WHEN OLD MEETS NEW: EVALUATING NUMERICAL AND MACHINE LEARNING BASED ECLIPSE PREDICTION METHODS

MARA SFERDIAN[1], MARC FRINCU[2]

[1]*Faculty of Mathematics and Computer Science, West University of Timisoara, Bld. Vasile Parvan No 4, Timisoara, Romania Email: mara.sferdian99@e-uvt.ro*
[2]*School of Science and Technology, Nottingham Trent University, Clifton Campus, Nottingham, UK Email: marc.frincu@ntu.ac.uk (corresponding author)*

Abstract. The Babylonians had some of the most advanced arithmetic models for the Lunar and planetary theory in ancient times. This allowed them to discover the Saros eclipse cycle. In this paper we investigate the accuracy of several eclipse prediction methods tested on worldwide occurring eclipses between 2020–2100 CE, ignoring the sophisticated modern models based on accurate ephemerides, in an attempt to understand how simple models would have worked in ancient times. First we propose two numerical methods relying on knowledge available in Babylonian times – lunar phases, lunar nodes, and the angular separation between the Sun and the Moon. Second, we assess the performance of four Machine Learning (ML) models modeling human inference by relying on the same data. The accuracy of the numerical methods is above 80% while the ML models achieve up to 98% accuracy. The algorithms perform better in case of lunar eclipses. While not 100% perfect, these methods are simplistic in terms of required information and enable us to get an insight into how efficient might have been ancient methods relying on visual observations.

Key words: Astronomy – Solar System – Solar-terrestrial relationships – History of Astronomy – Eclipse prediction – Machine Learning – Astronomical algorithms.

1. INTRODUCTION

Eclipses have fascinated humanity since the dawn of time. Written records exist from China, where in ∼2159 BCE (Brown, 1931), two astronomers were executed at the orders of the emperor for failing to announce the moment of the first contact. While this was probably an act of punishment not for failing to predict the eclipse but rather of their failure in their duty to watch the sky, it does give us an idea of the importance of such events in past societies. Skywatchers across the globe looked at the sky depicting their accounts in either rock art (Vaquero and Malville, 2014) or written documents (Stephenson, 2006).

Among ancient cultures, the Babylonians developed by the mid 1st millennium BCE

Table 1

Eclipses visible over Europe. Saros solar cycle 145 starting on August 11, 1999 and ending on August 21, 2017. Saros lunar cycle 124 starting on January 21, 2000 and ending on January 31, 2018. Strikeout lunar eclipses are not visible from Romania.

Date	Int.	Date	Int.	Date	Int.	Date	Int.
Aug 11, 1999 (S)	**5**	Jan 21, 2000 (L)	**6**	~~Jul 16, 2000 (L)~~	**6**	Jan 9, 2001 (L)	**6**
~~Jul 5, 2001 (L)~~	**5**	~~Dec 30, 2001 (L)~~	**6**	Jun 24, 2002 (L)	**5**	Nov 20, 2002 (L)	**6**
May 16, 2003 (L) May 31, 2003 (S)	**6**	Nov 9, 2003 (L)	**6**	May 4, 2004 (L)	**5**	Oct 28, 2004 (L)	**12**
Oct 3, 2005 (S)	**5**	Mar 16, 2006 (L) Mar 29, 2006 (S)	**6**	Sep 7, 2006 (L)	**6**	Mar 3, 2007 (L)	**11**
Feb 21, 2008 (L)	**6**	Aug 1, 2008 (S) Aug 16, 2008 (L)	**6**	Feb 9, 2009 (L)	**6**	Aug 6, 2009 (L)	**4/5**
Dec 31, 2009 (L)Jan 15, 2010 (S)	**11/12**	Dec 21, 2010 (L)Jan 4, 2011 (S)	**5**	Jun 15, 2011 (L)	**6**	Dec 10, 2011 (L)	**11**
Nov 28, 2012 (L)	**5**	Apr 25, 2013 (L)	**6**	Oct 18, 2013 (L)	**6**	~~Apr 15, 2014 (L)~~	**11**
Mar 20, 2015 (S)	**6**	Sep 28, 2015 (L)	**12**	Sep 16, 2016 (L)	**5**	Feb 11, 2017 (L)	**6**
Aug 7, 2017 (L)	**6**	Jan 31, 2018 (L)	–	–	–	–	—

a working theory allowing them to predict the ephemeris of solar system bodies with increased accuracy. Over time they developed methods for predicting lunar and solar eclipses and there is evidence that they knew the 18-year "Saros"[*] eclipse cycle used for the prediction of eclipses and of lunar phases near the syzygies[†] (Neugebauer, 1975a). The eclipse cycle is probably part of the earliest phases of Babylonian mathematical astronomy. In fact, based on the amount of data stored in discovered clay tablets, the lunar theory was more advanced than the planetary theory and more source material for it has been discovered than for the planets combined.

Similarly to the planetary theory, Babylonians had for the lunar theory two systems, called *System A* (described by a step function and for the vernal equinox located at ♈10°) and *System B* (described by a linear zigzag function and for the vernal equinox located at ♈8°). While a clear chronology between the two has not yet been

[*]The terminology was first used by Halley in 1691 (Neugebauer, 1975a) who borrowed it from an 11th century Byzantine lexicon. There is no evidence that Babylonians ever used this name.

[†]Alignment of 3 or more bodies. For the Sun-Earth-Moon system it occurs every Full/New Moon.

established (they do seem to be favored in different cities, *i.e.* , Babylon for System A and Uruk for System B) System A is used in the majority of the "procedure texts" and later ephemeris based on it is the direct continuation of older ones.

While Babylonians may have noticed the Saros cycle by observing lunar eclipses it is not without reason to assume that it was well-known in antiquity that eclipses occurs roughly every 6 months and occasionally every 11 months (Neugebauer, 1975b) – a recurrence known at least since the time of Hiparchus in the 2nd century BCE. While this may have been the general rule, not every 6-month or 11-month interval generates an eclipse visible from a given geographical position (see Table 1) and hence determining its periodicity empirically requires observers in different geographical locations. Also, some penumbral lunar eclipses are invisible or at least hard to observe with the naked eye due to the indiscernible change in lunar brightness.

Today we have accurate numerical methods for predicting eclipses that span 5,000 years (Espenak) but the quest for identifying the time and place of ancient eclipses (Stephenson, 2006) is far from over due to the well-known ΔT problem which links the Terrestrial Dynamic Time to the Universal Time (Morrison and Stephenson, 2004; Steele, 2005). The oldest identified eclipse which enables us to refine the past occurrence of an eclipse goes as far as the Late Bronze Age (October 30, 1207 BCE) in the time of Ramesses the Great (Humphreys and Waddington, 2017).

Babylonians compiled long lists of eclipses dating as far as the 5th century BCE (or even the 8th century according to Ptolemy) but were unaware that both solar and lunar eclipses were produced by the same phenomenon. At the same time we do not know how many of these were accurately predicted by their arithmetical methods.

In this article we investigate two prediction methods accessible to Babylonians through their knowledge. We also test several modern Machine Learning based methods which behave similarly to the human brain, by learning and inferring from past experience. We apply the algorithms on all worldwide occurring eclipses between 2020–2100. The results shed light on the potential and level of Babylonian astronomy.

The rest of the paper is structured as follows: Section 2 presents some of the existing prediction methods and introduces the mathematical requirements for an eclipse to take place. Section 3 describes the proposed eclipse prediction methods. Section 4 presents the performance metrics and Sect. 5 outlines the results. Finally, Sect. 6 summarizes the main results.

2. RELATED WORK

Arguably, the definite proof that ancient cultures had the ability to predict eclipses based on the Saros cycle is represented by the Antikythera mechanism which was analyzed thoroughly by Freeth (2014).

In modern times, the most comprehensive list of solar eclipses compiled to date is arguably the one from Espenak and Meeus (2006) spanning from 1999 BCE to 3000 CE. It uses the VSOP87 algorithm (Bretagnon and Francou, 1988) for computing the Sun coordinates and ELP-2000/82 (Chapront-Touze and Chapront, 1983) for the Moon. For the latter all periodic terms with coefficients smaller than $0.0005''$ in longitude and latitude, and smaller than 1 m in distance have been ignored. The value for ΔT is taken from various sources depending on the time span. Meeus (1998) has provided a series of simplified formulae for computing eclipses that provide reasonable results when high accuracy is not needed. For instance, all solar eclipses between 2020–2100 are correctly identified while two lunar eclipses are omitted, namely the one on July 18, 2027 ($penumbral\ magnitude = 0.028$) and June 6, 2096 ($penumbral\ magnitude = 0.03$ (Wikipedia contributors, 2021)).

The closest study to our own was recently published by Pingalkar (2020). The author uses a Gradient Boosting based algorithm with an accuracy of 96%. The results are however not thoroughly discussed and there are a lot of input parameters, with the algorithm requiring fine tuning which is not addressed in detail.

3. ECLIPSE PREDICTION METHODS

For an eclipse to occur a syzygy between the Sun-Earth-Moon has to take place. This means that the Moon and Sun must be near the lunar nodes and the Moon has to be either Full or New. For this to happen $P = m \cdot synodic\ month \approx n \cdot draconic\ months$. Also, for an eclipse to be similar with a previous one the Moon has to be at the same distance, *i.e.* , the period covered by a number of anomalistic months must be equal to P as well. The Saros cycle meets these conditions with $223 \cdot synodic\ month \approx 242 \cdot draconic\ month \approx 239 \cdot anomalistic\ month \approx 18\ years$.

To determine this relation the duration of the synodic, anomalistic and draconic months has to be known. Table 2 shows the values as known in ancient Babylon. The values for the draconic month for System A and System B have been taken from (Goldstein, 2002) which derives them based on Babylonian knowledge.[‡]

[‡]Goldstein uses a value for the Saros period of 6585.25 days. If we use the rounded value provided by Neugebauer we get a value of 27.210743802 for the draconic month or 27.212123966 if we use a Saros period of 6585.334 days that is inferred based on the length of the synodic month in System A.

Table 2

Mean values for various month types as known in ancient Babylon (Neugebauer, 1975a; Goldstein, 2002).

Type	Modern value	System A	System B	Older systems
synodic	29.530588853	29.53064437	**29.53059259**	29.53055556
anomalistic	27.55454988	**27.55458333**	27.55555556	n/a
draconic	27.212220817	**27.21212037** (?)	27,21212037 (?)	27.33333333

As eclipses can theoretically occur at each node we need to approximate the number of half draconic months per synodic months, *i.e.* , $\frac{2 \cdot synodic\ month}{draconic\ month} = 2.170391682$ by using common fractions since eclipses can only occur at integer (same eclipse type) or half integer (different eclipse types) multiples of the synodic month.

In case of the Saros cycle we obtain based on the previous relation a ratio of 484 half draconic months to 223 synodic months which gives a value of 2.170403587 that differs by only $1.1905 \cdot 10^{-5}$ from our target value.

The half year cycle is approximated by 13 half draconic months which fit 6 synodic months providing a ratio equal to 2.166666666. This is possible since the offset between a draconic month and a synodic month is of 13,9102108 days after 6 synodic months, which is short of half a draconic month by just 0.304098 days or roughly 7 hours and 18 minutes. In this interval the argument of latitude increases by $\Delta\overline{\omega} = 184.023^{\circ}$ which is within the $180 \pm 20^{\circ}$ limits for the separation of two syzygyes for a solar or lunar eclipse to take place (Neugebauer, 1975b).

It is clear from the above that the Babylonians had both the skills and the knowledge to observe eclipses and to possibly make predictions. The question that arises is "how accurate were these predictions?". In the following sections we address this issue by comparing the accuracy of predictions made using simple reasoning by taking into account various lengths for the synodic and draconic months (since we do not search for identical eclipses) as indicated by Babylonian sources. In addition, we will test the efficiency of some modern machine learning techniques in an informal game between ancient humans and computers.

3.1. NUMERICAL METHODS

3.1.1. Predicting eclipses using lunar phases and nodes

The first algorithm we propose in this paper attempts to predict dates of future eclipses requiring previous knowledge of just three important dates: one ascending and one descending lunar node dates, and the date of one New (for solar eclipses) or a Full (for lunar eclipses) Moon. All can be determined from observations.

This algorithm considers that an eclipse occurs if two conditions are met: firstly, the Moon and Sun need to either have the same ecliptic longitude (New Moon) or for them to differ by $180°$ (Full Moon). Secondly, the Moon needs be close to its nodes. In this algorithm we successively add the average values of the synodic and draconic months to predict the next lunar phase of interest and nodes. Obviously, the real variations in synodic and draconic month lengths will accumulate errors, but these will be ignored for now.

For each predicted lunar phase, we determine if the Moon is in one of the nodes, more explicitly, if it is also near to a full cycle equal to a draconic month. To achieve this we compute the difference in days between the considered date and the predicted nodes, divide this number by the length of a draconic month, and select the ones with the remainder closest to 0 (a full draconic cycle).

Due to the error generated by ignoring the variation in the actual synodic and draconic month lengths and to the fact that the alignment does not need to be perfect, we allow an error margin of ± 1.3 days; therefore, if the remainder previously calculated is smaller than this error margin, we consider an eclipse to take place on that date. The method is depicted in Algorithm 3.1.1. Given the nature of the calculations we always start with a Full/New Moon closest to a node (eclipse).

Algorithm 3.1.1 Predicting eclipses using lunar phases and nodes

```
function GETFUTUREECLIPSES(lunarPhase, ascNodeDate, descNodeDate, endDate)
    eclipses ← []
    while lunarPhase < endDate do
        ascNodeDif ← abs(lunarPhase-ascNodeDate)  mod draconicMonth
        descNodeDif ← abs(lunarPhase-descNodeDate)  mod draconicMonth
        ascNodeDif ← min(ascNodeDif, draconicMonth-ascNodeDif)
        descNodeDif ← min(descNodeDif, draconicMonth-descNodeDif)
        if min(ascNodeDif, descNodeDif) < ERRMARGIN then
            eclipses.add(lunarPhase)
        end if
        ascNode ← ascNode + draconicMonth
        descNode ← descNode + draconicMonth
        lunarPhase ← lunarPhase + synodicMonth
    end while
    return eclipses
end function

solarEclipses ← GETFUTUREECLIPSES(newMoon, ascNodeDate, descNodeDate, endDate)
lunarEclipses ← GETFUTUREECLIPSES(fullMoon, ascNodeDate, descNodeDate, endDate)
```

3.1.2. Predicting eclipses using apparent separation between the Sun and Moon

The second algorithm proposed in this paper considers eclipses as visual phenomena. It expresses an eclipse as a geometric relation between the apparent positions of the Sun and Moon, as seen by an observer somewhere on Earth. The main idea is to test whether or not the relation is valid at any point of the day. This method requires more information than the previous one. It consists of checking for an overlap of the apparent shapes of the Sun and Moon at certain moments during a day and certain locations on Earth. To detect an overlap, knowing the apparent radii of the Sun and Moon and the separation between them is required.

To obtain perfect accuracy (see Section 4 for a discussion on the evaluation metrics), we have to check every moment and every position on Earth, which is unfeasible as there are infinitely many points in space and time. Therefore, we must select a finite number of moments and locations for which to perform the check.

We will assume a separation small enough but slightly larger than the sum of radii indicates a solar eclipse. For lunar eclipses, we aim to find a value close to $180°$ (indicating that the Sun and Moon are in opposite positions). The difference between $180°$ and the angular separation is the distance between the centers of the Moon and Earth's shadow. Similarly to solar eclipses, where the separation does not need to be exactly 0, in case of lunar eclipses the separation does not need to be exactly $180°$ for the Moon to be overshadowed by Earth.

The downside of this approach is that it can identify "close calls" as eclipses, thus false positives. It is trivial to notice that as we increase the number of moments and locations the accuracy will converge to a theoretical value of 1 but will never reach it. In practice, as there is a finite number of eclipses during a given time, a theoretical perfect accuracy is not required for a 100% hit rate on the finite test data.

The algorithm attempts to find the best value detected in one day from certain points around the globe using the following formula:

$$best = \begin{cases} max(moon_{rad} + sun_{rad} - separation), & \text{in case of solar eclipses} \\ min(abs(separation \pm 180)), & \text{in case of lunar eclipses} \end{cases} \quad (1)$$

In case we want to predict solar eclipses, $best$ should be > 0 and in case of lunar eclipses, $best$ should be < 0.5, so that the geometric condition is true.

Algorithm 3.1.2 shows how this method can be used to predict solar eclipses. It should be noted that we do not calculate the angular distance between the Sun and Moon or their apparent radii ourselves but instead rely on the PyEphem Python library – based on the VSOP87 theory and Jean Meeus' algorithms.

3.1.3. Discussion on the application of these methods in Babylonian times

Both methods that we previously introduced can be solved by using Babylonian knowledge alone. The first method requires knowing the length of the synodic and

Algorithm 3.1.2 Predicting eclipses using apparent angular distance between the Sun and Moon

```
function IsInitialConditionValid(date)
    coord ← 0:0                                         ▷ (lon, lat)
    time ← 12:00
    separation ← GetSeparation(date : time, coord)
    return separation < 30
end function

function CheckIfCoordsValidateEq(date)
    for time in timeIntervals do
        for coord in coordinates do
            res ← moonRad + sunRad − GetSeparation(date : time, coord)
            if res > 0 then
                return true
            end if
        end for
    end for
    return false
end function

function IsDayEclipse(date)
    if IsInitialConditionValid(date) then
        isEclipse ← CheckIfCoordsValidateEq(date)
        return isEclipse
    else
        returnfalse
    end if
end function
```

draconic month, which we know from ancient text that they knew to a fair amount of precision (cf. Table 2). In Sect. 5 we analyze the accuracy of the method by using various lengths for the two types of moons.

While in the second method we mentioned – using modern terminologies – a geometrical method, the angular separation between the Moon and the Sun was already available to Babylonians through the Rising Time Schemes (Steele, 2017) where a method for determining the culmination times for a select groups of *Ziqpu Stars* with respect to sunrise or sunset is given. These provide an equivalent to modern day oblique ascensions. The position of the Moon is described in relation to zodiacal constellations and its speed is computed based on either System A or System B. We know that Hypsicles (2nd century BCE) derived a non-trigonometrical solution for the oblique ascensions by using Babylonian arithmetic astronomy (Montelle, 2016).

3.2. MACHINE LEARNING METHODS

Machine Learning (ML) has become extremely popular in recent years, being applied in numerous domains, such as fake news detection and medical diagnosis. It is both a simple and powerful tool simulating human learning through computer algorithms.

Supervised learning is one of the best-known areas of Machine Learning; it is specialized in predicting output values for certain inputs, when we possess knowledge of some inputs' outcomes. It takes human effort to compile the dataset, but it optimizes the strenuous task of making predictions on never-before-seen data (Müller and Guido, 2016). Supervised ML problems can be broken down into two main categories: regression and classification.

Classification problems consist of mapping each input variable to an output, known as class label; these output variables are discrete and the classes are disjoint. Regression aims to predict a continuous number, which represents an amount. To determine if either a classification or regression predictive model is best fit for a task, we must first look at the structure of the output. If there is continuity between these output variables, we are dealing with a regressive model; otherwise, a classification model maps our problem best. It should be noted that a classification problem can be modeled using regression; for example, the regressive function can split the output data in two or more categories, based on a selected threshold. Such a case is Logistic Regression, which, despite its name, is a classification algorithm, unlike Linear Regression. (Müller and Guido, 2016).

Our model should predict if a certain date is likely to be an eclipse or not, therefore it fits the binary classification predictive model. To achieve this, we can use either simple classification algorithms (*e.g.*, *k*-Nearest Neighbors or Decision Tree) or a Logistic Regression classifier which offers us the confidence that our input variable belongs to one class or the other. We will describe a few tested methods and while we argue for the case of binary classification, we will assess the performance of regressive models as well.

3.2.1. Tested Classification Based Predictive Models

***k*-NN** is a simple classifier that selects the k data points closest to the current input; if *k*=1, it assumes that the input belongs to the same class as the most similar data point while for $k > 1$ the majority decides.

The **Decision Tree classifier** algorithm computes a series of sequential decisions designed to reach a desired result. The danger this model faces is the overfitting of the training data (if the decision tree is too closely fitted to the training data, it may not make accurate predictions on the test data, because the model is not generalized enough). One potential solution to the problem of tree overfitting is an ensemble model called a **Random Forest** which is a collection of mildly different decision trees (Müller and Guido, 2016).

The statistic model of **Logistic Regression** uses the homonymous function to model a binary dependent variable. The definition set of the logistic function is (0,1); this

Table 3

Solar eclipses training sample.

Date	Separation	Is Eclipse
2019-06-30	23.450(6)	False
2019-07-01	10.399(8)	False
2019-07-02	0.0642(7)	True
2019-07-03	1.72102(7)	False

value represents the probability that a sample belongs to a desired category.

For our dataset we will once again use the aforementioned Python PyEphem library. We aim to skip arithmetically checking if, knowing the separation between the Sun and Moon at a certain date and from an observation point, an eclipse is likely to occur (cf. Sect. 3.1.2 for details and implementation). Therefore, our dataset is composed of three columns: the date, the best separation (angular distance between the Sun and Moon) during that day (depending on the type of eclipse), and whether or not an eclipse happened that day.

We have chosen to search for the best separation each (sharp) hour of a day in 12 points around the globe, more specifically at the points determined by each pair of coordinates (longitudes 0°, 90°, 180°, 270° and latitudes 0°, 45° and -45°). In the case of a solar eclipse, the separation should be as close to 0 as possible, while for lunar eclipses, its absolute value should near 180. Table 3 showcases the structure of the dataset for solar eclipses.

Section 5 outlines the performance of using classification models.

3.2.2. Regression Models

While we argue for classification models we have also considered – for comparison purposes – in our initial analysis regression models such as **Linear Regression**, **ARIMA**, and **Random Forest/Tree Regression**. For our problem we have considered a series where each value represents the length of the time interval (in days) between two consecutive eclipses, hence the objective was to predict the interval to the next eclipse (either lunar or solar). A similar method based on time intervals would have been available in Babylonian times through the large amount of eclipse tables although no direct evidence of its existence has been found yet. Figure 1 depicts the results when varying the number of future predicted eclipses. The plot shows the best MAPE (Mean Absolute Percentage Error) value out of all methods. It is seen that the error increases drastically when more than one eclipse is predicted. The best results are obtained when predicting the next worldwide lunar eclipse or next solar eclipse over Europe. In 55% of cases ARIMA was the best prediction method followed by

Linear Regression (25%) and Random Forests (20%). On average, the MAPE values were: 34.18% (solar eclipses over Europe), 34.85% (worldwide lunar eclipses), 42.14% (worldwide solar eclipses), and 46.29% (lunar eclipses over Europe).

The large average MAPE values (*i.e.* , 34–46%) indicate that the prediction date is on average off by more than a month for the tested regressive models. The only time they fall short of a month is when predicting only the next worldwide lunar eclipse date. In this case an average interval of ≈153 days between worldwide lunar eclipses provides a MAPE of ≈14% which translates to a prediction date within $\approx \pm 21$ days from the actual eclipse day. This interval increases to $\approx \pm 42$ days for the next worldwide solar eclipse, $\approx \pm 80$ days for the next lunar eclipse visible from Europe, and $\approx \pm 87$ days for the next solar eclipse visible from Europe. This matches our expectations, because the problem does not fit a regressive model.

4. EVALUATION METRICS

In order to evaluate the efficiency of the proposed eclipse prediction techniques, it is necessary to first establish the best-fitted performance measure.

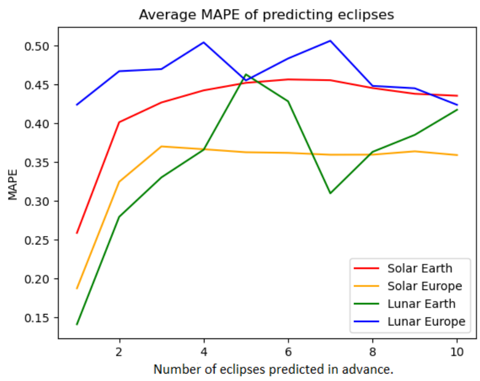

Fig. 1 – Accuracy of regression based prediction models.

Let us define: T as the number of total predictions, both positive and negative; P represents the real positive predictions, while N the real negative ones ($T = P + N$); F_N as the number of false negatives (dates of missed eclipses); F_P as the number of false positives (positive predicted dates in which an eclipse does not occur); T_P as the number of true positives ($T_P = P - F_N$), T_N as the number of true negatives ($T_N = N - F_P$) and A as the standard performance metric of accuracy ($\frac{T_P + T_N}{T}$).

Before proceeding it is important to explain why another more widely used metric (*e.g.*, accuracy, precision, recall) is not well-fitted enough to our problem and why all of them lose sight of significant drawbacks in our algorithms.

Since eclipses occur rarely, the standard *accuracy* will generate a large number ($A > 99\%$) for a naive solution that assumes all days are negatives (we define this as A_0).

Precision is defined as the ratio of accurately predicted positive observations (T_P) to the total of positive predictions ($T_P + F_P$). If our algorithm correctly predicts one eclipse and misses all others the precision score will be 1, a situation far from ideal.

The problem with using *recall* (the ratio of correctly predicted positives, T_P, to the actual number of positive observations, P) is quite similar; if our algorithm considers each tested date an eclipse, the recall will be 1, as we do not account for a large number of false positives.

For this reason, we measure the performance of an algorithm using a coefficient representing the increase in accuracy compared to A_0. This **accuracy coefficient**, which we define as C_A, will be 0 for an accuracy equal to A_0 and 1 for perfect accuracy.

Using simple arithmetic operations, we arrive at the following formulae:

$A = \frac{T_P + T_N}{T} = \frac{T - F_N - F_P}{T}$ and $A_0 = \frac{N}{T}$.

The variable of interest, $C_A \in [0, 1]$, where:

$$C_A = \begin{cases} 0, & \text{if } A = A_0 \\ 1, & \text{if } A = 1 \end{cases}$$

Therefore, it is easy to observe that it can be expressed as: $C_A = \frac{A - A_0}{\frac{P}{T}}$. By substituting the terms of the equation, we obtain:

$$C_A = \frac{T_P - F_P}{P} \tag{2}$$

Another relevant performance metric is the **F1 score**, more specifically the harmonic mean of precision and recall. The F1 score accounts for both false positives and negatives and is generally used when the data is imbalanced (in our case, there are much fewer days that contain an eclipse than days that do not).

Another metric that may be relevant for our task is **Cohen's Kappa coefficient**, widely used when dealing with imbalanced datasets, but more complex and therefore harder to interpret. Moreover, analyzing its behaviour on diverse examples of confusion matrices reveals an increased tolerance to false positives. For instance, an algorithm that predicts three times as many eclipses as their real number (the false positives are double the number of true positives) achieves a score of ≈ 0.5, which intuitively seems like a moderate score for what is otherwise a weak and clearly imbalanced algorithm. By using our metric, $C_A = 0$, as we would expect.

Therefore, to maximize the performance of our algorithms, we will use both the accuracy coefficient described by Relation 2 and the F1 score.

5. RESULTS

5.1. NUMERICAL METHODS

We tested the performance of the algorithm described in Sec. 3.1.1 by setting as input the first eclipse of 2020 as the lunar phase along its adjacent lunar nodes.

As already mentioned in the description of the algorithm, to ensure a high F1 score the error margin (in days) needs to be carefully selected. This requires searching for the value that balances the number of false positives with false negatives. For instance, if the number of false positives is greater than the number of false negatives, we need to have a smaller error margin (and vice-versa).

Table 5 showcases the parameters and error margins used to obtain the results from Table 4. The performance score can vary slightly depending on the three selected parameters we pass and we consider this method to be the most unstable of those presented in this paper. However, the results are surprisingly good for such a naive algorithm, especially for lunar eclipses (0.8 performance score and 0.9 F1 score).

The experiments considered the currently known values for the average draconic and synodic month length To determine the performance of this algorithm on lunar lengths known to Babylonians we also evaluated the algorithm using the values in Table 2). For System A and System B, the performance score is not greatly affected: ≈ 0.59 (39 F_Ps and 37 F_Ns) for solar eclipses and ≈ 0.74 (24 F_Ps and 26 F_Ns) for lunar eclipses. However, the values known in the older system, especially the average draconic month ($\approx 27.(3)$) results in a performance score < 0, meaning it performs worse than a solution that considers that no eclipses occur at all.

Table 4

Performance results of Algorithm 3.1.1 (using lunar phases and nodes).

Type	T_P	F_N	F_P	T_N	Performance	F1
Lunar	167 (0.57 %)	18 (0.06%)	19 (0.06%)	29,036 (99.30%)	**0.800**	**0.900**
Solar	149 (0.51 %)	33 (0.11%)	34 (0.12%)	28,897 (99.26%)	0.632	0.816

Table 5

Parameters used to achieve results presented in Table 4.

Type of eclipse	Full/New Moon	Ascending Node	Descending Node	Error Margin
Solar	2020-06-21 06:41	2020-06-21 04:24	2020-06-06 18:10	1.25
Lunar	2020-01-10 19:11	2020-01-09 23:29	2020-01-22 20:31	1.13

The second algorithm (cf Sect.3.1.2) – using the angular separation between the Sun and Moon – manages to predict all eclipses in the considered range accurately. However, the error margin also needs to be carefully selected.

If for solar eclipses we choose the best angular separation value detected in one day in various points around the globe as the maximum positive value, for lunar eclipses we choose the minimum value smaller than an error margin of ≈ 0.5 (cf. Sec.3.1.2). We search for eclipses three times per hour (every 20 minutes) for each point P, where $P \in S_{longitude} \times S_{latitude} \cup \{(0°, -90°), (0°, 90°)\}$. We have chosen $S_{longitude} = \{-170°, -160°, ..., 170°, 180°\}$ and $S_{latitude} = \{-80°, -70°, ..., 70°, 80°\}$.

By using these values, we miss one lunar and one solar eclipse, namely the partial solar eclipse of October 2098 – first of the Saros 164 series – and the penumbral lunar eclipse of July 2027 – last of the Saros 110 series. Initial and final eclipses of a Saros series occur at the limit of the eclipse occurring region around the nodes which makes them weak in magnitude. We found that they do not even feature in some future eclipses lists; the solar eclipse has a magnitude of 0.0056 and the lunar one has a penumbral magnitude of 0.028 (also missed by the Meeus algorithm).

To predict these two eclipses, we have to change the error margins slightly. For solar eclipses, we will accept a separation value > -0.003 (instead of 0), and for lunar eclipses a value < 0.56 (instead of the previously used 0.5). It is worth mentioning that this adjustment does not cause the algorithm to detect any false positives in the given time interval.

Even if this algorithm achieves a score of 1 (perfect accuracy) on eclipses between 2020 and 2100, it is not guaranteed that it will accurately predict every eclipse in any given time interval, because it is infeasible to check at every point on Earth, every second (cf. Sect. 3.1.2).

The main drawback of these two prediction methods is its very long runtime. Initially, both algorithms (for lunar and solar eclipses) completed running in about 8 hours. The PyEphem function which we mainly used in our calculations, that computes the angular separation between two bodies at a certain time and place, performs a single calculation in about 0.022 *ms* on an Intel i5-8250U CPU. Given the large number of calculations we perform (29,565 days, 72 times per day in 614 points around the globe), the algorithm runs in $\approx$ 29 million ms or 8 hours.

However, the runtime can be improved as follows. For example, for solar eclipses a significant optimization consists in performing an initial check. We calculate the angular separation at midday (12:00 UTC) at the Equator (*lat* 0°,*lon* 0°); if this separation is $>= 30$, then the chance that it will reach a value smaller than the sum of the apparent radii of the Sun and Moon is low, so we disregard this day as an eclipse candidate. While this reduces our running time to about 1 hour, it remains still high in comparison to existing algorithms.

To drastically reduce our running time, we may check in fewer points around the globe just once per hour. Therefore, we check in every point $P \in S_{longitude} \times S_{latitude}$, where $S_{longitude} = \{0°, 90°, 180°, 270°\}$ and $S_{latitude} = \{0°, 45°, -45°\}$. This completes running in about 2 minutes, but misses 5 lunar and 6 solar eclipses. These are also the points and frequency we use for the dataset which we will use to assess the Machine Learning models described in Sec. 3.2. Meeus' algorithm completes the same task in $\approx$33 ms.

5.2. MACHINE LEARNING METHODS

For each ML classification algorithm discussed in Sect. 3.2, we first selected the hyperparameters of each model in order to achieve the best accuracy coefficient and F1 score (cf. Sect.4) [§]

The training data is composed of the best angular separation values for each day in the 1950–2019 interval, while the test data consists of each day between 2020 and 2100. Because we are interested in the performance of our proposed methods on predicting only the chosen time interval, we do not use cross-validation, but simply validate each day's actual label with its predicted label. We did not use a separate validation set for hyperparameter tuning, because we do not expect the training set to have any issues which may cause biased evaluation of the model, such as over- or under-fitting. The reason for this is that, having a single feature (*e.g.*, the optimal separation, depending on the type of eclipse), small differences are notable and we need to take into account as much data as possible.

Tables 6 and 7 showcase the results for solar and lunar eclipses; for each model, we can observe the number of accurate and inaccurate predictions and the performance (cf. Relation 2) and F1 scores. It can be easily observed that for both solar and lunar eclipses, all models perform well, but two in particular achieve the highest scores,

[§] The relevant hyperparameters were selected as follows: for the *k*NN model we used *k* = 7; for the Decision Tree, maximum depth 2; for the Random Forest: maximum depth 2, 100 estimators, minimum samples split 30, criterion Gini Impurity; for Logistic Regression, we used the default parameters.

Table 6

Performance results of tested classification algorithms for solar eclipses.

Model	T_P	F_N	F_P	T_N	Perf.	F1
*k*NN (k=7)	179 (0.61 %)	3 (0.01%)	21 (0.07%)	29,381 (99.31%)	**0.868**	**0.937**
Decision Tree	179 (0.61 %)	3 (0.01%)	23 (0.08%)	29,379 (99.31%)	0.857	0.932
Random Forest	179 (0.61 %)	3 (0.01%)	21 (0.07%)	29,381 (99.31%)	**0.868**	**0.937**
Logistic Regr.	170 (0.57 %)	12 (0.04%)	20 (0.07%)	29,382 (99.32%)	0.824	0.921

Table 7

Performance results of tested classification algorithms for lunar eclipses.

Model	T_P	F_N	F_P	T_N	Perf.	F1
*k*NN (k=7)	182 (0.62 %)	3 (0.01%)	22 (0.07%)	29,377 (99.30%)	**0.865**	**0.936**
Decision Tree	182 (0.62 %)	3 (0.01%)	23 (0.08%)	29,376 (99.30%)	0.859	0.933
Random Forest	182 (0.62 %)	3 (0.01%)	23 (0.08%)	29,376 (99.30%)	**0.859**	**0.933**
Logistic Regr.	175 (0.59 %)	10 (0.03%)	20 (0.07%)	29,379 (99.31%)	0.838	0.921

namely *k*-Nearest Neighbors and Random Forest, with a performance score > 0.82. The Decision Tree and Logistic Regression classifiers are not far behind, with a score of ≈ 0.85 and ≈ 0.82, respectively. In the case of lunar eclipses, the Decision Tree performs as well as the Random Forest classifier.

It is important to note that the false positives detected by the four models are dates exactly before or after (± 1 day) an exact eclipse prediction. The misprediction usually takes place for eclipses that occur close to midnight (we use UTC throughout this paper), making it more likely to falsely mark the previous or next day as an eclipse.

As for the three false negatives missed by the best-performing models (*k*-NN and Random Forest) they are three penumbral lunar eclipses and three partial solar eclipses, all with very small coverage and barely perceptible visually (cf. Table 8).

The best-performing models only miss ≈ 3 eclipses, while the numerical algorithm that computes the best angular separation with the same frequency and number of points around the globe misses ≈ 5. However, the numerical algorithm does not detect any false positives, unlike our models. This indicates that our numerical method maybe is not perfectly calibrated in order to obtain a balance between false positives and false negatives. By changing the parameter that dictates where the separation between a positive and negative finding is, we can tune the sensitivity of the algorithm based on previous results. In order to achieve this balance, a solution would be manually finding the optimal parameter by binary-searching it (if we obtain too many false positives, we make it more restrictive; if we obtain too many false negatives, we make it more permissive). This highlights the advantage of ML methods, which are

Table 8

Magnitude of eclipses missed by the best-performing models.

Date	Eclipse Type	Magnitude
2054-08-03	Partial solar	0.0655
2069-05-20	Partial solar	0.0879
2083-07-15	Partial solar	0.0168
2060-11-08	Penumbral lunar	-0.938
2078-11-19	Penumbral lunar	-0.91
2092-07-19	Penumbral lunar	-0.90

designed to automatically detect and calibrate this type of parameters.

Also, contrary to our proposed numerical methods (cf. Sect. 3.1.2) the ML algorithms run faster. They need about 15 min to prepare the data (*e.g.*, calculate the best angular separation per day), after which training and testing take less than 4 seconds.

6. CONCLUSION

Since ancient times people have attempted to predict eclipses. Of all the cultures, the Babylonians developed sophisticated methods for recording and predicting the positions of celestial bodies. There are even records that they knew the Saros eclipse cycle.

Currently, we have sophisticated algorithms based on the Sun and Moon's ephemerides which accurately predict when and where eclipses are visible. The question that arises from studying ancient Babylonian sources is "how accurate would be the prediction of eclipses by using knowledge available at their time?" and "could a modern computer learn to predict eclipses similarly to a human ancient astronomer by relying on the dates when eclipses occurred?". In this research, we proposed several simple algorithms and tested their accuracy on solar and lunar eclipses occurring worldwide between 2020–2100 CE. We note here that even some modern day algorithms (*i.e.* Meeus (1998)) miss eclipses and require parameter adjustment. In addition, discrepancies between the NASA eclipse database (Espenak and Meeus (2006)) and the list on *www.timeanddate.com* exist. We have for instance the partial solar eclipse of October 2098 and the penumbral lunar eclipse of July 2027. Both figure in the NASA database as low magnitude eclipses, but not on *www.timeanddate.com*; the lunar eclipse is described as an "almost-eclipse", while the solar eclipse is not mentioned at all. Other low magnitude eclipses are however visible in both cases (*e.g.*, the penumbral lunar eclipse from May 2013).

The first algorithm uses simple arithmetic operations to predict eclipses, requiring as input three dates: a Full/New Moon that is also an eclipse (depending on whether we want to predict solar or lunar eclipses), and one ascending and one descending lunar node. The algorithm accurately predicts 90% of lunar and 81% of solar eclipses. For lunar eclipses it detects ≈ 18 false positives and false negatives; based on the dates we checked, that is $\approx 0.12\%$ falsely detected dates. For solar eclipses it falsely detects $\approx 0.25\%$ dates, out of which ≈ 33 are false positives and false negatives.

The second method treats eclipses as visual phenomena and attempts to check if the geometrical relation between the Sun and Moon (as seen by an observer located in a specific point of the globe), indicative of an eclipse, is validated. This algorithm's accuracy depends on the number of points and moments in time for which we perform the check. For instance if we check three times per hour in approximately 700 points around the globe, we achieve perfect accuracy (no false positives or negatives). However, the process takes about 7 hours. The runtime can be improved by reducing the frequency and number of points but this increases the percentage of missed eclipses, indicating a lack of balance between the false positives and negatives and a need for better tuning of the algorithm's parameters.

The ML models perform very well with the best models (k-Nearest Neighbors Classifier and Random Forest Classifier) accurately predicting 98% of both lunar and solar eclipses. In each case, we miss 3 eclipses and falsely detect ≈ 22 eclipses (which are always ± 1 day from an actual eclipse date).The code for all described methods is available at `https://github.com/marasferdian/eclipse-prediction`.

Our study showed that: (1) by using knowledge available to Babylonians we can predict eclipses with a reasonable level of accuracy; and (2) modern ML algorithms simulating human inference can predict eclipses with almost perfect accuracy.

Acknowledgements. We would like to thank students Alexandru Manafu, Marius Bijan, Petru-Cezar Comsa, and Denisa Ionitoiu from the Computer Science course at Faculty of Mathematics and Computer Science, West University of Timisoara Romania, who during their summer internship worked on the regression based prediction tests and compilation of the results.

REFERENCES

Bretagnon, P., Francou, G.: 1988, Planetary Theories in rectangular and spherical variables: VSOP87 solution. *Astron. Astrophys.* **202**, 309.

Brown, F.C.: 1931, The eclipse in China. *Popular Astronomy* **39**, 567.

Chapront-Touze, M., Chapront, J.: 1983, The lunar ephemeris ELP 2000. *Astron. Astrophys.* **124**, 50.

Espenak, F.: Eclipse predictions. NASA/GSFC Emeritus (Accessed March 29, 2021). https://eclipse.gsfc.nasa.gov/eclipse.html.

Espenak, F., Meeus, J.: 2006, Five millennium canon of solar eclipses: -1999 to +3000. NASA/TP-2006-214141.

Freeth, T.: 2014, Eclipse prediction on the ancient greek astronomical calculating machine known as the antikythera mechanism. *PLOS ONE* **9**(7), 1 – 15. doi:10.1371/journal.pone.0103275.

Goldstein, B.R.: 2002, On the Babylonian discovery of the periods of lunar motion. *Journal for the History of Astronomy* **33**(110), 1 – 13. doi:10.1177/002182860203300102.

Humphreys, C., Waddington, G.: 2017, Solar eclipse of 1207 BC helps to date pharaohs. *Astronomy Geophysics* **58**(5), 5 – 39542. doi:10.1093/astrogeo/atx178. https://doi.org/10.1093/astrogeo/atx178.

Meeus, J.: 1998, *Astronomical algorithms*, Atlantic Books, London.

Montelle, C.: 2016, *The Anaphoricus of Hypsicles of Alexandria*, Brill, Leiden, The Netherlands, 287 – 315. ISBN 9789004315631. doi:https://doi.org/10.1163/9789004315631$_0$11.

Morrison, L.V., Stephenson, F.R.: 2004, Historical values of the Earth's clock error ΔT and the calculation of eclipses. *Journal for the History of Astronomy* **35**(120), 327 – 336. doi:10.1177/002182860403500305.

Müller, A., Guido, S.: 2016, *Introduction to machine learning with python: A guide for data scientists*, O'Reilly Media, Sebastopol CA. ISBN 9781449369903. https://books.google.ro/books?id=1-4lDQAAQBAJ.

Neugebauer, O.: 1975a, Babylonian astronomy. In: *A History of Ancient Mathematical Astronomy Studies in the History of Mathematics and Physical Sciences*, Springer, Berlin Heidelberg, 347 – 555. ISBN 0172-570X.

Neugebauer, O.: 1975b, Theory of eclipses. In: *A History of Ancient Mathematical Astronomy Studies in the History of Mathematics and Physical Sciences*, Springer, Berlin Heidelberg, 118 – 145. ISBN 0172-570X.

Pingalkar, A.: 2020, Prediction of solar eclipses using extreme gradient boost algorithm. *International Journal for Research in Applied Science Engineering Technology* **8**, 1353 – 1357.

Steele, J.M.: 2005, Babylonian eclipse: Ptolemy, Babylon and the rotation of the Earth. *Astronomy and Geophysics* **46**, 5 – 11515. doi:10.1111/j.1468-4004.2005.46511.x.

Steele, J.: 2017, *Rising time schemes in babylonian astronomy*, Springer, New York.

Stephenson, F.R.: 2006, Babylonian Timings of Eclipse Contacts and the Study of the Earth's Past Rotation. *Journal of Astronomical History and Heritage* **9**(2), 145 – 150.

Vaquero, J., Malville, M.: 2014, On the solar corona petroglyph in the chaco canyon. *Journal of Mediterranean Archaeology and Archaeometry* **14**(3), 189 – 196.

Wikipedia contributors: 2021, *List of 21st-century lunar eclipses — Wikipedia, the free encyclopedia*. [Online; accessed 7-May-2021]. https://en.wikipedia.org/w/index.php?title=List_of_21st-century_lunar_eclipses&oldid=997779007.

Received on 23 April 2021