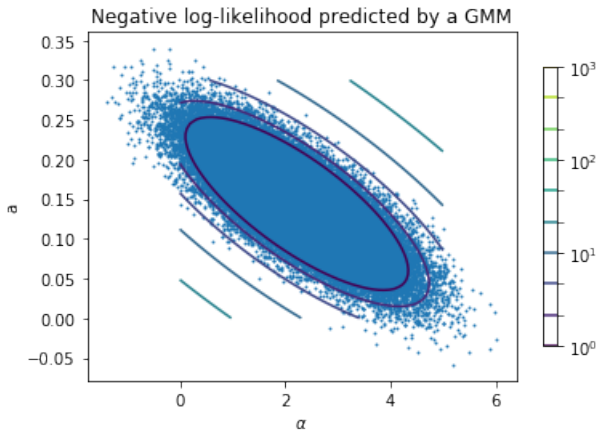

Figure 6: 2D scatter plot of α, a, and overplotted contours from a single gaussian fit. Note that the contours are plotted only within a smaller rectangle.

single gaussian fit, the posterior is given by

$$p(\mathbf{v}|\mathbf{d}) = \frac{1}{\sqrt{|2\pi\Sigma|}} \exp\left[-\frac{(\mathbf{v}-\bar{\mathbf{v}})^T\Sigma^{-1}(\mathbf{v}-\bar{\mathbf{v}})}{2}\right]$$

where $\mathbf{v} = (\alpha, a, b, c)$ and the routine returns the mean $\bar{\mathbf{v}} = (\bar{\alpha}, \bar{a}, \bar{b}, \bar{c})$ and the covariance Σ.

You may like to prove that fixing α at a value α_* and integrating over the other parameters gives a Bayesian Evidence Z (up to a constant that is the same for both $\alpha_* = 1.75$ and $\alpha_* = 0.875$),

$$\ln Z = 0.5[(\alpha_* - \bar{\alpha})^2 B^T D^{-1} B - A] + \text{constant}$$

where $D_{ij} = (\Sigma^{-1})_{ij}$ for $i, j = 1, 2, 3$, $B_i = (\Sigma^{-1})_{0i}$ and $A = (\Sigma^{-1})_{00}$.

How would you generalise to a mixture of Gaussians?

9 Bonus Project 4 (very quick)

Since it is cheap to produce long chains for this example, one can estimate the Bayes factor by counting the number of samples in the chains that are close to $\alpha = 1.75$ and $\alpha = 0.875$. This is essentially the Savage-Dickey density ratio (see Roberto's notes above for details of this). Take a long chain, and find the ratio of the number of points within some small tolerance of the two special values, and divide.