

PART IB MATHEMATICAL AND COMPUTATIONAL BIOLOGY.

PROJECT ONE. COULD THE RESULT OF THE 2024 UEFA EUROPEAN FOOTBALL CHAMPIONSHIP HAVE BEEN PREDICTED IN ADVANCE?

Summary

This project explores how FIFA World Rankings can be used to drive a simple statistical model of an international football match. By using this model in simulations of an entire football tournament, we try to understand the extent to which the outcome of Euro 2024 was predictable in advance.

Structure of the project

The project has two parts.

Part One (65% of the credit) is intended to introduce certain aspects of model fitting and simulation required by Part Two, in which you will fit and apply a model to Euro 2024. Since the tournament has already occurred, in Part Two you are asked to assess predictions of your model against real results.

Your write up for Part One should mainly consist of a single Jupyter notebook. This should include any written answers – and Python code – created in responding to the questions. However, in answering Question 3, you should not feel you need to typeset the mathematics in Jupyter (unless you want to learn the markup language required to do this). Therefore, any mathematical calculations associated with your answer to Question 3 may be submitted in a supplementary pdf document. Converting to pdf should be possible if you use word processing software to typeset calculations or even prefer simply to handwrite your work and take a photograph.

Part Two (35% of the credit) is more open ended. Your code for this part should **not** be handed in and will **not** be assessed. We will only assess a written summary of what you have done, focusing on your results (and how they are presented) and the quality of your analysis. The write up for Part Two should be submitted as a single pdf document, which should be **no longer than 5 pages of A4** (11-point font).

Both parts of the project must be handed in by **5pm on Weds 29th January 2025**. Submission will be done electronically via Moodle; we will provide further details about how to do this closer to the time.

Obtaining help

While submissions must be your own work, following the advice on avoiding plagiarism in the Course Handbook on Moodle, provision has been made for you to obtain help.

The main way to get help is to attend one (or more) of the “Drop In” sessions. These start in Week 7 of Michaelmas term; a full list of dates is given in the Course Handbook. You can also feel free to ask a demonstrator in a “normal” practical, should you have time to spare.

Within the constraint that what is submitted must be your own work, you are also free to seek support with problems, e.g., around getting a particular section of code to work, from others on the course, or from anyone else you know who already knows Python well.

How much time should you spend over the project?

The three assessed projects for this course should take around 60 hours to do in total. However, this first project is awarded slightly less credit (10% *versus* 15% for each of the other two), and so we feel this project should take slightly less time: 15 hours would be proportionate to the credit awarded.

Nik Cunniffe, October 2024

PART ONE. INTRODUCTION (65% OF THE MARKS ARE GIVEN TO PART ONE)**Modelling football games using a Poisson model**

A simple model of the number of goals scored by each team in a football game uses a Poisson distribution.

To introduce the idea, we initially imagine the parameters λ_A and λ_B are pre-specified and control the mean number of goals (per 90-minute game) scored by Team A and Team B, respectively.

Our model is simple. We just assume the number of goals scored by Team A, $G_A \sim \text{Pois}(\lambda_A)$, and similarly for Team B.

We assume G_A and G_B are *independent* random variables, and for now also assume that the rates at which goals are scored by each team is constant, with no dependence on the properties of the opposing team.

Since the model tracks the number of goals scored by each team, if we use it to simulate a game, we can easily identify the game's winner and loser (or perhaps instead that the game was a draw, with both teams scoring the same number of goals).

Question One (10%)

Write a Python function to simulate the result of a single game according to the Poisson model above.

The two arguments to the function should be λ_A and λ_B (although feel free to call the arguments whatever you wish in the signature of the Python function you create).

The function should return a list with three elements: numbers of goals scored by Teams A and B, as well as 'A' (if Team A won the game), 'B' (if Team B won the game), or 'D' (if the game was a draw).

Assuming the parameter values are $\lambda_A = 1.5$ and $\lambda_B = 2$, use your function in a Monte Carlo simulation with 1,000 trials to estimate the proportion of games won by team A.

Fitting a Poisson model by maximum likelihood

Simulating a game with the above function requires the mean numbers of goals for both teams to be known.

However, in Part Two, the dependence of these parameters on a team's "strength" (captured via the team's FIFA World Ranking), as well as upon the properties of the opposing team, must be modelled based on historical data.

Fitting this type of model to data was covered in my lectures for Block A, and the practical class in week 4. We show you the mechanics of fitting such a model in Python by recovering the parameters from a small, synthetic data set (which you will first construct).

We consider four teams, each scoring a fixed mean number of goals (irrespective of the opposition):

- Team W (with mean number of goals per match, $\lambda_W = e^1 \sim 2.72$)
- Team X ($\lambda_X = e^{0.5} \sim 1.65$)
- Team Y ($\lambda_Y = e^{0.75} \sim 2.12$)
- Team Z ($\lambda_Z = e^{1.25} \sim 3.49$)

Question Two (5%)

Write Python code using your function from Question One to generate a data set D which contains the results of $N = 10$ simulated matches between pairs of teams taken randomly from $\{W, X, Y, Z\}$ (this should be done in a way that ensures the two teams in each match are distinct, and that each team is involved in at least one match of the N in total).

This synthetic data set will be needed later, so the results should be stored in a Pandas data frame with columns: `TeamA`, `TeamB`, `goalA` and `goalB`.

For the data set D , write code to calculate each team's mean goal scoring rate per match (i.e., the total number of goals scored divided by the number of matches that team was involved in).

Your notebook should include statements printing the data set and team's mean goal scoring rates.

Given a data set D with N games, in which the i^{th} game is between Team s_i and Team t_i , where s_i and t_i are taken from $\{A, B, C, D\}$, with score (u_i, v_i) , where u_i and v_i are the numbers of goals scored by Team s_i and Team t_i , respectively, then the likelihood of obtaining the data set D depends on the parameter vector, $(\lambda_W, \lambda_X, \lambda_Y, \lambda_Z)^T$ and is given by

$$L(\lambda_W, \lambda_X, \lambda_Y, \lambda_Z) = \prod_{i=1}^N \left(\frac{(\lambda_{s_i})^{u_i} \exp(-\lambda_{s_i})}{(u_i)!} \right) \left(\frac{(\lambda_{t_i})^{v_i} \exp(-\lambda_{t_i})}{(v_i)!} \right).$$

Each term in the product corresponds to the contribution to the likelihood from the i^{th} game. The first bracket corresponds to the contribution from the score of Team s_i , and the second to that from the score of Team t_i .

Question Three (5%)

Find the log likelihood function and show that its only stationary point occurs when the goal scoring rate for each team is equal to the team's mean goal scoring rate in the data set D .

(This is the maximum likelihood estimate of each parameter, but you are not asked to prove this)

In numerical work it would also be unusual to explicitly write code to produce the logarithms of the individual terms in the large round brackets above. More common would be to use a function from `scipy.stats.poisson` to find the (logarithm of the) probability mass function.

Question Four (10%)

Write a Python function which uses `logpmf()` from `scipy.stats.poisson` in a suitable `for` loop to calculate the log likelihood function given the data set D (from Question Two) and particular values of $\lambda_W, \lambda_X, \lambda_Y$ and λ_Z .

Write further Python code using the function `minimize()` from `scipy.optimize` to find the maximum likelihood estimates of the parameters $\lambda_W, \lambda_X, \lambda_Y$ and λ_Z for the data set D .

From such a relatively small sample, you would not necessarily expect the estimated rates to precisely match the rates used to create the data (i.e., it would be no surprise if your estimated λ_W was not *that* close to $e^1 \sim 2.72$). However, in checking your work, the answer to Question Three should be useful.

Your notebook should include a clear statement of your estimates of $\lambda_W, \lambda_X, \lambda_Y, \lambda_Z$, as well as the value of AICc (the AIC corrected for a small sample size) for the data set D for this model.

Fitting a Poisson model as a generalised linear model

One way of completing the project would be to continue to write explicit functions for likelihoods and to continue to fit models via numerical optimisation¹. However, such an approach involves unnecessary work.

The simpler and more efficient way is to note that the model as posed above – as well as all other models which will be required later – fits into the framework of generalised linear modelling. What we are doing here is in fact just a Poisson regression. Noting this allows model fitting to be done much more easily using functions in `statsmodels`.

To fit the model this way, the format of the data set must be altered. We need stacked format, in which each observation is on a separate row (currently your data frame consists of two observations per row, since each game contributes two numbers of goals, one for each team).

Create a new Pandas data frame with two columns – `Team` and `Goals` – but with 2 rows per match.

As an example, if the first two rows of your original data frame were

TeamA	TeamB	goalA	goalB
X	Z	1	0
Z	Y	2	1

then the first four rows of the updated data frame would be

Team	Goals
X	1
Z	0
Z	2
Y	1

The code below fits the Poisson model as a generalised linear model and prints a summary of its results (it assumes the updated data frame is called `myData`).

```
import statsmodels.formula.api as smf
import statsmodels.api as sm

myMod = smf.glm(formula = 'Goals ~ Team',
                 data = myData,
                 family = sm.families.Poisson()).fit()

print(myMod.summary())
```

The coefficients of the model will be reported as natural logarithms. The coefficient reported for “Intercept” will be the (logarithm of the) goal scoring rate for Team W; the coefficients for Teams X, Y and Z will also be reported as logarithms, as offsets taken relative to the baseline from the intercept.

¹ In practice for larger examples, you would face severe numerical problems in the optimisation. Instead, this type of calculation is done in practice using a method designed specifically for solving the score equations numerically, rather than by feeding the log likelihood function into a general-purpose optimiser. One such numerical method (“Fisher scoring”) is built-in to the library functions for generalised linear models.

Question Five (5%)

Write Python code to reshape the data set D created in Question Two to use stacked format as described above, and to fit the model using Poisson regression.

Your notebook should report estimates of $\lambda_W, \lambda_X, \lambda_Y, \lambda_Z$ and AICc for the data set D for this model, confirming that the results are identical to those obtained in Question Three.

Parameterising the mean number of goals per match in terms of team strength

A predictive model needs a mechanism to estimate mean numbers of goals per match based on past data. A parsimonious way of doing this would use a notion of the “quality” of each team as its input. Fortunately, the FIFA World Rankings provide precisely this.

For now, assume that, if Team k is ranked, r_k , then the natural logarithm of the mean number of goals per match depends linearly on the team’s rank, with

$$\log(\lambda_k) = \alpha + \beta r_k.$$

Since the ranking is in reverse order, with the team ranked #1 being the strongest, we would expect that $\beta < 0$.

Assuming values of r_k were available in advance of a tournament – for example via the FIFA World Rankings – such a model would be fully specified if the values of α and β could be fitted by considering past data.

Imagine that, instead of knowing the parameters controlling the mean number of goals per match for each of our set of teams, we instead knew the following rankings.

- Team W (with rank $r_W = 2$)
- Team X ($r_X = 4$)
- Team Y ($r_Y = 3$)
- Team Z ($r_Z = 1$)

Note I did not select the above rankings at random. Instead, they are precisely those required to recover the original set of mean numbers of goals per match (i.e., the values of $\lambda_W, \lambda_X, \lambda_Y$ and λ_Z) for a certain pair of values of α and β .

In particular, the data set D created in Question Three would be a perfectly good sample of the output of the updated model, in the case $\alpha = 1.5, \beta = -0.25$.

Add a column `Rank` to the data frame created for Question Five, so its first four rows would be

Team	Goals	Rank
X	1	4
Z	0	1
Z	2	1
Y	1	3

Hopefully you can see how the values of α and β will be recovered by fitting the Poisson generalised linear model `Goals~Rank`.

Question Six (10%)

Write Python code which updates the stacked data frame created in Question Five to include a new column `Rank`, and which uses Poisson regression to fit the model `Goals ~ Rank` to the data set `D`.

Your notebook should report the estimates of α and β , as well as the AICc of the fitted model.

You should not expect the fitted values of α and β to be precisely 1.5 and -0.25, since the data set might provide better support for other values, but they should be consistent with the fit obtained in Questions Four and Five.

Your notebook should therefore also include a scatter graph, with one point per team, showing rank on the x-axis and the logarithm of the goal scoring rate on the y-axis. Add a line corresponding to the fitted values of α and β to this graph. Explain how it shows the fitted values of α and β are consistent with the results from Questions Four and Five.

Which of the models fitted in Questions Five and Six is the most parsimonious?

Simulating a tournament

Although the statistical models in Part Two of the project will be more complex, we now have all the technical tools in place to do model fitting. But to use this type of model to predict the European Championships, we also need to be able to simulate an entire tournament.

A highly simplified² model of Euro 2024 imagines it is simply a pure knock out tournament, with all games played at random. This is made slightly awkward because 24 teams qualified for Euro 2024. In the first round of our simplified simulation, we therefore select 8 teams which are given a “bye” to the next round and qualify without “playing” a match. The remaining 16 teams are put into 8 pairs, and a single game between each pair is simulated.

In the knockout part of Euro 2024, drawn matches were decided by extra time (or by a penalty shootout if the two teams remained drawn after extra time). However, a simple way of ensuring a conclusive result of each game in our simulation would simply be to simulate it, but to decide the winner of any drawn games purely at random (i.e., according to a single $p = 0.5$ Bernoulli trial), and without attempting to simulate extra time.

At the end of the first round, the 16 teams remain in the tournament (the 8 teams given a bye, and the 8 winners of matches). The second round could then consist of these 16 teams being put into 8 pairs and playing each other. This would lead to 8 winners, who would go onto the next round. And so on.

² A more elaborate simulation would account for the group structure, i.e., the idea that before the tournament started, teams were divided into 6 groups, with each team playing each other team in each group, and with either two or three teams with the most points from each group progressing to a knockout phase involving 16 teams. There are then further constraints on the knockout phase, with possible pathways of which teams will play each other encoded in the “bracket”, a list of matches available before the tournament even started (e.g., it was preordained that whichever team won Group A would play against whichever team came second in Group C). It is certainly entirely possible to code such a simulation, and indeed this type of simulation would allow some interesting questions to be investigated, e.g., the effect of the random selection of group on an individual team’s performance (the “Group of Death” idea beloved of the media). However, it rapidly becomes quite tricky to keep track of all the fine details, and in terms of predicting a winner, the extra complexity does not *markedly* affect the results. We therefore attempt to make predictions with the simplest possible simulation model.

Continuing this procedure until only 2 teams are left allows that game to function as a final; whichever team wins it would emerge as the winner of the tournament.

The numbers of teams yet to be knocked out at the start of each round would be Round One = 24, Round Two = 16, Round Three = 8, Round Four = 4 and Round Five = 2. This type of World Cup simulation therefore involves 5 rounds.

Although it is far from perfect, this model does allow us to predict a winner of Euro 2024 based on a measure of each team's strength, and so is sufficient for us to continue with Part Two of the project.

Question Seven (20%)

The 24 teams involved in Euro 2024 are specified in `2024_teamID.csv`. Current FIFA rankings can be read from the `Rank` column of the file `2024_rankings.csv`.

The focus of this task is therefore not really on the model of a football game, but instead on how the results of that model are can be used. Nevertheless, a model of the number of goals scored by each team in each game is required.

For now, as a placeholder model, please assume the goal scoring rate of team k follows $\log(\lambda_k) = \alpha + \beta r_k$, where r_k is the team's ranking, and the parameters are $\alpha = 1$ and $\beta = -0.05$.

Write code to simulate a Euro 2024 as a knockout tournament between the 24 teams, as described above, using a Poisson model with λ_k as the goal scoring rate of the k^{th} team.

Your notebook should print details of a single example tournament, printing sufficient information for us to check whether the code is functioning correctly.

By running the code 1,000 times, estimate the probability of each team winning the entire tournament (for the placeholder model described above).

Note you only need to produce a point estimate of the probability of winning for each team, i.e., you do not need to provide any estimate of the variability.

PART TWO. WAS EURO 2024 PREDICTABLE? (35% OF THE MARKS ARE GIVEN TO PART TWO)

After working through Part One, you should be ready to attempt the open-ended part of the project. You will simulate Euro 2024 using a model – which you will fit to data from European Championships from 1996-2021 – based on FIFA World Rankings and compare your prediction with what happened.

You can assume the numbers of goals scored per match by each of the pair of participating teams are independent random variables. However, your model should assume the goal scoring rates depend not only on the team's strength, but also on the strength of the team's opposition, as well as whether either team is playing at home. In particular, at least to start with, in a game between Team A and Team B, we assume the logarithm of the mean number of goals scored by Team A is

$$\log(\lambda_A) = \alpha + \beta r_A + \gamma r_B + v_A,$$

and the logarithm of the mean number of goals scored by Team B is

$$\log(\lambda_B) = \alpha + \beta r_B + \gamma r_A + v_B.$$

The covariates r_A and r_B are the FIFA World Rankings of Teams A and B at the start of the tournament (note these rankings change over time, meaning the ranking of a particular team directly before one European Championship will almost always be different to its ranking directly before any other).

The model parameter β is a measure of how the ranking of the focal team affects the number of goals it scores. Since stronger teams tend to score more goals, but lower rankings correspond to better teams, a reasonable expectation is that $\beta < 0$. The model parameter γ is a measure of how the ranking of the opposing team affects the number of goals scored by the focal team. Since stronger teams will be expected to be harder to score goals against, a reasonable expectation is that $\gamma > 0$. The remaining parameter α is simply an intercept to allow the model to reflect the average rate of goal scoring across all games in the data it is fitted to.

Parameters v_A and v_B allow the model to account for the home advantage (or lack of it). These parameters are restricted to take one of three values (for team x):

- $v_x = v_h$ if team x is playing at home (and the opposing team is not)
- $v_x = v_n$ if neither team involved in a match is playing at home
- $v_x = 0$ if the opposing team is playing at home

The parameter v_h is therefore the additional expected number of goals scored per game by the home team, and v_n is the additional number of goals scored per game (by both teams) in games in which both teams are neutral, i.e., neither team is playing at home. These quantities are relative to the additional expected number of goals scored by any team unfortunate enough to be playing against a team playing at home, which is fixed at $v_x = 0$.

The initial model therefore contains five parameters: $\alpha, \beta, \gamma, v_h$ and v_n . We leave it up to you to decide whether you want to do model selection to come to inferences about whether you prefer a simplification of this model, or an extension, or even a different model entirely.

Of course, fitting your model requires data.

The file `gameData.csv` contains the results of matches in past European Championships (1996-2021). Each match corresponds to a single row of the csv file; just like in Part One, the data will need to be put in stacked format, and each match will contribute two rows to the data set used to fit the model. The columns `goalA` and `goalB` show the score for TeamA and TeamB at the end of the first 90 minutes of each match. The column `Home` contains A or B if either TeamA or TeamB was the host nation in the

relevant year (there is normally just one host nation, but in 2000 and 2008 the competition was jointly hosted by two countries, and in 2021 matches were played in 13 different countries).

Although drawn games in the knockout stage will also have required extra time and/or penalties, such data will not be used in our model fitting, and so are not included `goalA` and `goalB` in the data set.

In fitting your model, you might perhaps find it helpful to reformat your data set to include two indicator variables, `Home` (which contains 1 if a nation is at home and 0 if not) and `Neutral` (which contains 1 if neither team is at home, and 0 if either team in the match was at home). For example, two matches from the 1996 European Championship (which was held in England) were

- England 1 – Switzerland 1
- Netherlands 0 – Scotland 0

Since we require a stacked format, the two matches above could be coded as

Team	Goals	Home	Neutral
England	1	1	0
Switzerland	1	0	0
Netherlands	0	0	1
Scotland	0	0	1

(In fitting the model, you will probably require the data frame to contain other columns, too).

Part Two. Was Euro 2024 Predictable?

The file `rankData.csv` contains FIFA world rankings taken just before the start of the tournament for each team in each European Championship since 1996.

Use these data and `gameData.csv`, to fit the five-parameter model described above to European Championships from 1996-2021 (extra time & penalty shoot outs are ignored in model fitting).

Use your fitted model to simulate Euro 2024. To do this, you will need the rankings just before the tournament of the 24 teams involved. Recall these can be found in the file `2024_rankings.csv`.

Your write up should describe (at least):

- * A justification – including any relevant visualisations – of why the model above is sensible.
- * Your model fitting and justification for any model selection you have done, and any other models you have tried to fit.
- * Results of your simulations of Euro 2024, ideally visualised in an attractive way.
- * A comparison of your prediction with the actual result of Euro 2024, again ideally visualised attractively (the actual results are provided in the file `2024_results.csv`).
- * Any limitations of the modelling exercise, or improvements you would suggest.

The write up for Part Two should be submitted as a pdf document and should be no longer than 5 pages of A4 (in a 11-point font), including all figures.

Remember no code should be submitted; only your written work for this part will be considered.